General Mathematics Units 3 & 4

Chapter 11 — Flow, matching and scheduling problems

Aboriginal and Torres Strait Islander content has been excluded as accuracy cannot be guaranteed, and it is better to be respectful than cover the entire curriculum inaccurately.

Significant effort has been made to ensure this content is legal. Please send any areas of concern to admin@vwx.au with appropriate document/legal references so errors can be verified and changes be made.

Contents

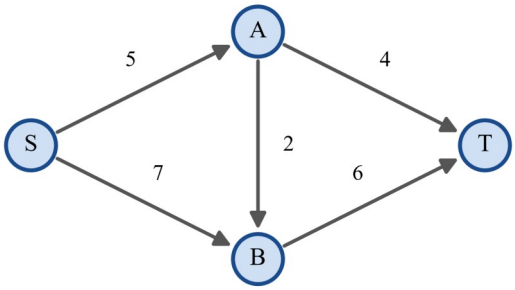
Theory

Many practical situations involve something that **moves through a network** in a definite direction: water through a system of pipes, cars along one-way roads, or data through a communications network. Each connection can only carry so much per unit of time, and we want to know the greatest amount that can travel from a starting point to a destination. This is a **flow problem**.

Modelling a flow problem.

A flow problem is modelled by a **directed network** (a **digraph** with weights):

- each **directed edge** (arc) points in the direction the material can travel;
- the weight on an edge is its **capacity** — the maximum rate at which material can pass along it (litres per minute, cars per hour, megabits per second, and so on);
- one vertex is the **source** — where all the flow originates (nothing flows *into* it);
- one vertex is the **sink** — where all the flow collects (nothing flows *out* of it);
- the remaining vertices are **intermediate** vertices, which merely pass flow on.



In the network above the source is S and the sink is T. The pipe $S \to A$ has capacity 5, the pipe $S \to B$ has capacity 7, and so on.

The two rules a flow must obey.

A **flow** is an assignment of an actual rate to every edge. To be possible it must satisfy:

- **the capacity rule** — the flow along any edge cannot exceed that edge's capacity; and
- **the conservation rule** — at every *intermediate* vertex the total flow in equals the total flow out (nothing is created or stored there).

The **value** of a flow is the total leaving the source, which by conservation is also the total arriving at the sink. The **maximum flow** is the largest value that any possible flow can have.

Finding a flow by inspection.

For a small network the maximum flow can be built up by sending flow along paths from source to sink. Along any single path the amount that can be pushed is limited by the **smallest capacity** on that path — its **bottleneck**. In the network above:

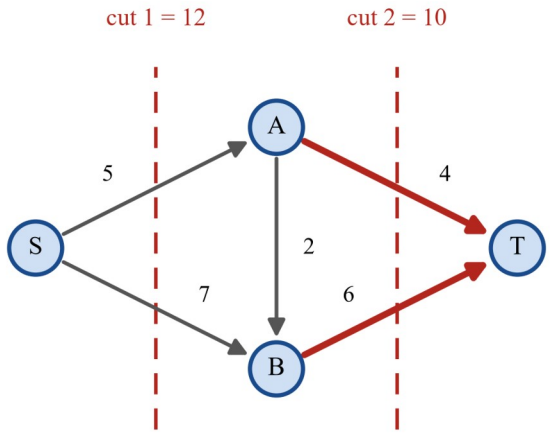
- the path $S \to A \to T$ has capacities 5 and 4, so it can carry 4;
- the path $S \to B \to T$ has capacities 7 and 6, so it can carry 6.

Together these carry $4 + 6 = 10$. After this, every edge into T is full ($A \to T$ carries 4 of 4, and $B \to T$ carries 6 of 6), so no further flow can reach the sink. The maximum flow is 10.

Cuts.

Building a flow shows a value *can* be achieved; to be sure it is the *maximum* we use a **cut**. A cut divides all the vertices into two groups — one containing the **source** and one containing the **sink** — and so "cuts" the network into a source side and a sink side. The **capacity of the cut** is the sum of the capacities of the edges that cross **forward**, that is, that run **from the source side to the sink side**. Edges that run backwards (from the sink side to the source side) are **not** counted.

The diagram shows two cuts of the network above.



- **Cut 1** separates $\{S\}$ from $\{A, B, T\}$. The forward edges are $S \to A$ and $S \to B$, so its capacity is $5 + 7 = 12$.
- **Cut 2** separates $\{S, A, B\}$ from $\{T\}$. The forward edges are $A \to T$ and $B \to T$, so its capacity is $4 + 6 = 10$.

Because **every** unit of flow must travel from the source side to the sink side, it must cross every cut. Hence **no flow can ever exceed the capacity of any cut** — each cut is an upper limit on the maximum flow. The tighter the cut, the better the limit.

The maximum-flow minimum-cut theorem.

The key result ties the two ideas together:

$$\text{maximum flow} = \text{capacity of the minimum cut},$$

where the **minimum cut** is the cut of smallest capacity. In words: the greatest flow the network can carry is exactly the capacity of its tightest cut — the true bottleneck of the whole network. For the network above the smallest cut we found is Cut 2, of capacity 10, and this agrees with the flow of 10 we built by inspection.

Using the theorem to solve larger problems.

For a network too large to exhaust by trial, the theorem gives a method: **find the minimum cut**. To do this by inspection,

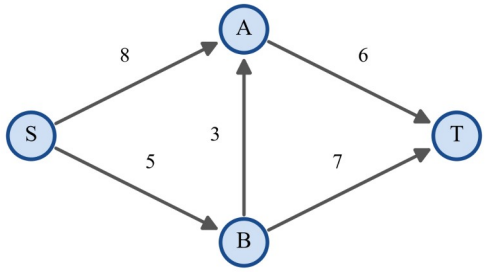
1. list several cuts — always try the cut around the source, the cut around the sink, and any cut through an obvious narrow "waist" of the network;
2. for each cut, add the capacities of the forward edges only;
3. the smallest total is the minimum cut, and by the theorem it equals the maximum flow.

Two cautions when finding a cut's capacity. First, a cut must completely separate source from sink — if any path from S to T avoids the cut, it is **not** a valid cut. Second, remember to **ignore backward edges**: only edges pointing from the source side to the sink side are added.

Worked examples

Example 1 — scaffolded

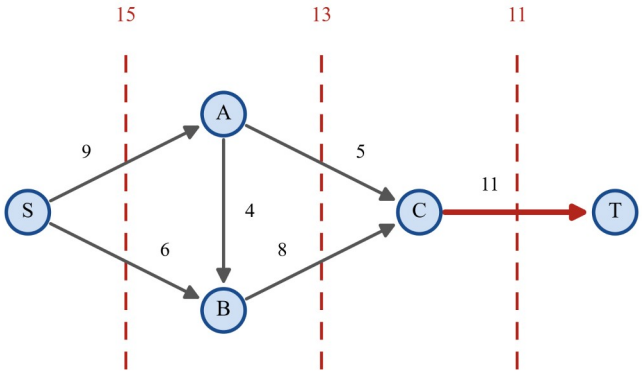
For the network below, find the maximum flow from the source S to the sink T. The numbers are the capacities of the pipes in litres per minute.



Working	Comment
Send flow along $S \to A \to T$: capacities 8 and 6, bottleneck $min(8,6)=6$.	The most a path can carry is its smallest capacity.
Send flow along $S \to B \to T$: capacities 5 and 7, bottleneck $min(5,7)=5$.	A second source-to-sink path.
Flow so far $=6+5=11$.	Value of a flow = total leaving S.
Now $A \to T$ carries 6 (full) and $B \to T$ carries 5 of 7; the only unused way into A is $B \to A$, but A's only exit $A \to T$ is already full.	No source-to-sink path has spare capacity on every edge.
So no more flow can be added: maximum flow $=11$ L/min.	Inspection is complete.
Check with a cut. Put $\{S, A\}$ on the source side and $\{B, T\}$ on the sink side. Forward edges: $S \to B(5)$ and $A \to T(6)$; the edge $B \to A$ runs backward, so it is ignored. Cut capacity $=5+6=11$.	A cut of capacity 11 equals the flow, so by the maximum-flow minimum-cut theorem 11 is the maximum.

Example 2 — scaffolded

The network below shows capacities in cars per minute along one-way roads from S to T. Find the capacity of each of the three cuts marked, and hence state the maximum flow.

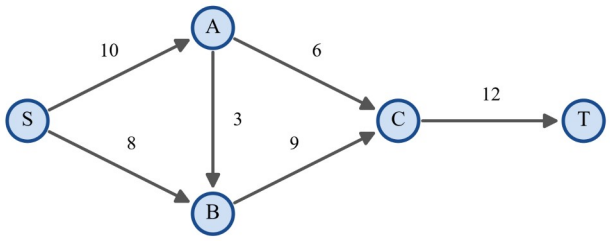


Working	Comment
Cut 1 separates $\{S\}$ from the rest. Forward edges $S \to A(9)$, $S \to B(6)$. Capacity $=9+6=15$.	The cut around the source.

Working	Comment
Cut 2 separates $\{S, A, B\}$ from $\{C, T\}$. Forward edges $A \to C(5)$, $B \to C(8)$. Capacity $= 5 + 8 = 13$.	The edge $A \to B$ lies **inside** the source side, so it does not cross the cut.
Cut 3 separates $\{S, A, B, C\}$ from $\{T\}$. Forward edge $C \to T(11)$. Capacity $= 11$.	The cut around the sink; everything funnels through $C \to T$.
Minimum cut $= min(15, 13, 11) = 11$.	The smallest cut is the true bottleneck.
By the maximum-flow minimum-cut theorem, maximum flow $= 11$ cars per minute.	Max flow = capacity of the minimum cut.

Example 3 — unscaffolded

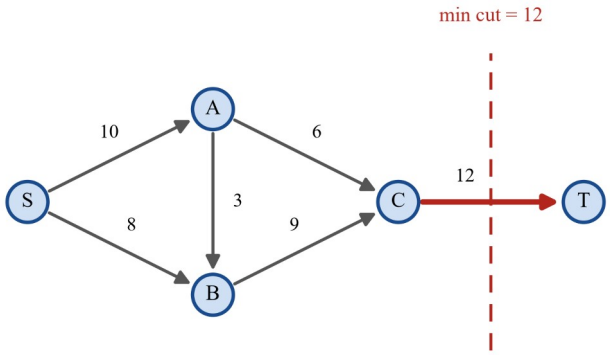
A reservoir S supplies water to a town T through a system of pipes, passing through pumping stations A and B and a junction C. The capacities, in kilolitres per minute, are shown below. Find the maximum rate at which water can be delivered to the town.



Every pipe eventually funnels into the junction C, and the only pipe out of C is $C \to T$, with capacity 12. Consider the cut that isolates the town: put $\{S, A, B, C\}$ on the source side and $\{T\}$ on the sink side. The only forward edge is $C \to T$, so this cut has capacity 12.

No flow can exceed this cut, so the delivery rate is at most 12. This rate can actually be achieved: the pipes into C have total capacity $A \to C(6) + B \to C(9) = 15 \geq 12$, and the stations can be supplied from the reservoir since $S \to A(10)$ and $S \to B(8)$ comfortably exceed what is needed (with $A \to B$ available to pass 3 across if required). So a flow of 12 into C, and out along $C \to T$, is possible.

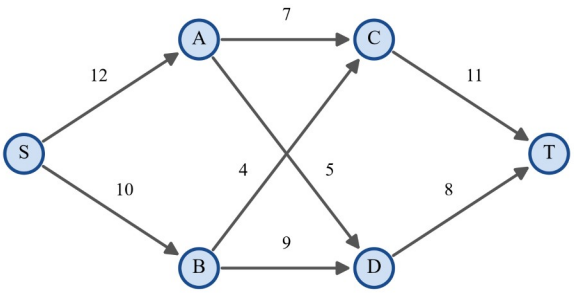
Every other cut is larger — for instance the cut around the source has capacity $10 + 8 = 18$ — so the minimum cut is the single pipe $C \to T$, shown below.



∴ the maximum rate of delivery to the town is 12 kilolitres per minute.

Example 4 — unscaffolded

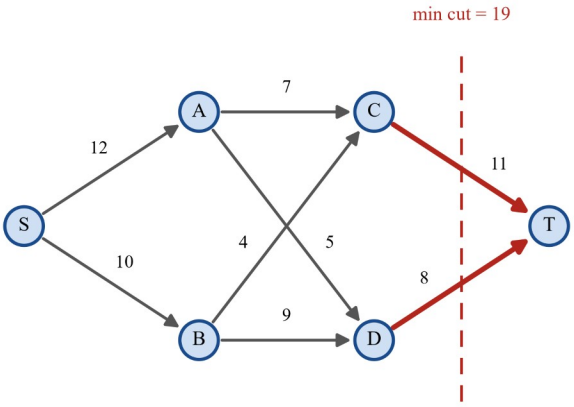
Find the maximum flow from S to T in the network below, where capacities are in megabits per second.



The network is too tangled to be certain by inspection alone, so we hunt for the minimum cut, testing the natural candidates.

- Cut around the source $\{S\}$: forward edges $S \to A(12)$, $S \to B(10)$; capacity $12+10=22$.
- Cut $\{S, A, B\}$ against $\{C, D, T\}$: forward edges $A \to C(7)$, $A \to D(5)$, $B \to C(4)$, $B \to D(9)$; capacity $7+5+4+9=25$.
- Cut around the sink $\{S, A, B, C, D\}$ against $\{T\}$: forward edges $C \to T(11)$, $D \to T(8)$; capacity $11+8=19$.

The smallest so far is the sink cut, 19. Testing the remaining cuts gives nothing smaller (for example, separating $\{S, A, B, D\}$ from $\{C, T\}$ also gives $7+4+8=19$), so the minimum cut has capacity 19.



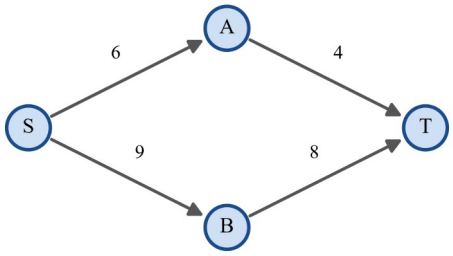
A flow of 19 is achievable: send 11 into C and 8 into D, which the incoming edges can supply ($A \to C$ and $B \to C$ give up to 11 into C; $A \to D$ and $B \to D$ give up to 14 into D), and the source can provide 19 since $S \to A(12)+S \to B(10)=22$.

$\therefore$ the maximum flow is 19 megabits per second, limited by the two edges $C \to T$ and $D \to T$ into the sink.

Practice questions

Scaffolded

Q1. In the network below the capacities are in litres per minute, with source S and sink T.



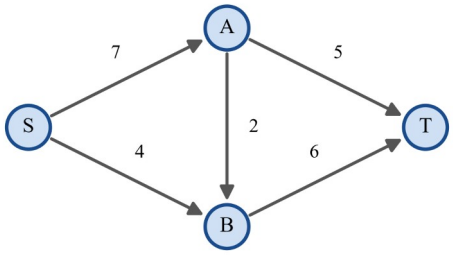
(a) Find the bottleneck (smallest capacity) of the path $S \to A \to T$.

(b) Find the bottleneck of the path $S \to B \to T$.

(c) Hence state the maximum flow.

(d) Find the capacity of the cut that separates $\{S, A, B\}$ from $\{T\}$, and confirm it equals your answer to (c).

Q2. The network below has capacities in cars per minute.

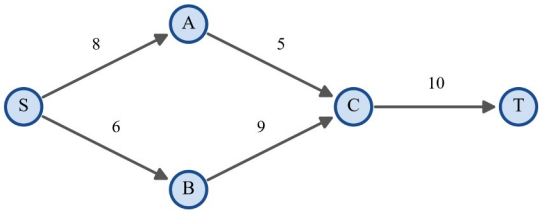


(a) Write down the two edges that leave the source S, and add their capacities to find the capacity of the cut around the source.

(b) Find the capacity of the cut that separates $\{S, A, B\}$ from $\{T\}$.

(c) State which cut is the minimum cut, and hence write down the maximum flow.

Q3. For the network below (capacities in megalitres per day):

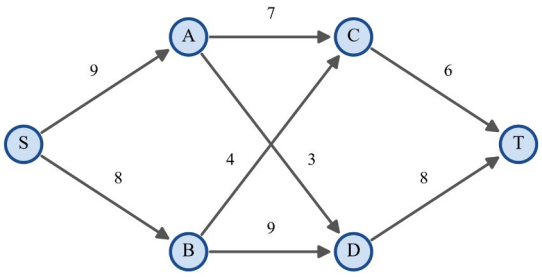


(a) Explain why every unit of flow must pass along the edge $C \to T$.

(b) Write down the capacity of the edge $C \to T$.

(c) Hence state the maximum flow from S to T.

Q4. The network below has capacities in tonnes per hour.

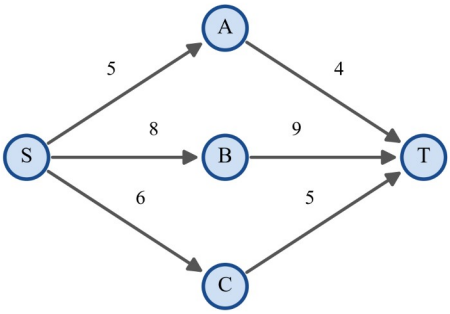


(a) List the forward edges of the cut that separates $\{S, A, B, C, D\}$ from $\{T\}$.

(b) Find the capacity of that cut.

(c) Given that no other cut is smaller, state the maximum flow.

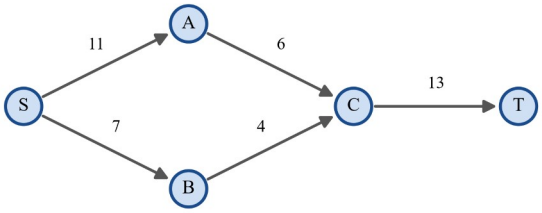
Q5. Three independent pipelines run from a depot S to a terminal T through booster stations A, B and C. Capacities are in kilolitres per hour.



(a) Find the bottleneck of each of the three paths $S \to A \to T$, $S \to B \to T$ and $S \to C \to T$.

(b) Hence find the maximum flow from S to T.

Q6. For the network below (capacities in gigabits per second):



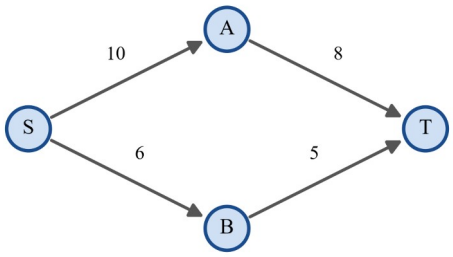
(a) Find the capacity of the cut around the source $\{S\}$.

(b) Find the capacity of the cut that separates $\{S, A, B\}$ from $\{C, T\}$.

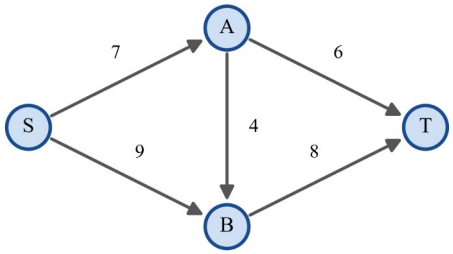
(c) State which of these two cuts is the minimum, and hence write down the maximum flow.

Unscaffolded

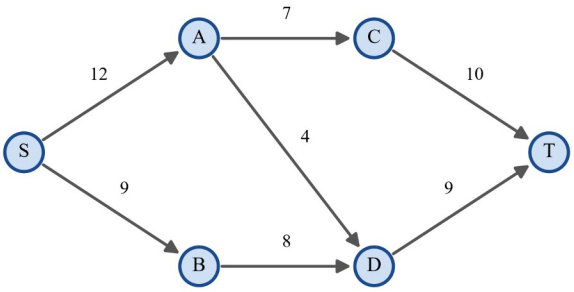
Q7. Find the maximum flow from S to T in the network below (capacities in litres per second).



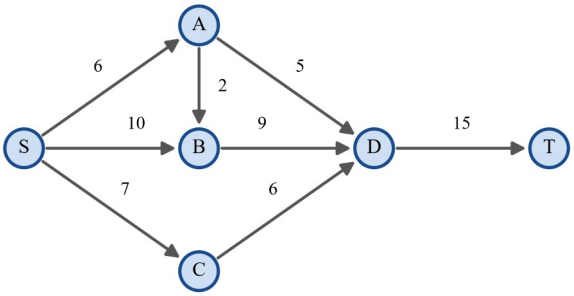
Q8. Find the maximum flow from S to T in the network below (capacities in cars per minute).



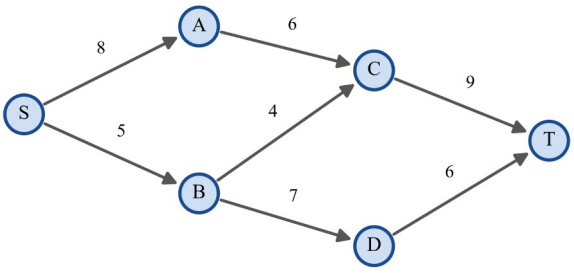
Q9. Find the maximum flow from S to T in the network below, and state the edges that form the minimum cut (capacities in megabits per second).



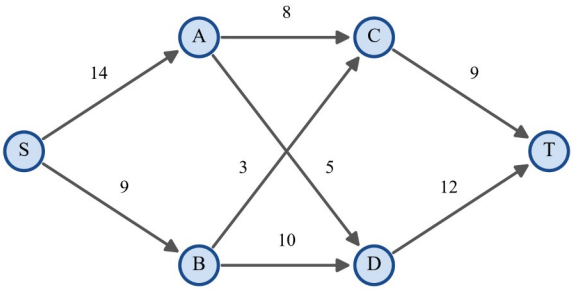
Q10. Water flows from a reservoir S to a town T through the network below, with capacities in kilolitres per minute. Find the maximum rate of delivery, and identify the single pipe that acts as the bottleneck.



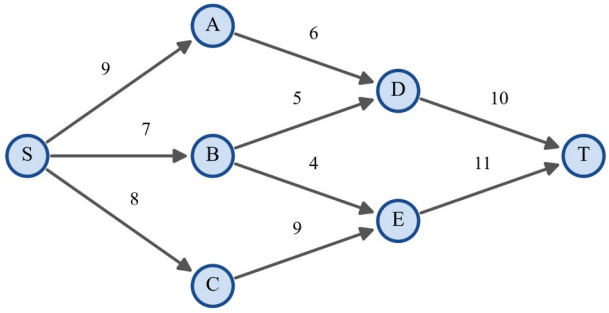
Q11. Find the maximum flow from S to T in the network below, and state the minimum cut (capacities in tonnes per hour).



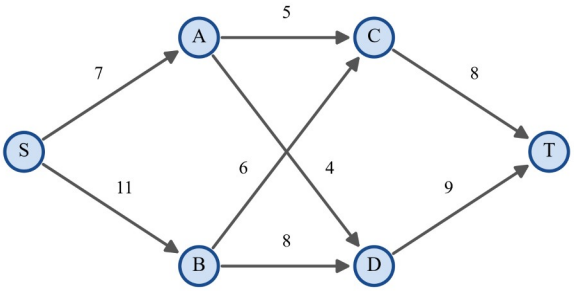
Q12. Find the maximum flow from S to T in the network below (capacities in cars per minute).



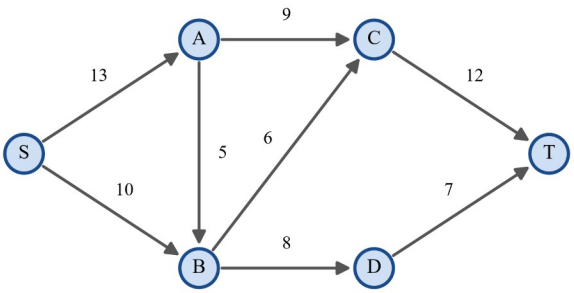
Q13. Find the maximum flow from S to T in the network below (capacities in megalitres per day).



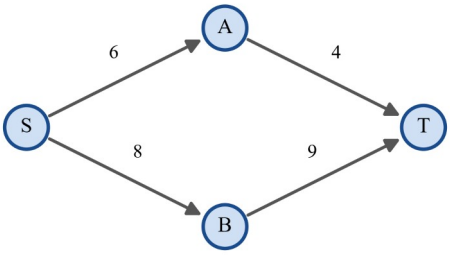
Q14. Find the maximum flow from S to T in the network below, and state the capacity of the minimum cut (capacities in gigabits per second).



Q15. Find the maximum flow from S to T in the network below (capacities in tonnes per hour).



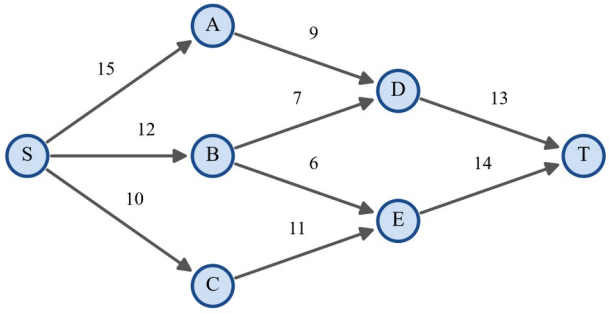
Q16. For the network below (capacities in litres per minute):



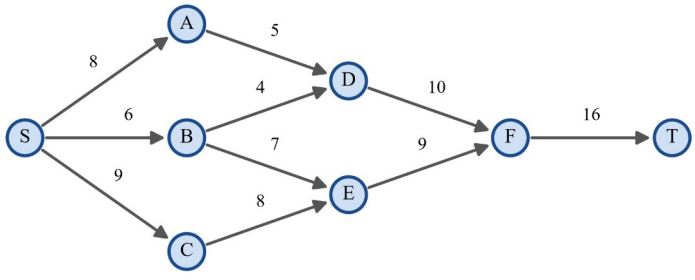
(a) Show that the cut around the sink $\{S, A, B\}|\{T\}$ has capacity 13.

(b) Find a cut of smaller capacity, and hence state the maximum flow.

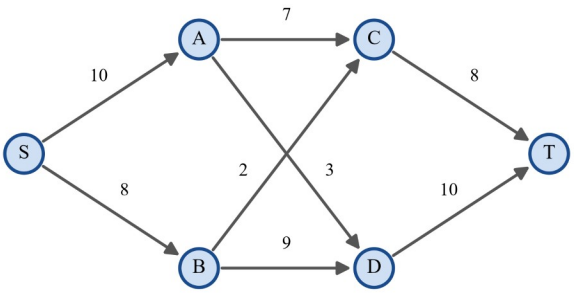
Q17. A data network carries traffic from a server S to a client T (capacities in megabits per second). Find the maximum flow.



Q18. Find the maximum flow from S to T in the network below, and identify the single edge that forms the minimum cut (capacities in kilolitres per minute).



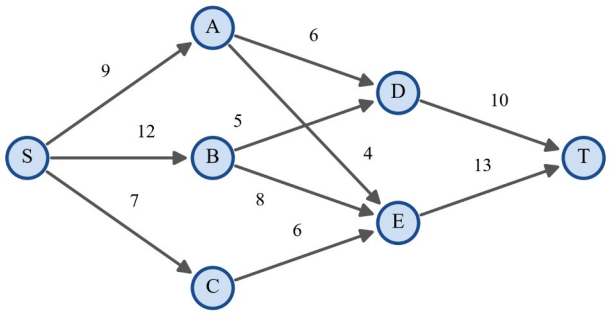
Q19. For the network below (capacities in cars per minute):



(a) Find the capacity of the cut around the source.

(b) Explain why this cut is the minimum cut, and hence state the maximum flow.

Q20. A water authority models its supply network as shown below, with capacities in megalitres per day. Find the maximum flow from S to T, and state the minimum cut.



Answers

Q1. (a) 4; (b) 8; (c) 12; (d) $A \to T(4) + B \to T(8) = 12$, equal to (c). **Q2.** (a) $S \to A(7)$, $S \to B(4)$, capacity 11; (b) $5 + 6 = 11$; (c) both equal 11; maximum flow 11 cars/min. **Q3.** (a) $C \to T$ is the only edge into T; (b) 10; (c) 10 ML/day. **Q4.** (a) $C \to T$ and $D \to T$; (b) $6 + 8 = 14$; (c) 14 tonnes/hour. **Q5.** (a) 4, 8, 5; (b) $4 + 8 + 5 = 17$ kL/hour. **Q6.** (a) $11 + 7 = 18$; (b) $6 + 4 = 10$; (c) the second cut, 10; maximum flow 10 Gbit/s. **Q7.** 13 L/s. **Q8.** 14 cars/min. **Q9.** 16 Mbit/s; minimum cut $A \to C(7)$ and $D \to T(9)$. **Q10.** 15 kL/min; bottleneck the pipe $D \to T$. **Q11.** 11 tonnes/hour; minimum cut $S \to B(5)$ and $A \to C(6)$. **Q12.** 21 cars/min. **Q13.** 21 ML/day. **Q14.** 17 Gbit/s; minimum cut capacity 17. **Q15.** 19 tonnes/hour. **Q16.** (a) $A \to T(4) + B \to T(9) = 13$; (b) cut $\{S, A\} | \{B, T\}$ has capacity $S \to B(8) + A \to T(4) = 12$; maximum flow 12 L/min. **Q17.** 27 Mbit/s. **Q18.** 16 kL/min; minimum cut the single edge $F \to T$. **Q19.** (a) $10 + 8 = 18$; (b) all flow must leave S along these two edges and 18 can be delivered, so it is the minimum cut; maximum flow 18 cars/min. **Q20.** 23 ML/day; minimum cut $D \to T(10)$ and $E \to T(13)$.

Fully-worked solutions

Q1. (a) Along $S \to A \to T$ the capacities are 6 and 4, so the bottleneck is $min(6, 4) = 4$. (b) Along $S \to B \to T$ the capacities are 9 and 8, so the bottleneck is $min(9, 8) = 8$. (c) The two paths are edge-disjoint and between them fill both edges into T, giving a flow of $4 + 8 = 12$; no more can reach T, so the maximum flow is 12 L/min. (d) The cut $\{S, A, B\} | \{T\}$ has forward edges $A \to T(4)$ and $B \to T(8)$, of total capacity 12. This equals the flow, confirming by the maximum-flow minimum-cut theorem that 12 is the maximum.

Q2. (a) The edges leaving S are $S \to A(7)$ and $S \to B(4)$, so the cut around the source has capacity $7 + 4 = 11$. (b) The cut $\{S, A, B\} | \{T\}$ has forward edges $A \to T(5)$ and $B \to T(6)$ (the edge $A \to B$ is inside the source side), of capacity $5 + 6 = 11$. (c) Both cuts equal 11, and no cut can be smaller than the source cut here, so the minimum cut is 11 and the maximum flow is 11 cars per minute.

Q3. (a) The only edge directed into T is $C \to T$, so every unit of flow reaching the sink must travel along it. (b) Its capacity is 10. (c) The cut isolating T therefore has capacity 10, and this is clearly the smallest cut (the source cut is $8 + 6 = 14$), so the maximum flow is 10 ML/day.

Q4. (a) The edges directed into T are $C \to T$ and $D \to T$, so these are the forward edges of the cut $\{S, A, B, C, D\} | \{T\}$. (b) Their capacities are 6 and 8, so the cut capacity is $6 + 8 = 14$. (c) Since no other cut is smaller, the minimum cut is 14 and the maximum flow is 14 tonnes per hour.

Q5. (a) The three paths are independent. $S \to A \to T$ has capacities 5 and 4, bottleneck 4; $S \to B \to T$ has 8 and 9, bottleneck 8; $S \to C \to T$ has 6 and 5, bottleneck 5. (b) Because the paths share no edges, their flows simply add: $4 + 8 + 5 = 17$. (Equivalently, the minimum cut takes the smaller edge of each path, $4 + 8 + 5 = 17$.) The maximum flow is 17 kL/hour.

Q6. (a) The edges leaving S are $S \to A(11)$ and $S \to B(7)$, so the source cut is $11 + 7 = 18$. (b) The cut $\{S, A, B\} | \{C, T\}$ has forward edges $A \to C(6)$ and $B \to C(4)$, capacity $6 + 4 = 10$. (c) The second cut, 10, is the smaller, and it is the minimum cut (everything must pass through the two edges into C), so the maximum flow is 10 Gbit/s.

Q7. The two paths $S \to A \to T$ and $S \to B \to T$ have bottlenecks $min(10, 8) = 8$ and $min(6, 5) = 5$. They fill both edges into T, giving $8 + 5 = 13$. The cut around the sink, $A \to T(8) + B \to T(5) = 13$, confirms this, so the maximum flow is 13 L/s.

Q8. Send 6 along $S \to A \to T$ and 8 along $S \to B \to T$; this fills the two edges into T ($A \to T(6)$ and $B \to T(8)$), and the edge $A \to B$ is not needed. The cut $\{S, A, B\} | \{T\}$ has capacity $6 + 8 = 14$, matching the flow, so the maximum flow is 14 cars per minute.

Q9. Test cuts. The source cut is $12 + 9 = 21$; the sink cut is $C \to T(10) + D \to T(9) = 19$. A smaller cut is $\{S, A, B, D\} | \{C, T\}$, whose forward edges are $A \to C(7)$ and $D \to T(9)$ (the edges $A \to D$ and $B \to D$ lie inside

the source side): capacity $7+9=16$. No cut is smaller, so the minimum cut is 16, formed by the edges $A \to C$ and $D \to T$, and the maximum flow is 16 Mbit/s.

Q10. Every path funnels into D, and the only edge out of D is $D \to T$, with capacity 15. The cut isolating T therefore has capacity 15. The edges into D can supply this ($A \to D(5)+B \to D(9)+C \to D(6)=20 \geq 15$) and the source can feed them ($S \to A(6)+S \to B(10)+S \to C(7)=23$), so 15 is achievable. The maximum rate of delivery is 15 kL/min, with the pipe $D \to T$ as the bottleneck.

Q11. The source cut is $8+5=13$ and the sink cut is $C \to T(9)+D \to T(6)=15$. Consider instead the cut $\{S,A\}|\{B,C,D,T\}$: its forward edges are $S \to B(5)$ and $A \to C(6)$ (A's only exit is $A \to C$), of capacity $5+6=11$. No smaller cut exists, so the minimum cut is 11, formed by $S \to B$ and $A \to C$, and the maximum flow is 11 tonnes per hour.

Q12. The sink cut $\{S,A,B,C,D\}|\{T\}$ has forward edges $C \to T(9)$ and $D \to T(12)$, capacity $9+12=21$. Check this is achievable: into C we can send $A \to C(8)+B \to C(3)=11 \geq 9$, and into D we can send $A \to D(5)+B \to D(10)=15 \geq 12$, while the source provides $14+9=23 \geq 21$. Every other cut is at least 21 (the source cut is 23), so the maximum flow is 21 cars per minute.

Q13. The source cut $\{S\}$ has capacity $9+7+8=24$. Try $\{S,A\}|\{B,C,D,E,T\}$: forward edges are $S \to B(7)$, $S \to C(8)$ and $A \to D(6)$ (since A's only exit is $A \to D$), of capacity $7+8+6=21$. Testing the remaining cuts (for example the sink cut $D \to T(10)+E \to T(11)=21$) gives nothing smaller, so the minimum cut is 21 and the maximum flow is 21 ML/day.

Q14. The sink cut has forward edges $C \to T(8)$ and $D \to T(9)$, capacity $8+9=17$. Into C the edges $A \to C(5)+B \to C(6)=11 \geq 8$ and into D the edges $A \to D(4)+B \to D(8)=12 \geq 9$ can supply the required amounts, and the source provides $7+11=18 \geq 17$, so 17 is achievable. As the source cut is 18 and no cut is smaller than 17, the minimum cut capacity is 17 and the maximum flow is 17 Gbit/s.

Q15. The sink cut $\{S,A,B,C,D\}|\{T\}$ has forward edges $C \to T(12)$ and $D \to T(7)$, capacity $12+7=19$. Into C we can send $A \to C(9)+B \to C(6)=15 \geq 12$, and $B \to D(8) \geq 7$ feeds D; the source supplies $13+10=23$, with $A \to B(5)$ available to balance, so 19 is achievable. No smaller cut exists, so the maximum flow is 19 tonnes per hour.

Q16. (a) The edges into T are $A \to T(4)$ and $B \to T(9)$, so the cut $\{S,A,B\}|\{T\}$ has capacity $4+9=13$. (b) The path through A is throttled to 4 by $A \to T$, so pairing A with the source gives a tighter cut: put $\{S,A\}$ on the source side and $\{B,T\}$ on the sink side. The forward edges are $S \to B(8)$ and $A \to T(4)$, of capacity $8+4=12$. This is smaller than 13, and it is the minimum, so the maximum flow is 12 L/min.

Q17. The sink cut has forward edges $D \to T(13)$ and $E \to T(14)$, capacity $13+14=27$. Into D the edges $A \to D(9)+B \to D(7)=16 \geq 13$ and into E the edges $B \to E(6)+C \to E(11)=17 \geq 14$ can supply the amounts needed, and the source provides $15+12+10=37 \geq 27$, so 27 is achievable. The source cut is 37 and no cut is smaller than 27, so the maximum flow is 27 Mbit/s.

Q18. Every path funnels into F, whose only exit is $F \to T$ with capacity 16. The cut isolating T therefore has capacity 16. The edges into F can supply this ($D \to F(10)+E \to F(9)=19 \geq 16$), and these in turn can be fed from the source ($S \to A(8)+S \to B(6)+S \to C(9)=23$), so 16 is achievable. The minimum cut is the single edge $F \to T$, and the maximum flow is 16 kL/min.

Q19. (a) The edges leaving S are $S \to A(10)$ and $S \to B(8)$, so the source cut has capacity $10+8=18$. (b) All flow out of the source must use these two edges, so no flow can exceed 18; and 18 can be delivered, because the edges out of A ($A \to C(7)+A \to D(3)=10$) and out of B ($B \to C(2)+B \to D(9)=11$) can carry everything the source sends, while the sink absorbs it ($C \to T(8)+D \to T(10)=18$). Hence the source cut is the minimum cut and the maximum flow is 18 cars per minute.

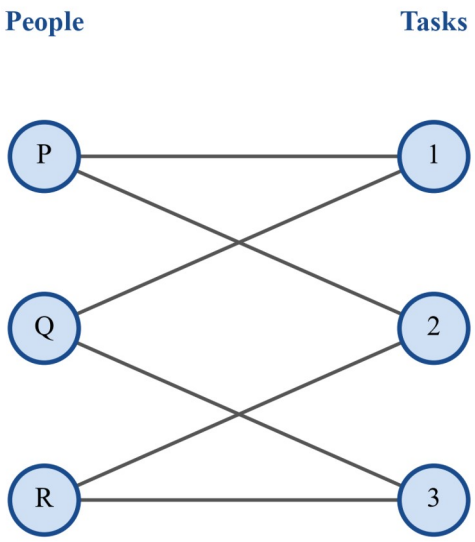
Q20. The sink cut $\{S, A, B, C, D, E\}|\{T\}$ has forward edges $D \to T(10)$ and $E \to T(13)$, capacity $10+13=23$. Into D the edges $A \to D(6)+B \to D(5)=11 \geq 10$, and into E the edges $A \to E(4)+B \to E(8)+C \to E(6)=18 \geq 13$, can supply the required amounts, and the source provides $9+12+7=28 \geq 23$, so 23 is achievable. Every other cut is at least this large (the source cut is 28), so the minimum cut is formed by $D \to T$ and $E \to T$, and the maximum flow is 23 ML/day.

Theory

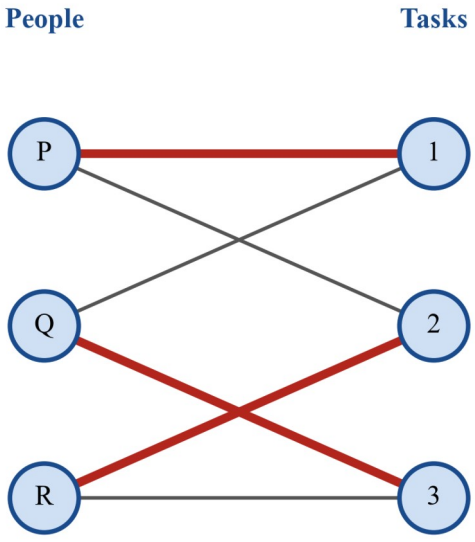
Many practical problems ask us to pair the members of one group with the members of another: workers with machines, players with positions, deliveries with drivers. This section shows how to model such a problem with a **bipartite graph**, and how to find the **cheapest** (or most valuable) pairing when every possible pairing has a cost, time or value attached.

Matching problems and bipartite graphs.

A **bipartite graph** is a graph whose vertices split into **two separate groups**, with every edge joining a vertex in one group to a vertex in the other. No edge joins two vertices of the same group. In a matching problem the two groups are, for example, a set of **people** and a set of **tasks**, and an edge is drawn whenever that person *can* do that task.



A **matching** is a set of edges chosen so that **each vertex is used at most once** — no person is given two tasks and no task is given to two people. A **complete matching** (also called a **perfect matching**) pairs **every** person with a task and **every** task with a person; this is only possible when the two groups are the same size. The highlighted edges below form one complete matching of the graph above.



The same information can be written in **tabular (matrix) form**: a table with the people as rows and the tasks as columns, and a tick (or a 1) wherever that person can do that task. For the graph above:

Person	Task 1	Task 2	Task 3
P	✓	✓	
Q	✓		✓
R		✓	✓

A matching chooses one tick in each row so that no column is used twice. The graph form and the table form carry exactly the same information; either may be given in a problem.

Solving a matching by inspection.

For a small matching problem a complete matching can usually be found by **inspection**. The key idea is to look for a **forced** choice — a person who can do only **one** task, or a task that only **one** person can do — make that pairing, cross out that person and task, and repeat with what remains. If at some stage a person has no task left (or a group of people can, between them, reach too few tasks) then **no complete matching exists**.

The allocation (assignment) problem.

Often each pairing has a **cost** — a price, a time, or a distance. The **allocation problem** (or **assignment problem**) is to pair every person with exactly one task, and every task with exactly one person, so that the **total cost is as small as possible**. The data are given as a square **cost matrix**: entry in row i, column j is the cost of giving task j to person i. For example:

Worker	Job 1	Job 2	Job 3
A	9	11	14
B	6	15	13
C	12	13	6

With n people and n tasks there are $n!$ possible allocations, which grows very quickly ($3! = 6$, but $5! = 120$ and $8! = 40\,320$), so testing every allocation is impractical for all but the smallest problems. A systematic method is needed.

The Hungarian algorithm.

The **Hungarian algorithm** finds a minimum-cost allocation. It rests on one fact: if a **constant is subtracted from every entry of a row** (or of a column) of the cost matrix, the allocation that is cheapest **does not change** — every complete allocation uses exactly one entry from that row (or column), so all totals drop by the same constant. The algorithm creates **zeros** by such subtractions until a complete allocation using only zero-cost entries can be found; that allocation is optimal. The steps are:

1. **Row reduction.** In each row, subtract the smallest entry in that row from every entry of the row. Each row now contains at least one zero.
2. **Column reduction.** In each column of the new matrix, subtract the smallest entry in that column from every entry of the column. Each column now contains at least one zero.
3. **Cover the zeros.** Cover all of the zeros using the **smallest possible number of horizontal and vertical lines**. If the least number of lines needed equals n (the number of rows), an optimal allocation of zeros is possible — go to step 5. Otherwise go to step 4.
4. **Adjust the matrix.** Find the smallest entry **not covered** by any line. **Subtract** it from every uncovered entry, and **add** it to every entry covered by **two** lines (the intersections). Entries covered by a single line are unchanged. Then return to step 3.
5. **Make the allocation.** Choose a set of zeros, one in each row and one in each column. A good order is to start with a row or column containing a **single** zero, allocate it, cross out its row and column, and repeat. The chosen zeros give the optimal pairing; read the **total cost from the original matrix**.

Maximisation problems.

Some problems ask for the allocation that **maximises** a total — for example the greatest total profit, score or output. The Hungarian algorithm minimises, so first **convert** the problem to a minimisation: find the **largest** entry in the whole matrix and **subtract every entry from it**. Applying the algorithm to this new matrix gives the allocation of greatest value; as always, read the **total value from the original matrix**.

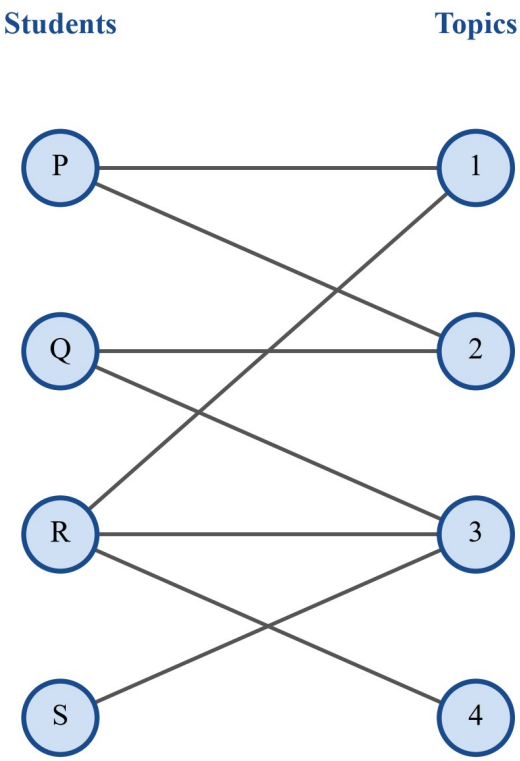
Interpreting the result.

The algorithm returns which person is paired with which task. Always answer in the context of the problem — "worker A should be given job 3" — and state the **total** cost or value obtained from the original figures. A problem may have **more than one optimal allocation**, but they all share the **same** minimum total (or maximum total). Small matchings and allocations can also be checked by inspection; the algorithm is what makes larger ones reliable.

Worked examples

Example 1 — scaffolded

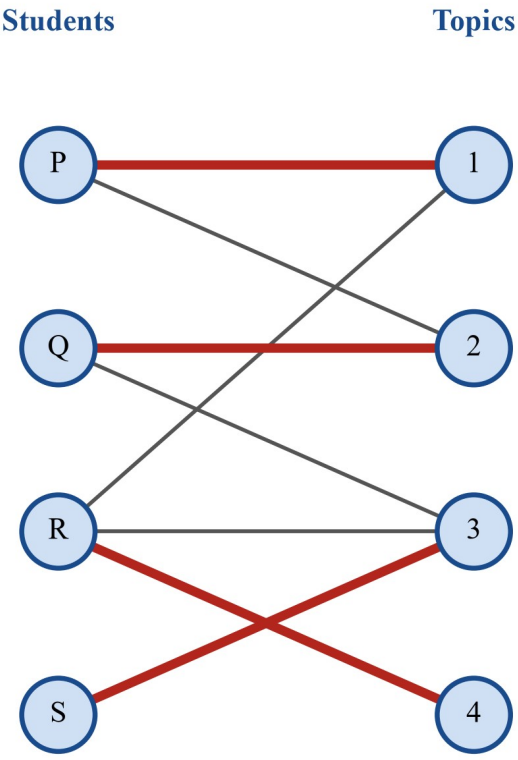
Four students P, Q, R, S are each to present **one** of four topics 1, 2, 3, 4. The bipartite graph shows which topics each student is willing to present.



Find a complete matching of students to topics.

Working	Comment
Student S is joined only to topic 3, so S must take topic 3.	Start with a **forced** choice: S can present only one topic.
Topic 3 is now used. Student Q is joined to topics 2 and 3; topic 3 is gone, so Q takes topic 2.	Cross out topic 3, then look for the next forced choice.
Topic 2 is now used. Student P is joined to topics 1 and 2; topic 2 is gone, so P takes topic 1.	Cross out topic 2.
Student R is joined to topics 1, 3, 4; topics 1 and 3 are gone, so R takes topic 4.	The last student takes the remaining topic.

The complete matching is P–1, Q–2, R–4, S–3, shown highlighted below.



Example 2 — scaffolded

A landscaping team must assign three gardeners — Aria, Beth and Cody — to three garden beds. The table gives the time (in hours) each gardener would take on each bed. Allocate the gardeners to the beds so that the **total time** is as small as possible.

Gardener	Bed 1	Bed 2	Bed 3
Aria	9	15	14
Beth	8	11	20
Cody	15	13	16

Working	Comment
Row reduction. Row minimums are Aria 9, Beth 8, Cody 13; subtract each from its row. Aria $0, 6, 5$; Beth $0, 3, 12$; Cody $2, 0, 3$.	Creates a zero in every row.
Column reduction. Column minimums are 0, 0, 3; subtract each from its column. Aria $0, 6, 2$; Beth $0, 3, 9$; Cody $2, 0, 0$.	Creates a zero in every column.
Cover the zeros. A line down Bed 1 and a line across the Cody row cover every zero — just 2 lines.	$2 < 3$, so this is **not** yet optimal.
Adjust. Smallest uncovered entry is 2. Subtract 2 from each uncovered entry; add 2 to the twice-covered entry (Cody, Bed 1). Aria $0, 4, 0$; Beth $0, 1, 7$; Cody $4, 0, 0$.	Only Cody–Bed 1 lies on two lines.

Working	Comment
Cover again. The zeros now need 3 lines, so an allocation is possible.	Lines $= n = 3$.
Allocate. Beth has a single zero (Bed 1) $\Rightarrow$ Beth–Bed 1. Then Aria–Bed 3, Cody–Bed 2.	One zero per row and column.
Total time $= 8 + 14 + 13 = 35$ hours (from the original table).	Beth 8 + Aria 14 + Cody 13.

The optimal allocation is Aria–Bed 3, Beth–Bed 1, Cody–Bed 2, taking 35 hours in total.

Example 3 — unscaffolded

A courier depot assigns four riders — Ada, Ben, Cal and Dan — to four delivery runs (North, East, South, West). The table gives the time in minutes each rider needs for each run. Find the allocation that minimises the total delivery time.

Rider	North	East	South	West
Ada	14	12	27	30
Ben	11	29	22	24
Cal	30	29	30	15
Dan	29	10	26	12

Row-reduce by subtracting the row minimums 12, 11, 15, 10:

	North	East	South	West
Ada	2	0	15	18
Ben	0	18	11	13
Cal	15	14	15	0
Dan	19	0	16	2

Column-reduce by subtracting the column minimums 0, 0, 11, 0:

	North	East	South	West
Ada	2	0	4	18
Ben	0	18	0	13
Cal	15	14	4	0
Dan	19	0	5	2

The zeros can be covered by a line across the Ben row and lines down the East and West columns — 3 lines. Since $3 < 4$ the allocation is not yet possible. The smallest uncovered entry is 2 (Ada, North). Subtracting 2 from every uncovered entry and adding 2 to the twice-covered entries (Ben–East and Ben–West) gives:

	North	East	South	West
Ada	0	0	2	18
Ben	0	20	0	15
Cal	13	14	2	0
Dan	17	0	3	2

Now 4 lines are needed, so an allocation is possible. Cal has a single zero (West) $\Rightarrow$ Cal–West; Dan then has a single zero (East) $\Rightarrow$ Dan–East; Ada then takes North and Ben takes South. The optimal allocation is

$$\text{Ada–North}, \text{Ben–South}, \text{Cal–West}, \text{Dan–East},$$

with total time $14 + 22 + 15 + 10 = 61$ minutes from the original table.

$\therefore$ the minimum total delivery time is 61 minutes.

Example 4 — unscaffolded

A company assigns three sales representatives — Priya, Quinn and Raj — to three regions. The table gives the expected weekly profit (in \$'000) for each representative in each region. Allocate the representatives to the regions to **maximise** the total profit.

Representative	North	Central	South
Priya	10	12	5
Quinn	13	7	4
Raj	15	7	10

This is a **maximisation**, so first convert it to a minimisation. The largest entry is 15; subtract every entry from 15:

	North	Central	South
Priya	5	3	10
Quinn	2	8	11
Raj	0	8	5

Row-reduce (subtract $3, 2, 0$), then column-reduce (subtract $0, 0, 5$):

	North	Central	South
Priya	2	0	2
Quinn	0	6	4
Raj	0	8	0

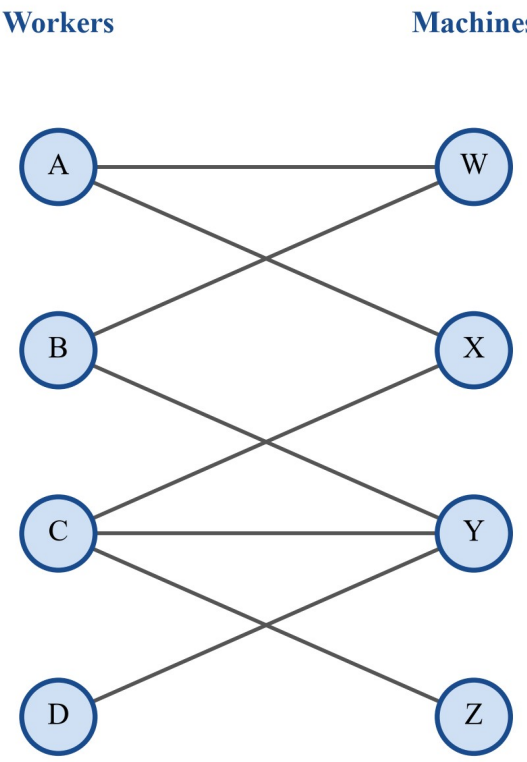
The zeros need 3 lines, so an allocation is possible. Priya has a single zero (Central) $\Rightarrow$ Priya–Central; South has a single zero (Raj) $\Rightarrow$ Raj–South; Quinn then takes North. Reading the profits from the original table, the total is $12 + 13 + 10 = 35$.

$\therefore$ the maximum total profit is \$35,000, from Priya–Central, Quinn–North, Raj–South.

Practice questions

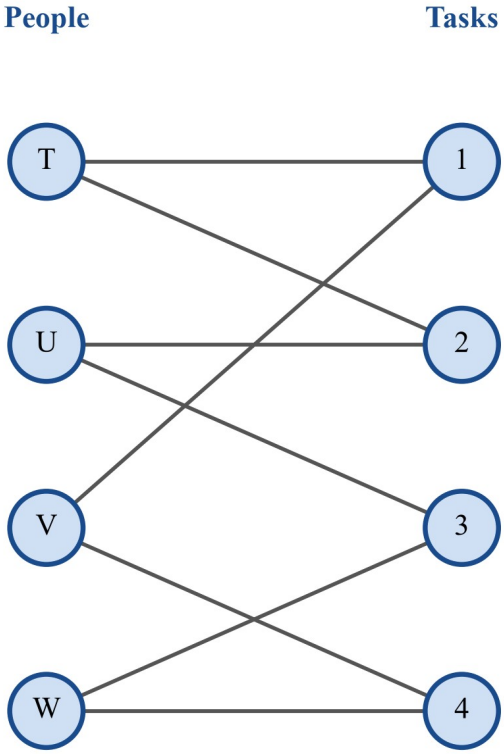
Scaffolded

Q1. Four workers A, B, C, D are each to be assigned to one of four machines W, X, Y, Z. The bipartite graph shows which machines each worker is qualified to operate.



(a) Which machines can worker C operate?
(b) Which worker is qualified to operate only one machine?
(c) Explain why worker D must be assigned to machine Y.
(d) Hence find a complete matching of workers to machines.

Q2. The bipartite graph shows four people T, U, V, W and the tasks 1, 2, 3, 4 each is able to do.



(a) Copy and complete the table, placing a tick where the person can do the task.

Person	1	2	3	4
T				
U				
V				
W				

(b) Find a complete matching of people to tasks.

(c) Find a **second** complete matching different from your answer to (b).

Q3. Three employees are to be assigned to three tasks A, B, C. The table gives the time (minutes) each employee takes on each task. Use the Hungarian algorithm to minimise the total time.

Employee	A	B	C
1	10	17	21
2	20	20	9
3	12	9	15

(a) Carry out the row reduction.

(b) Carry out the column reduction.

(c) Show that the zeros can be covered by 3 lines, so no adjustment is needed.

(d) State the optimal allocation and the minimum total time.

Q4. Three machines are to make three components. The table gives the cost ($) of each machine making each component.

Machine	Comp. 1	Comp. 2	Comp. 3
M1	16	10	22
M2	15	20	14
M3	19	16	22

(a) Carry out the row and column reductions.

(b) Show that the zeros can be covered by only 2 lines, and carry out one adjustment.

(c) State the optimal allocation and the minimum total cost.

Q5. Three painters are each to paint one of three rooms. The table gives the cost ($) for each painter to paint each room. Find the allocation that minimises the total cost, and state that cost.

Painter	Room 1	Room 2	Room 3
1	19	15	12
2	19	20	11
3	17	9	22

Q6. Three workers are to be assigned to three tasks. The table gives the number of units each worker produces on each task. The allocation should **maximise** total output.

Worker	Task 1	Task 2	Task 3
1	8	14	13
2	7	10	14
3	12	15	14

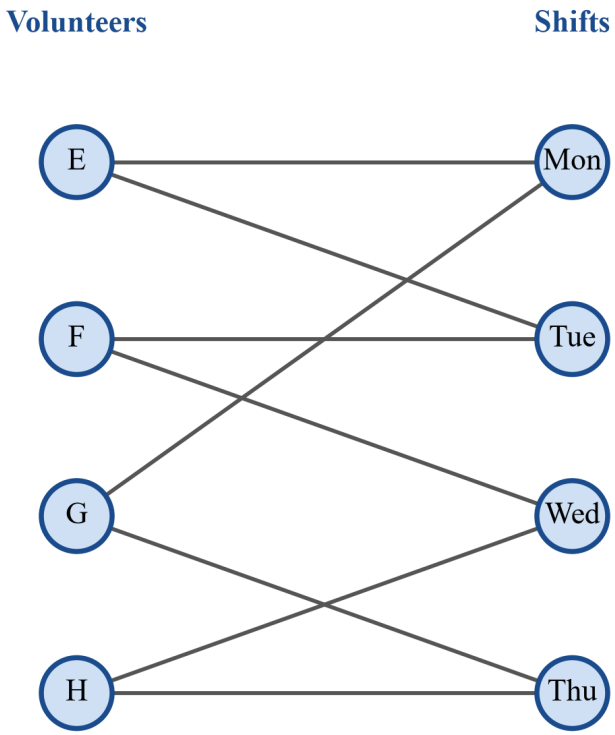
(a) Convert the maximisation to a minimisation by subtracting every entry from the largest entry.

(b) Apply the Hungarian algorithm to the new table.

(c) State the optimal allocation and the maximum total output.

Unscaffolded

Q7. Four volunteers E, F, G, H are to be rostered onto four shifts (Mon, Tue, Wed, Thu), one volunteer per shift. The bipartite graph shows the shifts each volunteer is available for. Find a complete matching (a valid roster).



Q8. Three tutors are to be assigned to three subjects. The table gives the preparation time (hours) each tutor needs for each subject. Find the allocation that minimises the total time.

Tutor	Subject 1	Subject 2	Subject 3
1	17	18	8
2	12	17	11
3	18	22	22

Q9. A garage assigns three mechanics to three services. The table gives the time (minutes) each mechanic takes for each service. Determine the allocation that minimises the total time and state that time.

Mechanic	Service 1	Service 2	Service 3
1	21	9	8
2	15	20	21
3	10	16	17

Q10. Four programmers are to be assigned to four modules. The table gives the number of days each programmer needs for each module. Use the Hungarian algorithm to find the allocation that minimises the total number of days.

Programmer	Mod 1	Mod 2	Mod 3	Mod 4
1	17	19	13	33
2	22	25	14	12
3	12	10	22	27
4	19	35	34	11

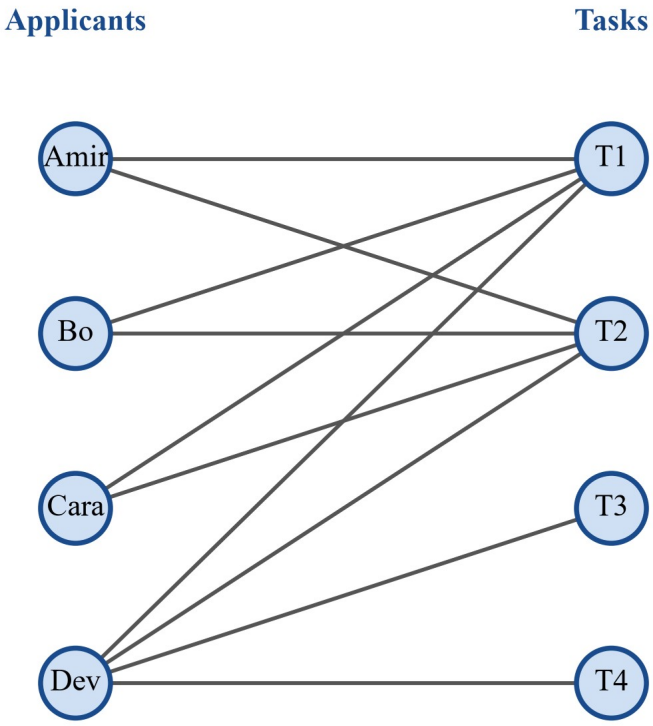
Q11. Three market stalls are to be run by three vendors. The table gives the expected daily profit ($) for each vendor at each stall. Allocate the vendors to **maximise** total profit.

Vendor	Stall 1	Stall 2	Stall 3
1	6	14	5
2	18	12	9
3	13	8	8

Q12. A cleaning contractor assigns four cleaners to four offices. The table gives the time (hours) each cleaner takes for each office. Find the allocation that minimises the total time.

Cleaner	Office 1	Office 2	Office 3	Office 4
1	17	24	23	25
2	11	17	23	24
3	17	30	23	16
4	25	16	11	11

Q13. Four applicants — Amir, Bo, Cara and Dev — are to be matched to four tasks $T1$, $T2$, $T3$, $T4$. The bipartite graph shows the tasks each applicant is qualified for.



(a) Explain why **no** complete matching of the four applicants to the four tasks is possible.

(b) State a single extra qualification that would make a complete matching possible.

Q14. A company assigns four salespeople to four regions. The table gives the expected annual profit ($'000). Allocate the salespeople to the regions to **maximise** total profit, and state that profit.

Salesperson	Region 1	Region 2	Region 3	Region 4
1	26	54	27	22
2	30	44	37	47
3	37	48	31	44
4	21	52	38	33

Q15. A transport firm assigns three trucks to three routes. The table gives the fuel cost ($) for each truck on each route. Find the minimum-cost allocation and state the total fuel cost.

Truck	Route 1	Route 2	Route 3
1	20	15	15
2	18	14	20
3	11	9	15

Q16. In the Hungarian algorithm: (a) Explain why subtracting a constant from every entry of one row does not change which allocation is cheapest. (b) Explain what it means, in the reduced matrix, when a complete allocation using only zeros can be found. (c) A student has row- and column-reduced a 4×4 matrix and can cover all the zeros with 3 lines. State what this tells them, and what they should do next.

Q17. Three athletes are each to compete in one of three events. The table gives the points each athlete is expected to score in each event. Allocate the athletes to the events to **maximise** the total points scored.

Athlete	Event 1	Event 2	Event 3
1	7	8	15
2	7	18	13
3	11	16	5

Q18. A restaurant assigns three chefs to three dishes. The table gives the time (minutes) each chef needs for each dish.

Chef	Dish 1	Dish 2	Dish 3
1	14	11	18
2	22	20	11
3	19	20	14

(a) Find the allocation that minimises the total time, and state that time.

(b) Interpret your answer in the context of the problem.

Q19. A studio assigns three designers to three projects. The table gives the value ($'000) each designer would add to each project. Allocate the designers to **maximise** the total value added, and state that value.

Designer	Project 1	Project 2	Project 3
1	15	17	6
2	7	17	14
3	5	9	17

Q20. A construction project has four stages, each to be carried out by one of four teams. The table gives the cost ($'000) for each team to carry out each stage. Determine the allocation that minimises the total cost, and state that cost.

Team	Stage 1	Stage 2	Stage 3	Stage 4
1	34	25	22	13
2	12	13	29	35
3	21	26	23	23
4	32	34	24	12

Answers

Q1. (a) X, Y, Z; (b) D (only Y); (c) D can operate only Y, so Y is forced; (d) A–X, B–W, C–Z, D–Y. **Q2.** (a) T: 1,2; U: 2,3; V: 1,4; W: 3,4; (b) e.g. T–1, U–2, V–4, W–3; (c) T–2, U–3, V–1, W–4. **Q3.** Employee 1–A, 2–C, 3–B; minimum total time 28 minutes. **Q4.** M1–Comp. 2, M2–Comp. 3, M3–Comp. 1; minimum total cost \$43. **Q5.** Painter 1–Room 1, 2–Room 3, 3–Room 2; minimum total cost \$39. **Q6.** Worker 1–Task 2, 2–Task 3, 3–Task 1; maximum total output 40 units. **Q7.** e.g. E–Mon, F–Tue, G–Thu, H–Wed (also E–Tue, F–Wed, G–Mon, H–Thu). **Q8.** Tutor 1–Subject 3, 2–Subject 1, 3–Subject 2; minimum total time 42 hours. **Q9.** Mechanic 1–Service 3, 2–Service 2, 3–Service 1; minimum total time 38 minutes. **Q10.** Programmer 1–Mod 1, 2–Mod 3, 3–Mod 2, 4–Mod 4; minimum total 52 days. **Q11.** Vendor 1–Stall 2, 2–Stall 1, 3–Stall 3; maximum total profit \$40. **Q12.** Cleaner 1–Office 1, 2–Office 2, 3–Office 4, 4–Office 3; minimum total time 61 hours. **Q13.** (a) Amir, Bo and Cara can do only tasks from $\{T1, T2\}$ — three applicants but only two possible tasks, so at least one cannot be placed; (b) e.g. allow Cara (or Amir or Bo) to do $T3$. **Q14.** Salesperson 1–Region 2, 2–Region 4, 3–Region 1, 4–Region 3; maximum total profit \$176,000. **Q15.** Truck 1–Route 3, 2–Route 2, 3–Route 1; minimum total fuel cost \$40. **Q16.** (a) every allocation takes exactly one entry from that row, so all totals fall by the same constant and the ranking is unchanged; (b) that allocation has total 0 in the reduced matrix, hence minimum cost in the original; (c) the allocation is not yet optimal ($3 < 4$ lines), so adjust the matrix and cover again. **Q17.** Athlete 1–Event 3, 2–Event 2, 3–Event 1; maximum total points 44. **Q18.** (a) Chef 1–Dish 2, 2–Dish 3, 3–Dish 1; minimum total time 41 minutes; (b) see solution. **Q19.** Designer 1–Project 1, 2–Project 2, 3–Project 3; maximum total value \$49,000. **Q20.** Team 1–Stage 3, 2–Stage 2, 3–Stage 1, 4–Stage 4; minimum total cost \$68,000.

Fully-worked solutions

Q1. (a) The edges from C reach X, Y and Z. (b) D is joined only to Y, so D is qualified for only one machine. (c) Because D can operate only Y, machine Y must be given to D — this is a forced choice. (d) With Y taken by D: B is joined to W and Y, so B takes W; A is joined to W and X, and W is gone, so A takes X; finally C takes the remaining machine Z. A complete matching is A–X, B–W, C–Z, D–Y.

Q2. (a) Reading the edges: T to $1, 2$; U to $2, 3$; V to $1, 4$; W to $3, 4$.

Person	1	2	3	4
T	✓	✓		
U		✓	✓	
V	✓			✓
W			✓	✓

(b) Take T–1. Then V (joined to $1, 4$) takes 4; W (joined to $3, 4$) takes 3; U takes 2. This gives T–1, U–2, V–4, W–3. (c) Alternatively take T–2. Then U takes 3, W takes 4, and V takes 1, giving the second complete matching T–2, U–3, V–1, W–4. (These are the only two.)

Q3. Row-reduce (subtract 10, 9, 9):

	A	B	C
1	0	7	11
2	11	11	0
3	3	0	6

The column minimums are already 0, so column reduction changes nothing. Each of columns A, B, C contains a single zero, so the zeros need 3 lines — an allocation is possible with no adjustment. Employee 1–A (0), Employee 2–C (0), Employee 3–B (0). The minimum total time is $10 + 9 + 9 = 28$ minutes.

Q4. Row-reduce (subtract 10, 14, 16) then column-reduce (subtract 1, 0, 0):

	Comp. 1	Comp. 2	Comp. 3
M1	5	0	12
M2	0	6	0
M3	2	0	6

The zeros are covered by a line across the M2 row and a line down Comp. 2 — only 2 lines, so this is not yet optimal. The smallest uncovered entry is 2; subtract it from every uncovered entry and add it to the twice-covered entry (M2, Comp. 2):

	Comp. 1	Comp. 2	Comp. 3
M1	3	0	10
M2	0	8	0
M3	0	0	4

Now 3 lines are needed. Comp. 3 has a single zero (M2) $\Rightarrow$ M2–Comp. 3; M1 then takes Comp. 2; M3 takes Comp. 1. The minimum total cost is $10 + 14 + 19 = 43$, that is \$43.

Q5. Row-reduce (subtract 12, 11, 9) then column-reduce (subtract 7, 0, 0):

	Room 1	Room 2	Room 3
1	0	3	0
2	1	9	0
3	1	0	13

(After the row reduction column Room 1 reads 7, 8, 8, so 7 is subtracted from that column; Room 2 and Room 3 already contain a zero and are left unchanged.) The zeros need 3 lines. Room 2 has a single zero (Painter 3) $\Rightarrow$ Painter 3–Room 2; Room 3's zeros are Painter 1 and Painter 2, and Painter 1 also has a zero in Room 1. Taking Painter 2–Room 3 (its only zero) leaves Painter 1–Room 1. The minimum total cost is $19 + 11 + 9 = 39$, that is \$39.

Q6. (a) The largest entry is 15; subtracting every entry from 15 gives

	Task 1	Task 2	Task 3
1	7	1	2
2	8	5	1
3	3	0	1

(b) Row-reduce (subtract 1, 1, 0) then column-reduce (subtract 3, 0, 0):

	Task 1	Task 2	Task 3
1	3	0	1
2	4	4	0
3	0	0	1

The zeros need 3 lines. (c) Task 3 has a single zero (Worker 2) $\Rightarrow$ Worker 2–Task 3; Worker 1 then takes Task 2 (its only remaining zero); Worker 3 takes Task 1. Reading the original table, the maximum total output is $14 + 14 + 12 = 40$ units.

Q7. Look for forced choices, or build the roster edge by edge. Take E–Mon. Then G (available Mon, Thu) must take Thu; H (available Wed, Thu) then takes Wed; and F takes Tue. A valid roster is E–Mon, F–Tue, G–Thu, H–Wed. (Starting instead with E–Tue gives the only other complete matching, E–Tue, F–Wed, G–Mon, H–Thu.)

Q8. Row-reduce (subtract 8, 11, 18) then column-reduce (subtract 0, 4, 0):

	Subject 1	Subject 2	Subject 3
1	9	6	0

	Subject 1	Subject 2	Subject 3
2	1	2	0
3	0	0	4

The zeros are covered by lines down Subject 3 and across the Tutor 3 row — only 2 lines. The smallest uncovered entry is 1; adjust (subtract from uncovered, add to the twice-covered Tutor 3, Subject 3):

	Subject 1	Subject 2	Subject 3
1	8	5	0
2	0	1	0
3	0	0	5

Now 3 lines are needed. Subject 2 has a single zero (Tutor 3) $\Rightarrow$ Tutor 3–Subject 2; Tutor 1 takes Subject 3; Tutor 2 takes Subject 1. The minimum total time is $8 + 12 + 22 = 42$ hours.

Q9. Row-reduce (subtract 8, 15, 10) then column-reduce (subtract 0, 1, 0):

	Service 1	Service 2	Service 3
1	13	0	0
2	0	4	6
3	0	5	7

The zeros are covered by lines across the Mechanic 1 row and down Service 1 — 2 lines. The smallest uncovered entry is 4; adjusting gives:

	Service 1	Service 2	Service 3
1	17	0	0
2	0	0	2
3	0	1	3

Now 3 lines are needed. Service 3 has a single zero (Mechanic 1) $\Rightarrow$ Mechanic 1–Service 3; Mechanic 3's only zero is Service 1 $\Rightarrow$ Mechanic 3–Service 1; Mechanic 2 takes Service 2. The minimum total time is $8 + 20 + 10 = 38$ minutes.

Q10. Row-reduce (subtract 13, 12, 10, 11) then column-reduce (subtract 2, 0, 0, 0):

	Mod 1	Mod 2	Mod 3	Mod 4
1	2	6	0	20
2	8	13	2	0
3	0	0	12	17
4	6	24	23	0

The zeros can be covered by 3 lines (down Mod 4, across the Programmer 3 row, and down Mod 3), so adjust. The smallest uncovered entry is 2; adjusting gives a matrix still coverable by 3 lines, so adjust a second time (smallest uncovered entry 2 again). The result is:

	Mod 1	Mod 2	Mod 3	Mod 4
1	0	4	0	22
2	4	9	0	0
3	0	0	14	21
4	2	20	21	0

Now 4 lines are needed. Mod 2 has a single zero (Programmer 3) ⇒ Programmer 3–Mod 2; Programmer 4 then has a single zero (Mod 4) ⇒ Programmer 4–Mod 4; Programmer 2 then takes Mod 3 and Programmer 1 takes Mod 1. The minimum total is $17 + 14 + 10 + 11 = 52$ days.

Q11. The largest entry is 18; subtracting every entry from 18, then row-reducing (subtract 4, 0, 5) and column-reducing (subtract 0, 0, 5) gives:

	Stall 1	Stall 2	Stall 3
1	8	0	4
2	0	6	4
3	0	5	0

The zeros need 3 lines. Stall 2 has a single zero (Vendor 1) ⇒ Vendor 1–Stall 2; Stall 3's only zero is Vendor 3 ⇒ Vendor 3–Stall 3; Vendor 2 takes Stall 1. Reading the original profits, the maximum total is $14 + 18 + 8 = 40$, that is \$40.

Q12. Row-reduce (subtract 17, 11, 16, 11) then column-reduce (subtract 0, 5, 0, 0):

	Office 1	Office 2	Office 3	Office 4
1	0	2	6	8
2	0	1	12	13
3	1	9	7	0
4	14	0	0	0

The zeros are covered by 3 lines (down Office 1, down Office 4, across the Cleaner 4 row), so adjust. The smallest uncovered entry is 1; adjusting gives:

	Office 1	Office 2	Office 3	Office 4
1	0	1	5	8
2	0	0	11	13
3	1	8	6	0
4	15	0	0	1

Now 4 lines are needed. Cleaner 1 has a single zero (Office 1) ⇒ Cleaner 1–Office 1; Cleaner 2 then takes Office 2; Cleaner 3 takes Office 4; Cleaner 4 takes Office 3. The minimum total time is $17 + 17 + 16 + 11 = 61$ hours.

Q13. (a) Amir, Bo and Cara are each joined only to $T1$ and $T2$. Between them these three applicants can be matched only to the two tasks $T1$ and $T2$, and two tasks cannot accommodate three different people, so at least one of Amir, Bo, Cara must be left without a task — no complete matching is possible. (b) Giving one of them an extra qualification for a third task, for example allowing Cara to do $T3$, means Amir, Bo and Cara together reach the three tasks $T1$, $T2$, $T3$; then Dev takes $T4$ and a complete matching exists (e.g. Amir–$T1$, Bo–$T2$, Cara–$T3$, Dev–$T4$).

Q14. The largest entry is 54; subtract every entry from 54, then row-reduce (subtract 0, 7, 6, 2) and column-reduce (subtract 11, 0, 10, 0). Covering the zeros needs 3 lines (down Region 1, down Region 2 and across the Salesperson 2 row), so one adjustment (smallest uncovered entry 4) is made, giving:

	Region 1	Region 2	Region 3	Region 4
1	17	0	13	28
2	10	7	0	0
3	0	0	3	0
4	20	0	0	15

Now 4 lines are needed. Region 1 has a single zero (Salesperson 3) ⇒ Salesperson 3–Region 1; Salesperson 1's only zero is Region 2 ⇒ Salesperson 1–Region 2; Salesperson 4 then takes Region 3; Salesperson 2 takes Region 4. Reading the original profits, the maximum total is $54 + 47 + 37 + 38 = 176$, that is \$176,000.

Q15. Row-reduce (subtract 15, 14, 9) then column-reduce (subtract 2, 0, 0):

	Route 1	Route 2	Route 3
1	3	0	0
2	2	0	6
3	0	0	6

The zeros need 3 lines. Route 3 has a single zero (Truck 1) ⇒ Truck 1–Route 3; Route 1's only zero is Truck 3 ⇒ Truck 3–Route 1; Truck 2 takes Route 2. The minimum total fuel cost is $15 + 14 + 11 = 40$, that is \$40.

Q16. (a) Every complete allocation chooses exactly one entry from each row. If a constant c is subtracted from a whole row, then each allocation includes exactly one of those reduced entries, so every allocation total decreases by the same c. The differences between allocation totals are unchanged, so the cheapest allocation is still the cheapest. (b) If a complete allocation can be made using only zeros of the reduced matrix, its total in the reduced matrix is 0 — the least possible, since no entry is negative — so the corresponding allocation is a minimum-cost allocation of the original matrix. (c) Being able to cover all zeros with only 3 lines (fewer than $n = 4$) means an optimal allocation is **not** yet available; the student should adjust the matrix (subtract the smallest uncovered entry from all uncovered entries and add it at the intersections) and then try to cover the zeros again.

Q17. The largest entry is 18; subtract every entry from 18, then row-reduce (subtract 3, 0, 2) and column-reduce (subtract 5, 0, 0):

	Event 1	Event 2	Event 3
1	3	7	0
2	6	0	5
3	0	0	11

The zeros need 3 lines. Event 3 has a single zero (Athlete 1) ⇒ Athlete 1–Event 3; Event 1's only zero is Athlete 3 ⇒ Athlete 3–Event 1; Athlete 2 takes Event 2. Reading the original points, the maximum total is $15 + 18 + 11 = 44$.

Q18. (a) Row-reduce (subtract 11, 11, 14) then column-reduce (subtract 3, 0, 0):

	Dish 1	Dish 2	Dish 3
1	0	0	7
2	8	9	0
3	2	6	0

The zeros — Chef 1 in Dish 1 and Dish 2, and Chef 2 and Chef 3 in Dish 3 — are covered by a line across the Chef 1 row and a line down Dish 3, just 2 lines. The smallest uncovered entry is 2; adjusting gives:

	Dish 1	Dish 2	Dish 3
1	0	0	9
2	6	7	0
3	0	4	0

Now 3 lines are needed. Dish 2 has a single zero (Chef 1) ⇒ Chef 1–Dish 2; Chef 2 then has a single zero (Dish 3) ⇒ Chef 2–Dish 3; Chef 3 takes Dish 1. The minimum total time is $11 + 11 + 19 = 41$ minutes. (b) The kitchen works fastest when Chef 1 prepares Dish 2, Chef 2 prepares Dish 3 and Chef 3 prepares Dish 1; no other assignment of the three chefs to the three dishes finishes in less than 41 minutes of total preparation time.

Q19. The largest entry is 17; subtract every entry from 17. Each row already contains a 17, so every row minimum is 0 and row reduction changes nothing; column-reduce (subtract 2, 0, 0):

	Project 1	Project 2	Project 3
1	0	0	11
2	8	0	3
3	10	8	0

The zeros need 3 lines. Project 3 has a single zero (Designer 3) $\Rightarrow$ Designer 3–Project 3; Project 1's only zero is Designer 1 $\Rightarrow$ Designer 1–Project 1; Designer 2 takes Project 2. Reading the original values, the maximum total is $15 + 17 + 17 = 49$, that is \$49,000.

Q20. Row-reduce (subtract 13, 12, 21, 12) then column-reduce (subtract 0, 1, 2, 0):

	Stage 1	Stage 2	Stage 3	Stage 4
1	21	11	7	0
2	0	0	15	23
3	0	4	0	2
4	20	21	10	0

The zeros can be covered by 3 lines (across the Team 2 and Team 3 rows and down Stage 4), so adjust. The smallest uncovered entry is 7; adjusting gives:

	Stage 1	Stage 2	Stage 3	Stage 4
1	14	4	0	0
2	0	0	15	30
3	0	4	0	9
4	13	14	3	0

Now 4 lines are needed. Stage 2 has a single zero (Team 2) $\Rightarrow$ Team 2–Stage 2; Team 4 then has a single zero (Stage 4) $\Rightarrow$ Team 4–Stage 4; Team 1 then takes Stage 3 and Team 3 takes Stage 1. The minimum total cost is $22 + 13 + 21 + 12 = 68$, that is \$68,000.

Theory

A **project** is a task made up of a number of smaller **activities**. Each activity takes a known **duration** (a time to complete), and some activities cannot begin until certain others have finished. Building a house, staging a concert or assembling a car all work this way: the walls cannot go up before the foundations are set. This section shows how to record these ordering rules in a table and then draw them as a network. The next section uses the network to work out the shortest time in which the whole project can be completed.

Precedence tables.

The ordering rules of a project are recorded in a **precedence table** (also called a **predecessor table**). It lists, for every activity:

- the activity (usually a letter);
- its **duration**; and
- its **immediate predecessors** — the activities that must be **completed immediately before** it can begin.

An **immediate predecessor** of an activity is one that comes *directly* before it. If activity B can only start after A finishes, then A is an immediate predecessor of B. Predecessors carry through a chain, but only the **immediate** ones are listed. For example, if A must finish before B, and B before C, then A must certainly finish before C — but A is *not* an immediate predecessor of C, because B sits between them. The table records only the direct link, B, for C.

An activity with a dash (—) in the predecessor column has no predecessors, so it can begin straight away at the start of the project.

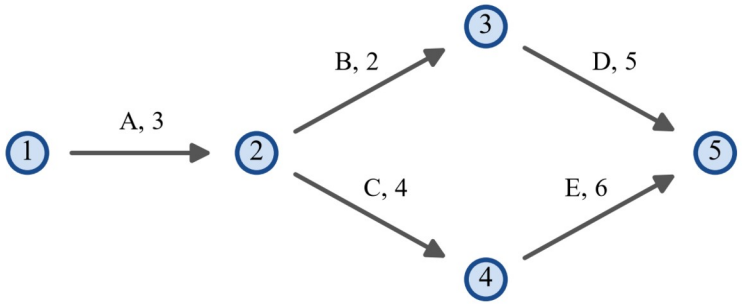
Activity networks.

A precedence table can be drawn as an **activity network** (also called an **arrow diagram**). The convention used in this course is **activity-on-edge**:

- each **activity** is drawn as a **directed edge** (an arrow), labelled with its letter and duration, written "letter, duration";
- each **node** is an **event** — a moment in time marking the *start* or *finish* of one or more activities. Events are numbered;
- the whole project has a single **start node** (where every activity with no predecessor begins) and a single **finish node** (where every activity with no successor ends);
- the arrows point in the direction of time, so an activity's arrow leaves the event at which **all of its immediate predecessors have finished**.

The network below is the activity network for this precedence table.

Activity	Duration	Immediate predecessors
A	3	—
B	2	A
C	4	A
D	5	B
E	6	C



Reading it back: A leaves the start node (node 1); B and C both leave node 2, because each needs only A; D follows B and E follows C, and both finish at node 5, the end of the project.

Constructing a network from a table.

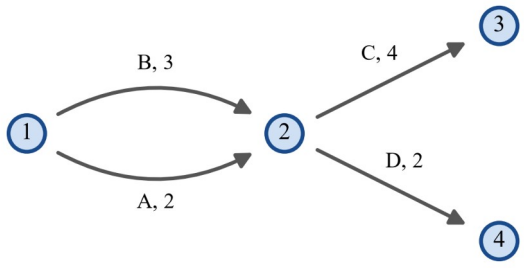
Work through the activities in order, always placing an activity so that it starts at the event where all of its immediate predecessors finish:

1. draw the start node; every activity with no predecessor leaves it;
2. an activity begins at the node where its immediate predecessor(s) end. If two activities share exactly the same predecessors, they may begin at the same node;
3. activities that are nobody's predecessor end at the single finish node.

Dummy activities.

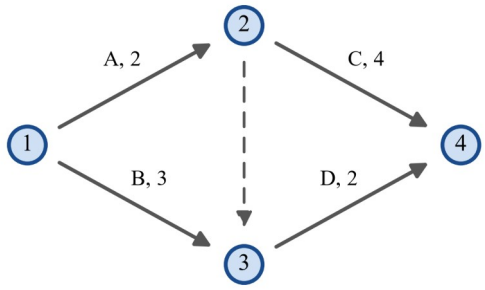
Sometimes the ordering cannot be drawn correctly with ordinary edges alone. A **dummy activity** is used to fix this. A dummy is drawn as a **dashed arrow**, has **zero duration**, and represents no real work — it exists only to show precedence correctly. A dummy is needed in two situations.

To keep the logic correct. Suppose C needs only A, while D needs both A and B. If A and B were made to finish at the *same* node, then C starting there would wrongly appear to depend on B as well.



In the diagram above A and B share an end node, so C *and* D both look as though they need A and B. To repair this, let A and B finish at **different** nodes and run a **dummy** from the end of A into the start of D. Now C leaves A's node (needing only A), while D leaves B's node, which the dummy has also fed with A — so D needs both.

Activity	Duration	Immediate predecessors
A	2	—
B	3	—
C	4	A
D	2	A, B



The dashed arrow from node 2 to node 3 is the dummy. It carries A's completion across to D without forcing A onto C.

To keep activities distinct. If two activities would share **both** their start node and their finish node (they run between the same pair of events), a dummy is added so that each activity has its own unique pair of end nodes.

Reading a network back into a table.

To recover the precedence table from a network, take each activity edge in turn. Its immediate predecessors are the activities that finish at the node where it begins — following any **dummy** arrows backwards, since a dummy passes an activity's completion on to a later node. Its duration is read from the edge label.

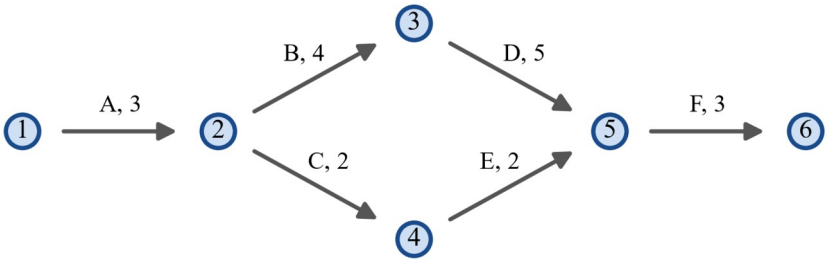
Worked examples

Example 1 — scaffolded

A landscaping job has the precedence table below. Construct its activity network.

Activity	Duration	Immediate predecessors
A	3	—
B	4	A
C	2	A
D	5	B
E	2	C
F	3	D, E

Working	Comment
A has no predecessor, so it leaves the start node (node 1) and ends at node 2.	Every activity with a dash starts the project.
B and C both have only A as predecessor, so both leave node 2.	Activities with identical predecessors share a start node.
D follows B; E follows C. Draw D from the end of B and E from the end of C.	Each begins where its predecessor ends.
F needs both D and E, so D and E are made to finish at the **same** node, and F leaves it.	F starts only once *both* D and E are complete.
F has no successor, so it ends at the finish node. The only activities sharing a predecessor are B and C, whose predecessor sets are *identical*, not partly overlapping; and no two activities run between the same pair of nodes. So **no dummy is needed**.	Both dummy triggers must be ruled out: partly overlapping predecessor sets, and parallel edges.

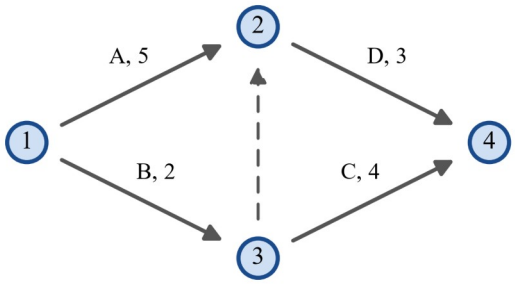


Example 2 — scaffolded

Construct the activity network for the precedence table below.

Activity	Duration	Immediate predecessors
A	5	—
B	2	—
C	4	B
D	3	A, B

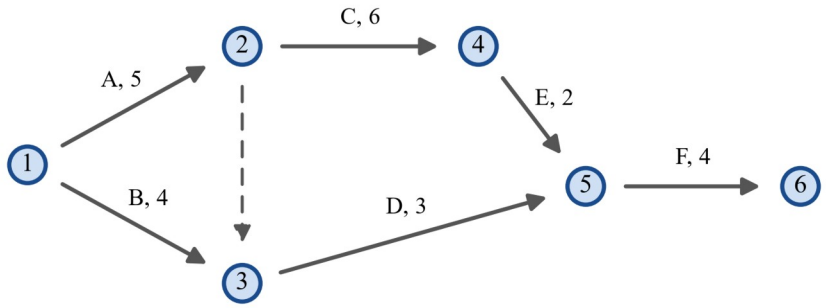
Working	Comment
A and B have no predecessors, so both leave the start node (node 1). Let A end at node 2 and B end at node 3.	Give them separate end nodes to keep the logic flexible.
C needs only B, so C leaves node 3.	C starts where B finishes.
D needs both A and B. It must start at a node reached by *both*.	D cannot start until A and B are done.
Run a **dummy** (dashed) from node 3 to node 2, carrying B's completion to node 2. Now node 2 has both A and B behind it, so D leaves node 2.	Without the dummy, C (at node 3) would be forced to wait for A as well.
C and D both finish the project at node 4.	They are nobody's predecessor, so they reach the finish node.



The dummy lets D depend on both A and B while C still depends on B alone.

Example 3 — unscaffolded

The activity network below models a project. The dashed arrow is a dummy activity. Write out its precedence table.



Take each activity edge and read off its immediate predecessors — the activities finishing at the node where it starts, following the dummy backwards.

A and B both leave the start node (node 1), so neither has a predecessor. C leaves node 2, where A finishes, so C needs A. The dummy runs from node 2 to node 3, carrying A's completion to node 3; B also finishes there, so D, which leaves node 3, needs both A and B. E leaves node 4, where C finishes, so E needs C. Finally F leaves node 5, where both D and E finish, so F needs D and E. Durations are read from the labels.

Activity	Duration	Immediate predecessors
A	5	—
B	4	—
C	6	A
D	3	A, B
E	2	C
F	4	D, E

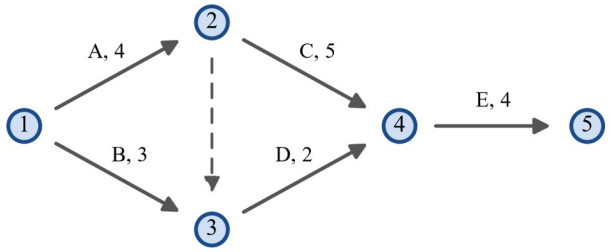
$\therefore$ the precedence table is as shown above, with the dummy recording that A is also an immediate predecessor of D.

Example 4 — unscaffolded

Construct the activity network for the precedence table below, including any dummy activities.

Activity	Duration	Immediate predecessors
A	4	—
B	3	—
C	5	A
D	2	A, B
E	4	C, D

Activities A and B have no predecessors, so both leave the start node; let A end at node 2 and B end at node 3. Activity C needs only A, so it leaves node 2. Activity D needs both A and B: a dummy is run from node 2 to node 3 so that node 3 is reached by both A (through the dummy) and B, and D leaves node 3. Because C needs only A it is unaffected by the dummy. Activity E needs both C and D, so C and D are made to finish at the same node (node 4), and E leaves it to reach the finish node.



$\therefore$ the network is as shown, with one dummy activity carrying A's completion across to D.

Practice questions

Scaffolded

Q1. A project has the precedence table below.

Activity	Duration	Immediate predecessors
A	2	—
B	5	—
C	3	A
D	4	A, B
E	2	C

(a) Which activities can start immediately?

(b) Which activities have A as an immediate predecessor?

(c) List the immediate predecessors of D.

(d) Which activity must be complete before E can begin?

Q2. A project has the precedence table below.

Activity	Duration	Immediate predecessors
A	3	—
B	2	A
C	4	A
D	5	B
E	2	C

(a) Which activity has no predecessor?

(b) After A finishes, which two activities can begin?

(c) State the immediate predecessor of D and of E.

(d) Draw the activity network.

Q3. A project has the precedence table below.

Activity	Duration	Immediate predecessors
A	4	—
B	2	—
C	3	A, B
D	5	B

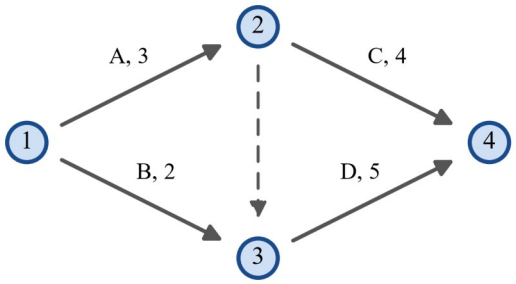
(a) List the immediate predecessors of C.

(b) State the immediate predecessor of D.

(c) Activity B must be completed before both C and D, but C also needs A while D does not. Explain why a dummy activity is needed.

(d) Draw the activity network, including the dummy.

Q4. The activity network below models a project; the dashed arrow is a dummy activity.



(a) Which activities leave the start node, and what does this tell you about their predecessors?

(b) What does the dashed arrow tell you about activity D?

(c) Copy and complete the precedence table.

Activity	Duration	Immediate predecessors
A		
B		
C		
D		

Q5. A project has the precedence table below.

Activity	Duration	Immediate predecessors
A	3	—
B	4	A
C	2	B
D	5	C

(a) State the immediate predecessor of D.

(b) Is A a predecessor of D? Is it an *immediate* predecessor? Explain.

(c) Draw the activity network.

Q6. (a) State the duration of a dummy activity. (b) How is a dummy activity shown on an activity network? (c) Give the two situations in which a dummy activity is needed. (d) For the table below, explain whether a dummy activity is required.

Activity	Duration	Immediate predecessors
A	2	—
B	3	—
C	4	A
D	5	A, B

Unscaffolded

Q7. Construct the activity network for the precedence table below.

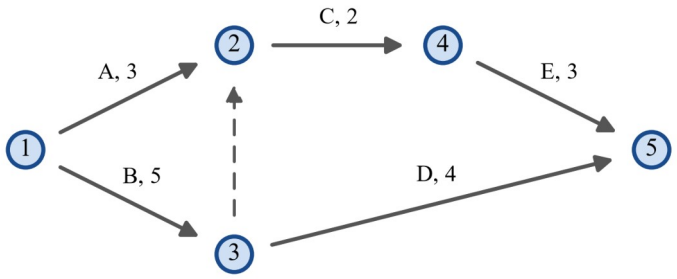
Activity	Duration	Immediate predecessors
A	3	—

Activity	Duration	Immediate predecessors
B	2	A
C	4	A
D	5	B
E	3	C, D

Q8. Construct the activity network for the precedence table below, including any dummy.

Activity	Duration	Immediate predecessors
A	5	—
B	3	—
C	2	A
D	4	A, B

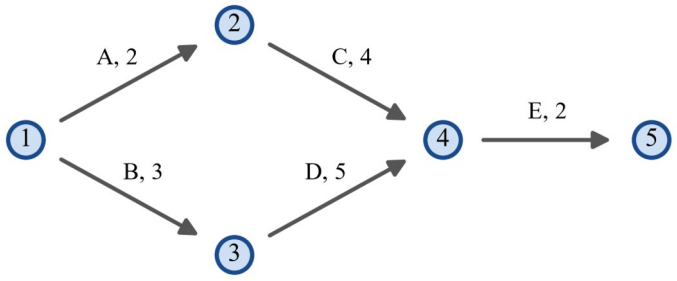
Q9. The activity network below models a project; the dashed arrow is a dummy activity. Write out its precedence table.



Q10. A car service centre plans a job with the precedence table below. Construct its activity network.

Activity	Duration	Immediate predecessors
A	2	—
B	3	A
C	4	A
D	2	B
E	5	C
F	3	D, E

Q11. The activity network below models a project. Write out its precedence table.

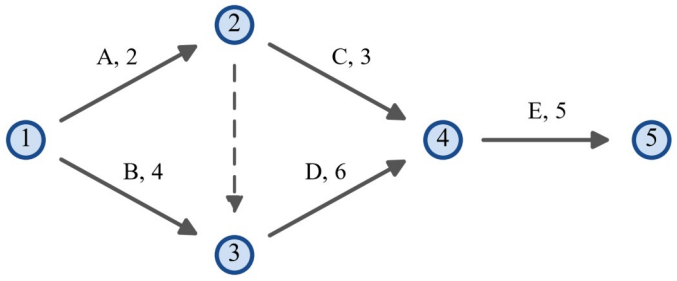


Q12. Construct the activity network for the precedence table below, including any dummy.

Activity	Duration	Immediate predecessors
A	4	—

Activity	Duration	Immediate predecessors
B	3	—
C	2	A
D	5	A, B
E	3	D

Q13. The activity network below models a project; the dashed arrow is a dummy activity. Write out its precedence table.



Q14. (a) In an activity-on-edge network, what does a directed edge represent, and what does a node represent? (b) Why must an activity network have a single start node and a single finish node? (c) A student says a dummy activity makes the project take longer because it is an extra arrow. Explain why this is incorrect.

Q15. Construct the activity network for the precedence table below.

Activity	Duration	Immediate predecessors
A	3	—
B	2	A
C	4	A
D	5	B
E	2	C
F	3	D
G	4	E, F

Q16. A project has the precedence table below.

Activity	Duration	Immediate predecessors
A	2	—
B	3	—
C	4	A
D	5	A, B
E	2	C, D
F	3	D

(a) Which activities can start immediately?

(b) Which activities are terminal (have no successor)?

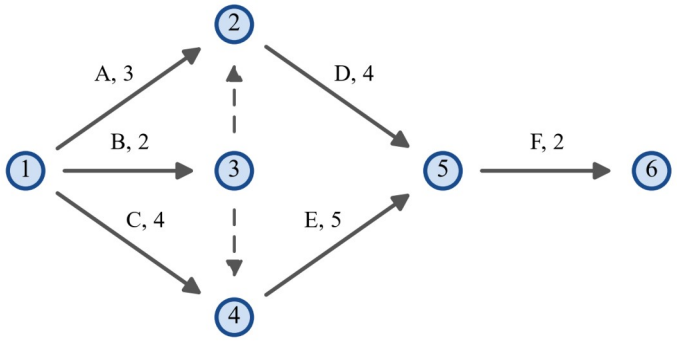
(c) State which activities have D as an immediate predecessor.

Q17. Construct the activity network for the precedence table below, including any dummies.

Activity	Duration	Immediate predecessors
A	2	—

Activity	Duration	Immediate predecessors
B	3	—
C	4	—
D	5	A, B
E	2	B, C

Q18. The activity network below models a project; the dashed arrows are dummy activities. Write out its precedence table.



Q19. Construct the activity network for the precedence table below, including any dummy.

Activity	Duration	Immediate predecessors
A	4	—
B	2	—
C	5	A
D	3	A, B
E	2	C, D
F	3	E

Q20. An event is organised using the precedence table below.

Activity	Duration	Immediate predecessors
A	3	—
B	4	—
C	2	A
D	5	A, B
E	3	B

(a) Draw the activity network, including any dummy activities.

(b) State how many dummy activities are needed.

(c) Which activities can start immediately, and which are terminal?

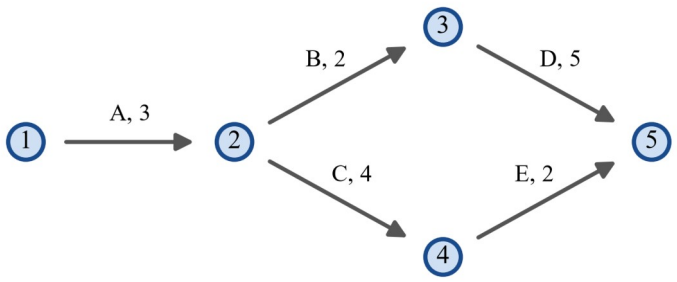
Answers

Q1. (a) A, B; (b) C, D; (c) A and B; (d) C. **Q2.** (a) A; (b) B and C; (c) D needs B, E needs C; (d) see network (no dummy). **Q3.** (a) A and B; (b) B; (c) B feeds both C and D, but C also needs A and D does not, so a dummy separates them; (d) see network (one dummy). **Q4.** (a) A and B — they have no predecessors; (b) A is also an immediate predecessor of D; (c) A: 3, —; B: 2, —; C: 4, A; D: 5, A, B. **Q5.** (a) C; (b) yes, A is a predecessor of D but not an immediate one (B and C lie between); (c) see network (a chain). **Q6.** (a) 0; (b) a dashed arrow; (c) to keep the precedence logic correct, and to give two activities that share the same start and finish nodes distinct end nodes; (d) yes — A precedes both C and D while B precedes only D, so a dummy is required. **Q7.** See network (no dummy); C and D both finish at the node where E begins. **Q8.** See network (one dummy from the end of A to the start of D). **Q9.** A: 3, —; B: 5, —; C: 2, A, B; D: 4, B; E: 3, C. **Q10.** See network (no dummy). **Q11.** A: 2, —; B: 3, —; C: 4, A; D: 5, B; E: 2, C, D. **Q12.** See network (one dummy from the end of A to the start of D). **Q13.** A: 2, —; B: 4, —; C: 3, A; D: 6, A, B; E: 5, C, D. **Q14.** (a) an edge is an activity, a node is an event (the start/finish of activities); (b) so the project has one clear beginning and one clear end; (c) a dummy has zero duration and represents no work, so it adds no time. **Q15.** See network (no dummy); E and F both finish at the node where G begins. **Q16.** (a) A, B; (b) E, F; (c) E and F. **Q17.** See network (two dummies, both from the end of B). **Q18.** A: 3, —; B: 2, —; C: 4, —; D: 4, A, B; E: 5, B, C; F: 2, D, E. **Q19.** See network (one dummy from the end of A to the start of D). **Q20.** (a) see network; (b) 2; (c) A and B can start immediately; C, D and E are terminal.

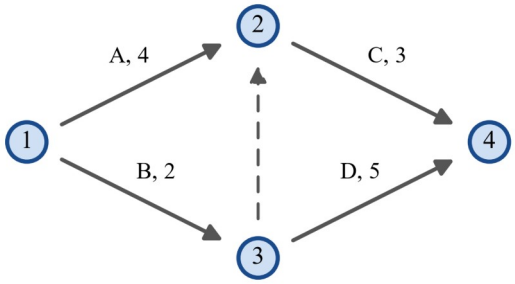
Fully-worked solutions

Q1. Reading the table: (a) A and B have a dash, so they can start immediately. (b) The activities listing A as a predecessor are C (predecessor A) and D (predecessors A, B). (c) The immediate predecessors of D are A and B. (d) E lists C, so C must be complete before E begins.

Q2. (a) Only A has a dash, so A has no predecessor. (b) B and C each list only A, so both can begin once A finishes. (c) D lists B and E lists C. (d) A leaves the start node; B and C both leave the end of A; D follows B and E follows C, finishing together at the end node. No dummy is needed: the only activities sharing a predecessor are B and C, whose predecessor sets are identical ($\{A\}$) rather than partly overlapping, and no two activities run between the same pair of nodes.



Q3. (a) C lists A and B. (b) D lists B. (c) Both C and D need B, so B's completion must reach both. But C also needs A while D needs only B, so A must not be allowed to reach D. Letting A and B finish at different nodes and running a dummy from B's node to C's start node carries B across to C without forcing A onto D. (d) The network has A ending at node 2 and B at node 3; a dummy runs from node 3 to node 2, so C (leaving node 2) has both A and B behind it, while D (leaving node 3) has only B.



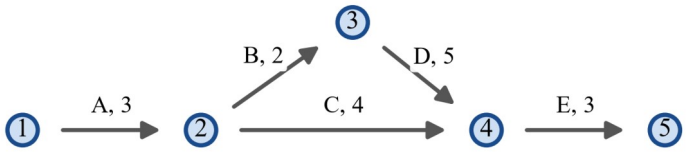
Q4. (a) A and B leave the start node, so they have no predecessors. (b) The dashed arrow is a dummy running from the end of A into the start of D, so A is an immediate predecessor of D as well as B. (c) Completing the table:

Activity	Duration	Immediate predecessors
A	3	—
B	2	—
C	4	A
D	5	A, B

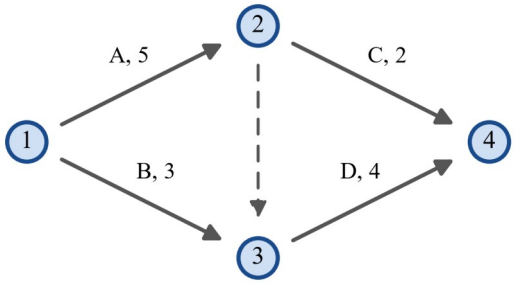
Q5. (a) D lists C, so C is its immediate predecessor. (b) Following the chain $A \to B \to C \to D$, activity A must finish before D, so A *is* a predecessor of D; but it is not an *immediate* predecessor, because B and C lie between them and only the direct link (C) is listed. (c) The network is a single chain of four edges through five events.

Q6. (a) A dummy activity has duration 0. (b) It is drawn as a dashed arrow. (c) A dummy is needed either to keep the precedence logic correct (when activities share some but not all of their predecessors), or to give two activities that would otherwise share the same start and finish nodes their own distinct end nodes. (d) A is an immediate predecessor of both C and D, while B is an immediate predecessor of only D. The predecessor sets $\{A\}$ and $\{A, B\}$ overlap but are not equal, so a dummy is required to carry A to C and D correctly.

Q7. A leaves the start node. B and C both leave the end of A. D follows B. Since E needs both C and D, activities C and D are made to finish at the same node, and E leaves it to reach the finish node. No dummy is needed: the only activities sharing a predecessor are B and C, whose predecessor sets are identical ($\{A\}$) rather than partly overlapping, and no two activities run between the same pair of nodes.



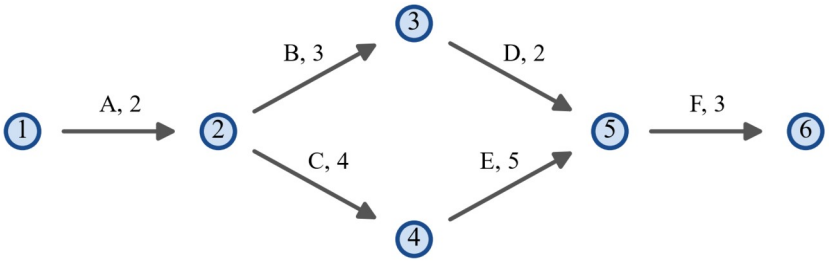
Q8. A and B leave the start node, ending at nodes 2 and 3. C needs only A, so it leaves node 2. D needs both A and B: a dummy runs from node 2 (end of A) to node 3, so node 3 has both A and B behind it, and D leaves node 3. C and D finish the project together.



Q9. Reading each edge: A and B leave the start node (node 1), so both have no predecessor. The dummy runs from node 3 (end of B) to node 2, so node 2 has both A and B behind it; C leaves node 2 and therefore needs A and B. D leaves node 3 and needs only B. E leaves node 4, where C finishes, so E needs C.

Activity	Duration	Immediate predecessors
A	3	—
B	5	—
C	2	A, B
D	4	B
E	3	C

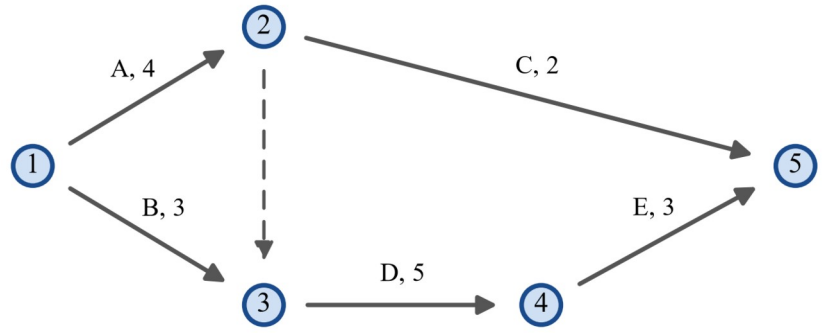
Q10. A leaves the start node. B and C both leave the end of A. D follows B and E follows C. Activity F needs both D and E, so D and E finish at the same node, and F leaves it to reach the finish node. No dummy is needed: only B and C share a predecessor and their predecessor sets are identical, and no two activities run between the same pair of nodes.



Q11. A and B leave the start node, so neither has a predecessor. C leaves the end of A (needs A) and D leaves the end of B (needs B). C and D finish at the same node, from which E leaves, so E needs both C and D.

Activity	Duration	Immediate predecessors
A	2	—
B	3	—
C	4	A
D	5	B
E	2	C, D

Q12. A and B leave the start node, ending at nodes 2 and 3. C needs only A, so it leaves node 2. D needs both A and B: a dummy runs from node 2 to node 3, so D leaves node 3 with both A and B behind it, and E follows D. Neither C nor E is listed as a predecessor, so both are terminal and both end at the single finish node (node 5).

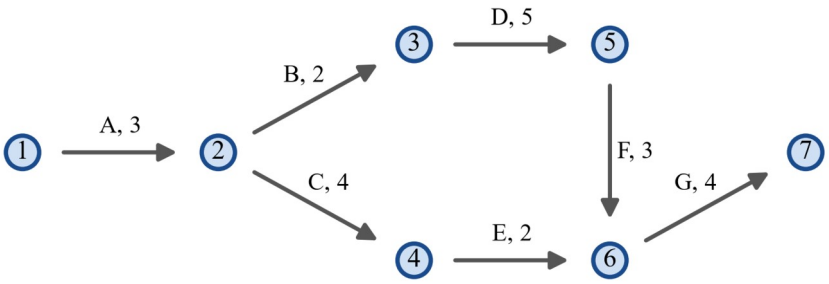


Q13. A and B leave the start node. C leaves the end of A (needs A). The dummy runs from node 2 (end of A) to node 3, so node 3 has both A and B behind it, and D (leaving node 3) needs A and B. C and D finish at the same node, from which E leaves, so E needs both C and D.

Activity	Duration	Immediate predecessors
A	2	—
B	4	—
C	3	A
D	6	A, B
E	5	C, D

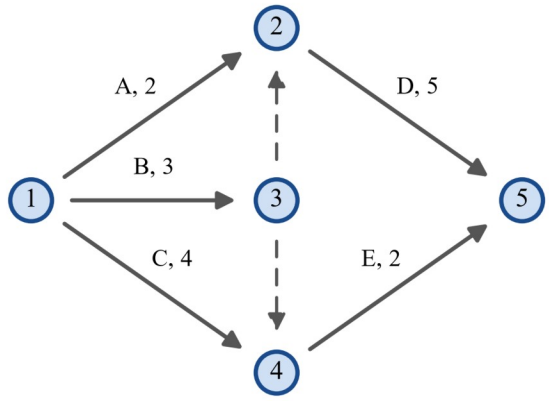
Q14. (a) In an activity-on-edge network a directed edge represents an **activity** (with its duration), and a node represents an **event** — the moment marking the start or finish of one or more activities. (b) A single start node gives the project one clear beginning (where all predecessor-free activities start) and a single finish node gives it one clear end (where all activities with no successor finish), so the whole project has a well-defined start and completion. (c) A dummy activity has zero duration and represents no real work; it only records a precedence relationship, so it adds no time to the project.

Q15. A leaves the start node; B and C both leave the end of A. D follows B and E follows C; F follows D. Activity G needs both E and F, so E and F finish at the same node, from which G leaves to reach the finish node. No dummy is needed: only B and C share a predecessor and their predecessor sets are identical, and no two activities run between the same pair of nodes.



Q16. (a) A and B have a dash, so they can start immediately. (b) A terminal activity is nobody's predecessor. Scanning the predecessor column, E appears nowhere and F appears nowhere, so E and F are terminal (while A, B, C, D each appear as some activity's predecessor). (c) The activities listing D as a predecessor are E (needs C, D) and F (needs D).

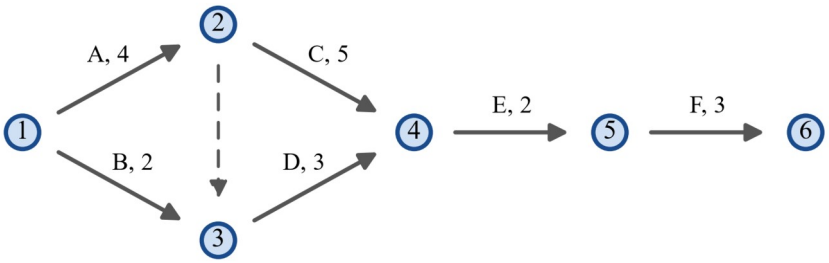
Q17. A, B and C all leave the start node, ending at nodes 2, 3 and 4. D needs A and B; E needs B and C. Since B must feed both D and E, a dummy runs from node 3 (end of B) to node 2, so node 2 has A and B behind it and D leaves it; a second dummy runs from node 3 to node 4, so node 4 has B and C behind it and E leaves it. Two dummies are needed because B is shared between two activities with different predecessor sets.



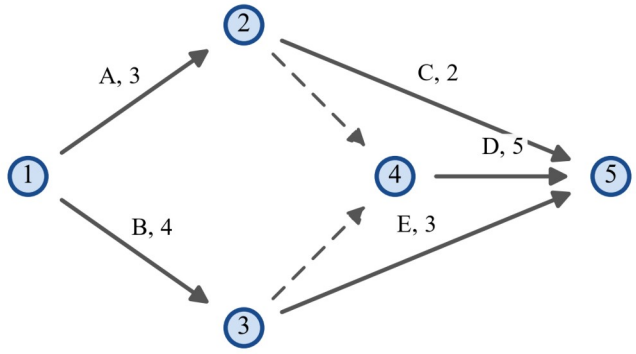
Q18. A, B and C leave the start node, so none has a predecessor. One dummy runs from node 3 (end of B) to node 2 and another from node 3 to node 4, so node 2 has A and B behind it and node 4 has B and C behind it. Thus D (leaving node 2) needs A, B and E (leaving node 4) needs B, C. D and E finish at the same node, from which F leaves, so F needs D and E.

Activity	Duration	Immediate predecessors
A	3	—
B	2	—
C	4	—
D	4	A, B
E	5	B, C
F	2	D, E

Q19. A and B leave the start node, ending at nodes 2 and 3. C needs only A, so it leaves node 2. D needs A and B: a dummy runs from node 2 to node 3, so D leaves node 3 with both behind it. Activity E needs both C and D, so C and D finish at the same node, from which E leaves; F then follows E to the finish node.



Q20. (a) A and B leave the start node, ending at nodes 2 and 3. Activity C needs only A, so it leaves node 2; activity E needs only B, so it leaves node 3. Activity D needs both A and B: a dummy runs from node 2 to a new node 4 and a second dummy from node 3 to node 4, so node 4 has both A and B behind it and D leaves it. Activities C, D and E all finish at the end node.



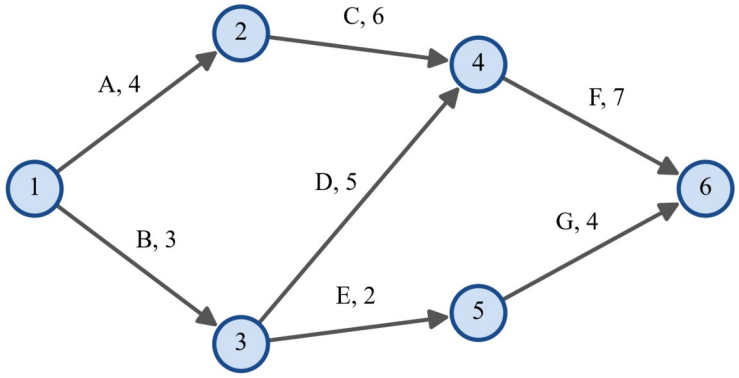
(b) Two dummy activities are needed, because A and B must each feed D while C needs only A and E needs only B. (c) A and B can start immediately; C, D and E are terminal (none is listed as a predecessor).

Theory

A large project is made up of many separate **activities**, some of which cannot begin until others are finished. In an earlier section these precedence relationships were drawn as an **activity network**: each activity is a directed arc labelled with its name and its **duration**, each numbered circle is an **event** (a moment when a set of activities is complete), and the arrows show the order in which the work must be done. This section takes a completed activity network and answers the two scheduling questions that matter most:

- What is the **shortest time** in which the whole project can be finished?
- Which activities are **critical** — must run exactly on time or the whole project is delayed — and which have some **slack**?

The network below models a small landscaping project. Every arc is an activity with a duration in days; we will analyse it as the running example.



The forward scan — earliest start and finish times.

Working **left to right** through the network, the **earliest start time** (EST) of an activity is the earliest moment all of its immediate predecessors are complete, and its **earliest finish time** (EFT) is

$$\text{EFT} = \text{EST} + \text{duration}.$$

An activity with no predecessors has $\text{EST} = 0$. Where several activities lead into the same event, the next activity cannot start until the **last** of them has finished, so

$$\text{EST} = \text{largest EFT of the immediate predecessors}.$$

For the landscaping project:

- A: no predecessors, $\text{EST} = 0$, $\text{EFT} = 0 + 4 = 4$.
- B: no predecessors, $\text{EST} = 0$, $\text{EFT} = 0 + 3 = 3$.
- C: after A, $\text{EST} = 4$, $\text{EFT} = 4 + 6 = 10$.
- D: after B, $\text{EST} = 3$, $\text{EFT} = 3 + 5 = 8$.
- E: after B, $\text{EST} = 3$, $\text{EFT} = 3 + 2 = 5$.
- F: after C **and** D, $\text{EST} = max(10, 8) = 10$, $\text{EFT} = 10 + 7 = 17$.
- G: after E, $\text{EST} = 5$, $\text{EFT} = 5 + 4 = 9$.

The **minimum completion time** of the project is the largest earliest finish time of all — here 17 days.

The backward scan — latest start and finish times.

Working **right to left**, the **latest finish time** (LFT) of an activity is the latest moment it can finish without pushing out the whole project, and its **latest start time** (LST) is

$$\text{LST} = \text{LFT} - \text{duration}.$$

The scan starts by setting the LFT of the final activities equal to the minimum completion time. Where an activity is followed by several others, it must finish in time for the **earliest** of them to still start on time, so

$$\text{LFT} = \text{smallest LST of the immediate successors}.$$

Starting from a completion time of 17:

- F: last activity, $\text{LFT} = 17$, $\text{LST} = 17 - 7 = 10$.
- G: last activity, $\text{LFT} = 17$, $\text{LST} = 17 - 4 = 13$.
- C: before F, $\text{LFT} = 10$, $\text{LST} = 10 - 6 = 4$.
- D: before F, $\text{LFT} = 10$, $\text{LST} = 10 - 5 = 5$.
- E: before G, $\text{LFT} = 13$, $\text{LST} = 13 - 2 = 11$.
- A: before C, $\text{LFT} = 4$, $\text{LST} = 4 - 4 = 0$.
- B: before D and E, $\text{LFT} = min(5, 11) = 5$, $\text{LST} = 5 - 3 = 2$.

Float (slack).

The **float** of an activity is how long it could be delayed, or how much longer it could take, without delaying the whole project:

$$\text{float} = \text{LST} - \text{EST} = \text{LFT} - \text{EFT}.$$

The two forms are equal because each finish time is exactly the duration ahead of its own start time — $\text{EFT} = \text{EST} + \text{duration}$ and $\text{LFT} = \text{LST} + \text{duration}$ — so subtracting the earliest pair from the latest pair gives $\text{LFT} - \text{EFT} = \text{LST} - \text{EST}$. An activity with a large float has plenty of slack; an activity with **zero float** has none at all.

The critical path.

The **critical path** is the chain of activities running from the start of the project to the end that has **zero float**. Equivalently, it is the **longest path** through the network, and its length equals the minimum completion time. Every activity on it must start and finish at its one possible time — delaying any critical activity by one day delays the entire project by one day. Non-critical activities are exactly those with a positive float.

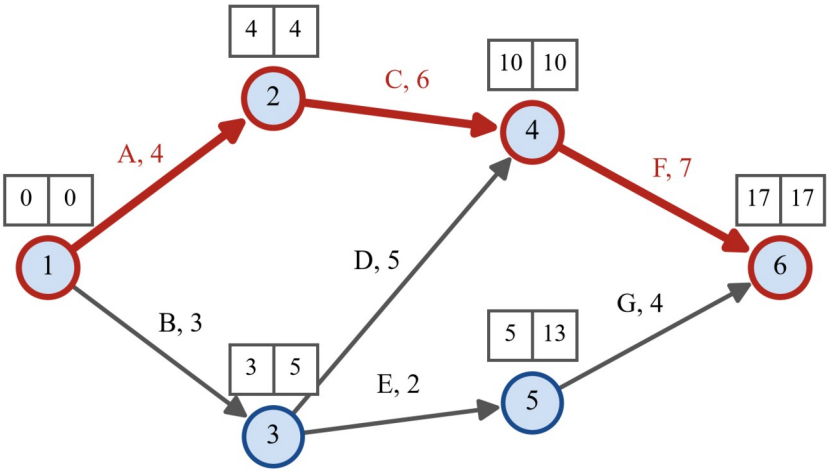
For the landscaping project the floats are

Activity	Duration	EST	LFT	Float
A	4	0	4	0
B	3	0	5	2
C	6	4	10	0
D	5	3	10	2
E	2	3	13	8
F	7	10	17	0
G	4	5	17	8

so the critical activities are A, C and F. The critical path is $A \to C \to F$, of length $4 + 6 + 7 = 17$ days, matching the minimum completion time.

It is often convenient to record the two scan results **at the events** rather than at the activities: the forward scan fills each node with its **earliest event time** (the earliest all activities into it are done) and the backward scan fills its **latest**

event time. In the diagram below each event carries a box showing earliest | latest. An event is on the critical path when these two numbers are equal, and the critical path (in red) links the events where there is no slack. (An activity network may also contain a **dummy activity** of duration 0, drawn dashed; it is scanned like any other activity and simply carries a precedence forward.)



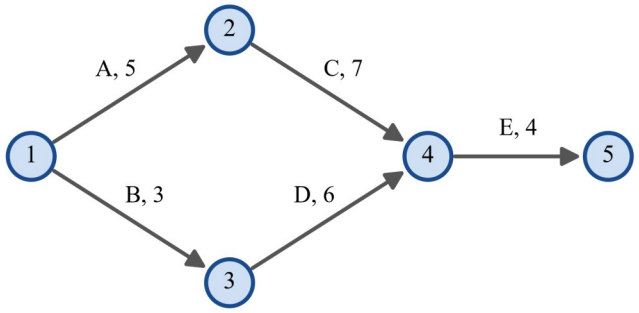
To summarise the method:

- **forward scan** (left to right): EST = largest EFT of predecessors, $\text{EFT} = \text{EST} + \text{duration}$; the project's minimum completion time is the largest EFT;
- **backward scan** (right to left): LFT = smallest LST of successors, $\text{LST} = \text{LFT} - \text{duration}$;
- **float** $= \text{LST} - \text{EST}$; the **critical path** is the zero-float chain from start to finish, and its length is the minimum completion time.

Worked examples

Example 1 — scaffolded

A team building a small website must complete five activities. The activity network is shown below (durations in days); C follows A, D follows B, and E follows both C and D. Find the earliest and latest times, the float of each activity, the critical path and the minimum completion time.



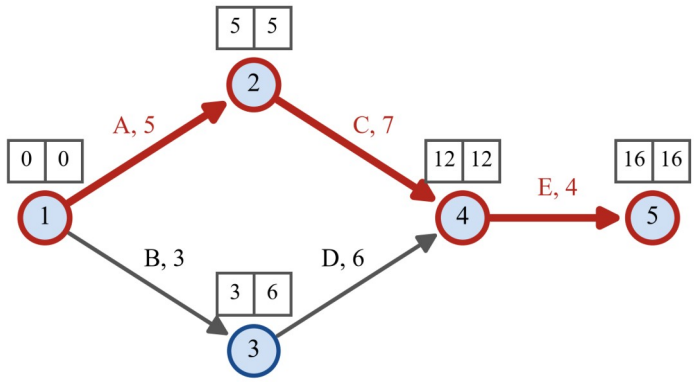
Working	Comment
A: $\text{EST} = 0$, $\text{EFT} = 0 + 5 = 5$. B: $\text{EST} = 0$, $\text{EFT} = 0 + 3 = 3$.	Forward scan. Activities with no predecessor start at 0.
C: $\text{EST} = 5$, $\text{EFT} = 5 + 7 = 12$. D: $\text{EST} = 3$, $\text{EFT} = 3 + 6 = 9$.	C follows A; D follows B.
E: $\text{EST} = max(12, 9) = 12$, $\text{EFT} = 12 + 4 = 16$.	E waits for the later of C and D.

Working	Comment
Minimum completion time $= 16$ days.	Largest EFT in the network.
E: LFT $= 16$, LST $= 16 - 4 = 12$.	Backward scan begins at 16.
C: LFT $= 12$, LST $= 12 - 7 = 5$. D: LFT $= 12$, LST $= 12 - 6 = 6$.	Both feed into E, so both have LFT $=$ LST$(E) = 12$.
A: LFT $= 5$, LST $= 0$. B: LFT $= 6$, LST $= 3$.	A before C; B before D.
Floats: A 0, B 3, C 0, D 3, E 0.	float $=$ LST $-$ EST.

The schedule is

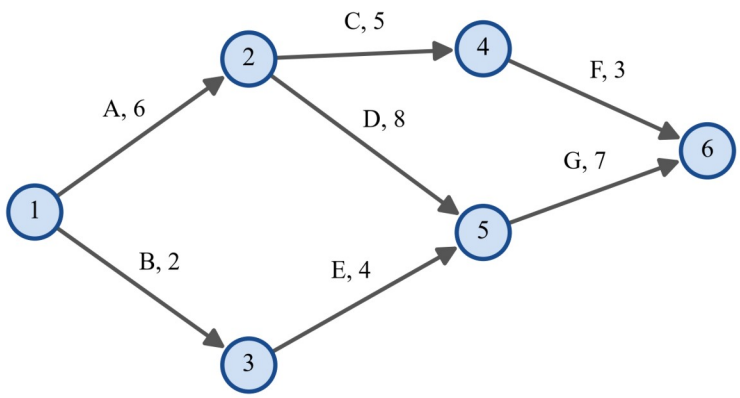
Activity	Duration	EST	LFT	Float
A	5	0	5	0
B	3	0	6	3
C	7	5	12	0
D	6	3	12	3
E	4	12	16	0

The zero-float activities are A, C and E, so the critical path is $A \to C \to E$ (length $5 + 7 + 4 = 16$ days) and the minimum completion time is 16 days.



Example 2 — scaffolded

A caterer plans a wedding using the activity network below (durations in days). Carry out the forward and backward scans, tabulate the schedule, and state the critical path and minimum completion time.

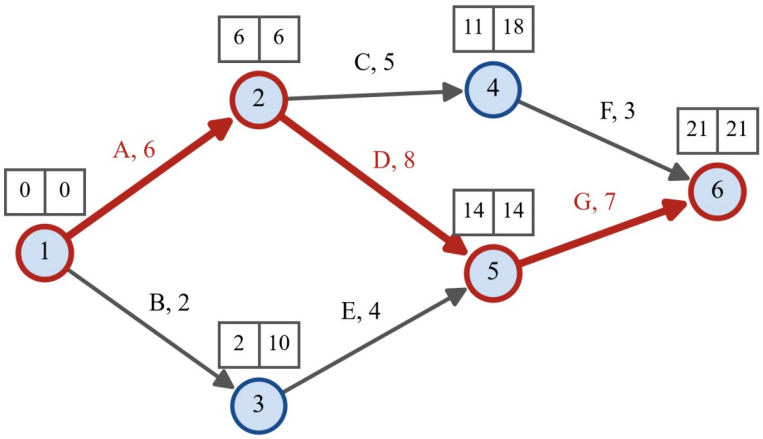


Working	Comment
A: $0 \to 6$. B: $0 \to 2$.	Forward scan: EST $\to$ EFT for the starting activities.
C: EST $= 6$, EFT $= 11$. D: EST $= 6$, EFT $= 14$.	Both follow A.
E: EST $= 2$, EFT $= 6$.	E follows B.
F: EST $= 11$, EFT $= 14$.	F follows C.
G: EST $= max(14, 6) = 14$, EFT $= 21$.	G follows D and E; the later is D.
Minimum completion time $= 21$ days.	Largest EFT.
G: LFT $= 21$, LST $= 14$. F: LFT $= 21$, LST $= 18$.	Backward scan from 21; F and G are last.
C: LFT $= 18$, LST $= 13$. D: LFT $= 14$, LST $= 6$. E: LFT $= 14$, LST $= 10$.	C before F; D and E before G.
A: LFT $= min(13, 6) = 6$, LST $= 0$. B: LFT $= 10$, LST $= 8$.	A feeds C and D; take the smaller LST.

The schedule is

Activity	Duration	EST	LFT	Float
A	6	0	6	0
B	2	0	10	8
C	5	6	18	7
D	8	6	14	0
E	4	2	14	8
F	3	11	21	7
G	7	14	21	0

The critical path is $A \to D \to G$ (length $6 + 8 + 7 = 21$), so the minimum completion time is 21 days.



Example 3 — unscaffolded

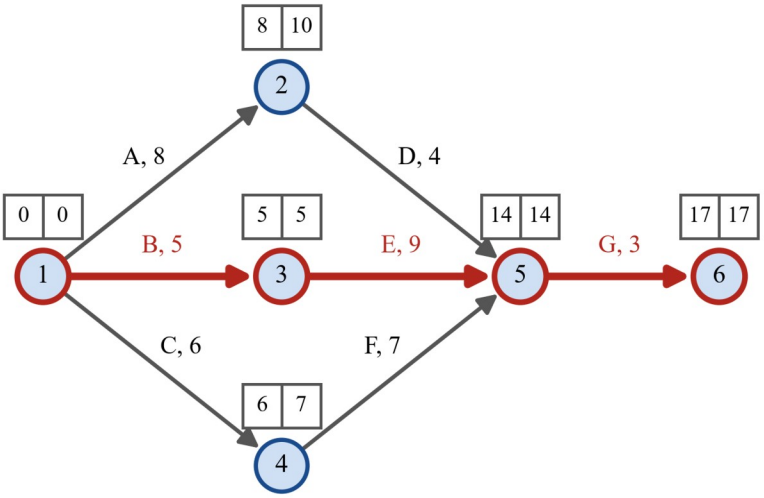
A shop refit has seven activities. Activities A, B and C can all begin immediately; D follows A; E follows B; F follows C; and G follows D, E and F. The durations (in days) are $A\,8$, $B\,5$, $C\,6$, $D\,4$, $E\,9$, $F\,7$, $G\,3$. Find the minimum completion time and the critical path.

Forward scan. The three opening activities start at 0, giving $\text{EFT}(A) = 8$, $\text{EFT}(B) = 5$ and $\text{EFT}(C) = 6$. Then D starts at 8 and finishes at 12, E starts at 5 and finishes at 14, and F starts at 6 and finishes at 13. Activity G must wait for the latest of D, E and F, so $\text{EST}(G) = max(12, 14, 13) = 14$ and $\text{EFT}(G) = 17$. The minimum completion time is 17 days.

Backward scan from 17. For G, LFT $= 17$ and LST $= 14$. Then D, E and F each have LFT $= 14$, giving LST$(D) = 10$, LST$(E) = 5$ and LST$(F) = 7$. Working back to the openers, A leads into D so LFT$(A) =$ LST$(D) = 10$ and LST$(A) = 10 - 8 = 2$; likewise LFT$(B) =$ LST$(E) = 5$ gives LST$(B) = 5 - 5 = 0$, and LFT$(C) =$ LST$(F) = 7$ gives LST$(C) = 7 - 6 = 1$.

The floats are A 2, B 0, C 1, D 2, E 0, F 1, G 0, so the zero-float chain is $B \to E \to G$, of length $5 + 9 + 3 = 17$.

$\therefore$ the minimum completion time is 17 days and the critical path is $B \to E \to G$.



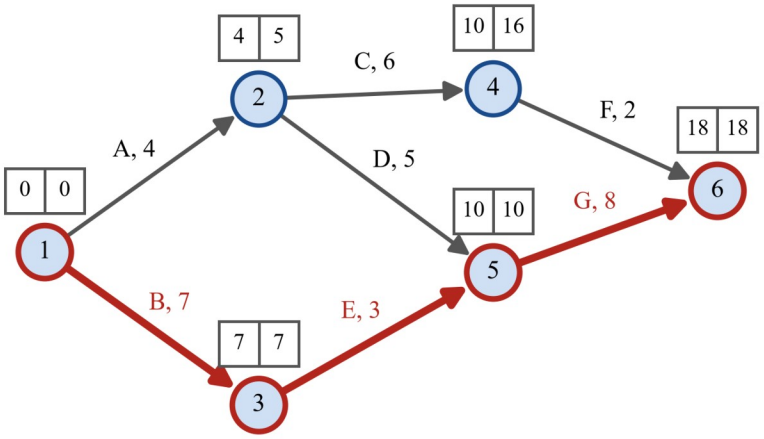
Example 4 — unscaffolded

A festival is organised with seven activities of durations (in days) A 4, B 7, C 6, D 5, E 3, F 2, G 8, where C and D follow A, E follows B, F follows C, and G follows D and E. Find the critical path and minimum completion time, then determine how long the catering activity C could overrun without delaying the festival, and the effect of activity A running 2 days late.

The forward scan gives earliest finishes A 4, B 7, C 10, D 9, E 10, F 12 and, for G, EST $= max(9, 10) = 10$ with EFT $= 18$; the minimum completion time is 18 days. The backward scan from 18 gives latest starts G 10, F 16, C 10, D 5, E 7, A 1, B 0, so the floats are A 1, B 0, C 6, D 1, E 0, F 6, G 0. The zero-float chain is $B \to E \to G$, of length $7 + 3 + 8 = 18$.

Activity C has a float of 6 days, so it could overrun by up to 6 days and still not delay the festival. Activity A has a float of only 1 day; running 2 days late uses up that 1 day of slack and pushes the finish out by $2 - 1 = 1$ day, to 19 days.

$\therefore$ the critical path is $B \to E \to G$ with a minimum completion time of 18 days; C may overrun by 6 days with no effect, but a 2-day delay to A extends the project to 19 days.



Practice questions

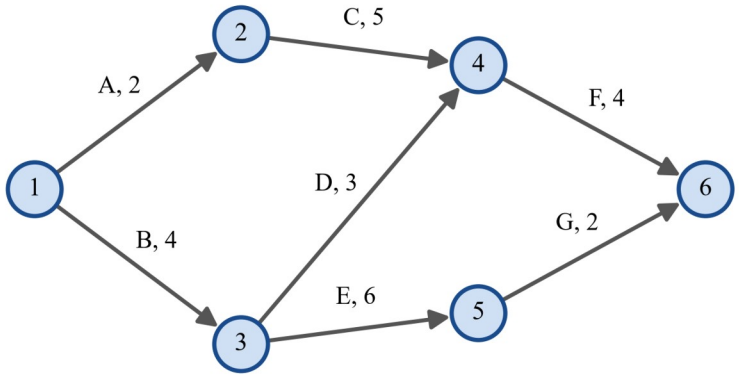
Scaffolded

Q1. A project has five activities with the precedence table below (durations in days).

Activity	Duration	Predecessors
A	3	—
B	5	—
C	6	A
D	2	B
E	4	C, D

(a) Carry out the forward scan, finding the EST and EFT of each activity.
(b) State the minimum completion time.
(c) Carry out the backward scan, finding the LST and LFT of each activity.
(d) Find the float of each activity.
(e) State the critical path.

Q2. The activity network below shows a project with durations in days.



(a) Carry out the forward scan (EST and EFT of each activity).
(b) State the minimum completion time.
(c) Carry out the backward scan (LST and LFT of each activity).
(d) Find the float of each activity and hence state the critical path.

Q3. A project has activities A (5 days), B (6), C (2), D (7) and E (3). C and D both follow A; E follows B and C. (a) Find the EST of every activity. (b) Two activities lead into E. State which one determines the EST of E, and why. (c) State the minimum completion time. (d) State the critical path.

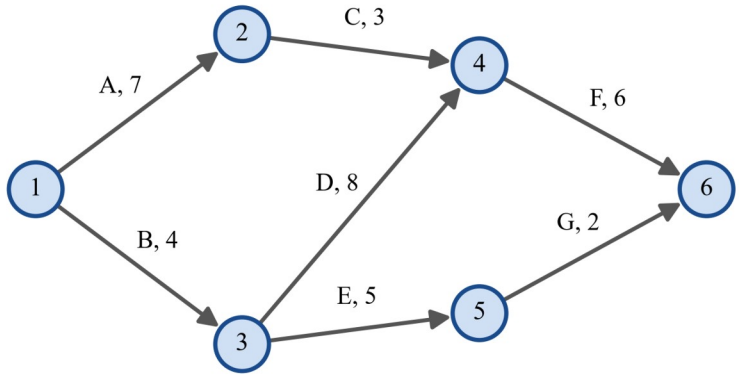
Q4. A project has the precedence table below (durations in days).

Activity	Duration	Predecessors
A	5	—
B	3	—
C	7	A
D	4	B
E	6	C, D

(a) Carry out the forward scan.

(b) Carry out the backward scan.
(c) Find the float of each activity.
(d) State which activities are critical, and write down the critical path.

Q5. The activity network below shows a project with durations in days.



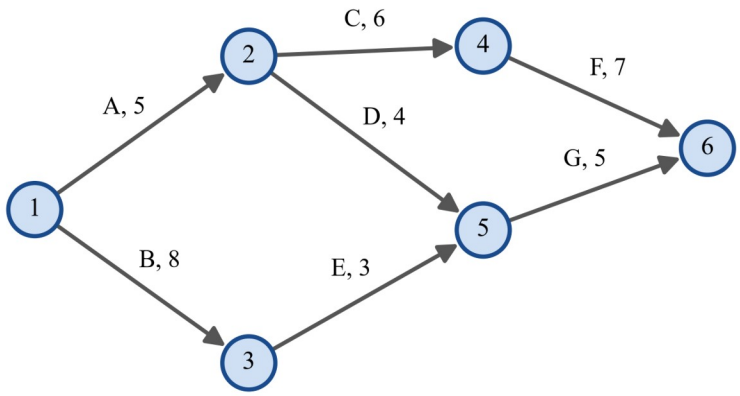
(a) Use the forward scan to find the minimum completion time.
(b) Carry out the backward scan (LST and LFT of each activity).
(c) Find the float of each activity.
(d) State the critical path.

Q6. A project has activities A (6 days), B (3), C (2), D (7) and E (5). C follows A, D follows B, and E follows C and D. (a) Carry out the forward and backward scans and tabulate the schedule. (b) State the critical path and the minimum completion time. (c) Activity A has the longest duration of the two opening activities, yet it is not on the critical path. Explain why.

Unscaffolded

Q7. A project has activities A (6 days), B (4), C (3), D (7) and E (5). C and D follow A; E follows B and C. Find the minimum completion time, the critical path, and the float of each non-critical activity.

Q8. For the activity network below (durations in days), determine the schedule (EST, LFT and float of each activity), the critical path and the minimum completion time.



Q9. A project has activities A (3 days), B (6), C (4), D (7), E (5), F (8) and G (2). Activities A, B and C start immediately; D follows A, E follows B, F follows C, and G follows D, E and F. Find the minimum completion time, the critical path, and the float of activity D.

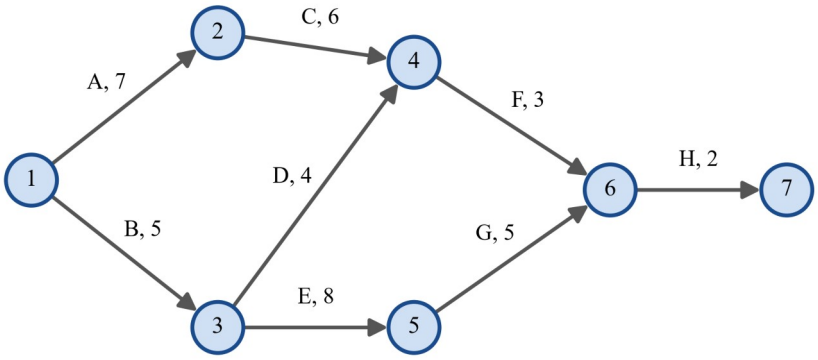
Q10. A project has the precedence table below (durations in days).

Activity	Duration	Predecessors
A	9	—
B	5	—
C	4	A
D	8	B
E	6	A
F	3	C, D
G	7	E

Find the critical path and the minimum completion time, and determine by how many days activity D could be delayed without delaying the project.

Q11. A project has activities A (4 days), B (3), C (2), D (6), E (5) and F (4). C and D follow A; E follows B and C; F follows D and E. Find the critical path and the minimum completion time, and state whether activity D is critical.

Q12. For the activity network below (durations in days), find the EST, LFT and float of every activity, the critical path and the minimum completion time.

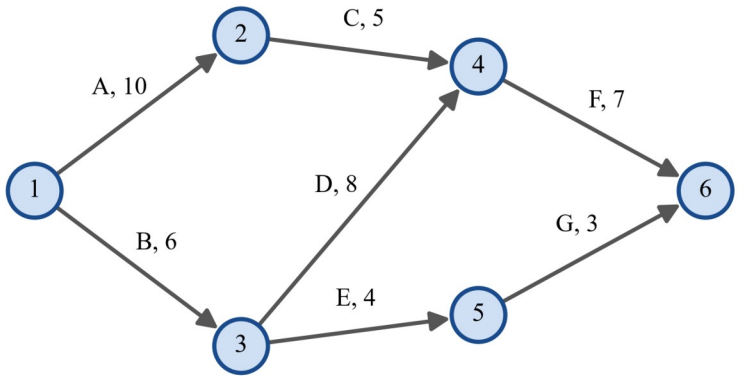


Q13. A project has activities A (5 days), B (7), C (8), D (3), E (6), F (4) and G (9). C and D follow A; E follows B; F follows C; G follows D and E. Find the critical path, the minimum completion time, and the float of activity C.

Q14. A project has activities A (6 days), B (4), C (5), D (7), E (3), F (6) and G (4). Activities A, B and C start immediately; D follows A, E follows B, F follows C, and G follows D, E and F. (a) Find the critical path and the minimum completion time. (b) State the effect on the completion time if activity B finishes 3 days late. (c) State the effect on the completion time if activity A finishes 2 days late.

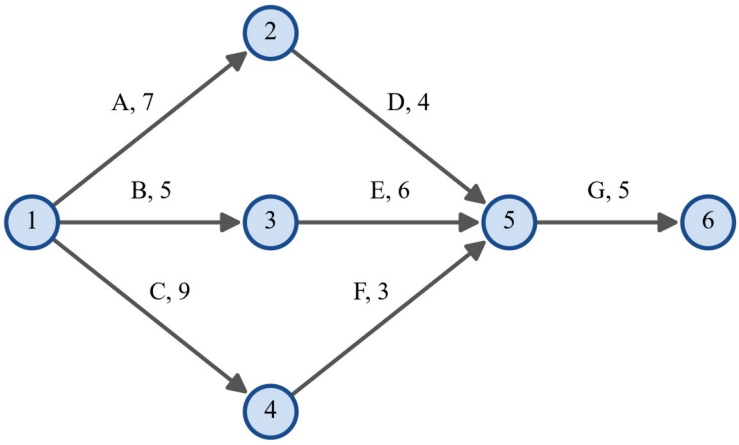
Q15. A project has activities A (3 days), B (8), C (5), D (2), E (6) and F (4). C follows A; D and F follow B; E follows C and D. Find the critical path, the minimum completion time, and the float of activity F.

Q16. For the activity network below (durations in days), find the schedule, the critical path, the minimum completion time and the float of activity E.



Q17. A project has activities A (4 days), B (6), C (3), D (5), E (8), F (7) and G (2). C and D follow A; E follows B; F follows C; G follows D and E. Find the critical path and the minimum completion time, and state the floats of activities D and F.

Q18. For the activity network below (durations in days), find the schedule, the critical path and the minimum completion time.



Q19. A project has activities A (5 days), B (3), C (6), D (8), E (4) and F (7). C follows A; D and F follow B; E follows C and D. (a) Find the minimum completion time and list the critical activities. (b) This project has **two** critical paths. Write both of them down. (c) State the float of activity F.

Q20. A project has the precedence table below (durations in days).

Activity	Duration	Predecessors
A	4	—
B	6	—
C	7	A
D	5	A
E	8	B
F	3	C
G	6	D, E

(a) Find the EST, LFT and float of every activity.
(b) State the critical path and the minimum completion time.
(c) State which activities have the greatest float, and explain what that float means in practical terms.

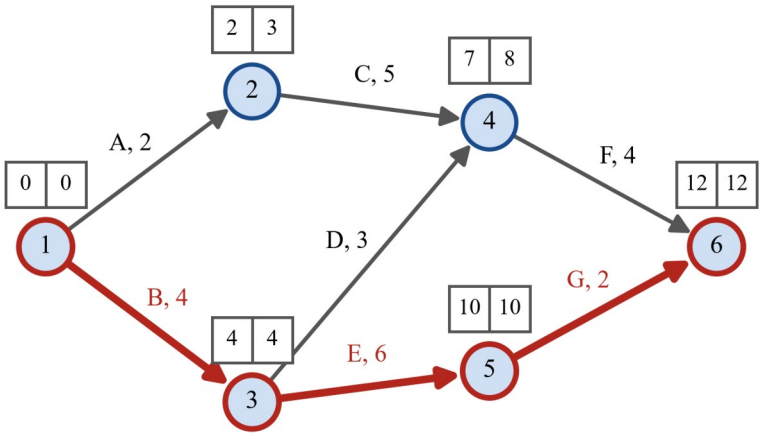
Answers

Q1. (a) $A\,0/3$, $B\,0/5$, $C\,3/9$, $D\,5/7$, $E\,9/13$ (EST/EFT). (b) 13 days. (c) $A\,0/3$, $B\,2/7$, $C\,3/9$, $D\,7/9$, $E\,9/13$ (LST/LFT). (d) $A\,0$, $B\,2$, $C\,0$, $D\,2$, $E\,0$. (e) $A \to C \to E$. **Q2.** (a) EFT: $A\,2$, $B\,4$, $C\,7$, $D\,7$, $E\,10$, $F\,11$, $G\,12$. (b) 12 days. (c) LST: $A\,1$, $B\,0$, $C\,3$, $D\,5$, $E\,4$, $F\,8$, $G\,10$. (d) Floats $A\,1$, $B\,0$, $C\,1$, $D\,1$, $E\,0$, $F\,1$, $G\,0$; critical path $B \to E \to G$. **Q3.** (a) $A\,0$, $B\,0$, $C\,5$, $D\,5$, $E\,7$. (b) C, since $\text{EFT}(C) = 7 > \text{EFT}(B) = 6$; E waits for the later one. (c) 12 days. (d) $A \to D$. **Q4.** (a) EFT: $A\,5$, $B\,3$, $C\,12$, $D\,7$, $E\,18$. (b) LST: $A\,0$, $B\,5$, $C\,5$, $D\,8$, $E\,12$. (c) $A\,0$, $B\,5$, $C\,0$, $D\,5$, $E\,0$. (d) Critical A, C, E; path $A \to C \to E$. **Q5.** (a) 18 days. (b) LST/LFT: $A\,2/9$, $B\,0/4$, $C\,9/12$, $D\,4/12$, $E\,11/16$, $F\,12/18$, $G\,16/18$. (c) $A\,2$, $B\,0$, $C\,2$, $D\,0$, $E\,7$, $F\,0$, $G\,7$. (d) $B \to D \to F$. **Q6.** (a) EST/EFT $A\,0/6$, $B\,0/3$, $C\,6/8$, $D\,3/10$, $E\,10/15$; LST/LFT $A\,2/8$, $B\,0/3$, $C\,8/10$, $D\,3/10$, $E\,10/15$. (b) $B \to D \to E$, 15 days. (c) A lies on the shorter path $A \to C$ (8 days) into the merge, while $B \to D$ (10 days) is longer and sets the schedule, so A gains a float of 2 days; the critical path is the longest path, not the one with the single longest activity. **Q7.** Minimum completion 14 days; critical path $A \to C \to E$; non-critical floats $B\,5$, $D\,1$. **Q8.** Schedule (EST/LFT/float): $A\,0/5/0$, $B\,0/10/2$, $C\,5/11/0$, $D\,5/13/4$, $E\,8/13/2$, $F\,11/18/0$, $G\,11/18/2$. Critical path $A \to C \to F$; minimum completion 18 days. **Q9.** Minimum completion 14 days; critical path $C \to F \to G$; float of D is 2 days. **Q10.** Critical path $A \to E \to G$; minimum completion 22 days; D may be delayed up to 6 days. **Q11.** Critical path $A \to C \to E \to F$; minimum completion 15 days; D is not critical (float 1). **Q12.** EST/LFT/float: $A\,0/9/2$, $B\,0/5/0$, $C\,7/15/2$, $D\,5/15/6$, $E\,5/13/0$, $F\,13/18/2$, $G\,13/18/0$, $H\,18/20/0$. Critical path $B \to E \to G \to H$; minimum completion 20 days. **Q13.** Critical path $B \to E \to G$; minimum completion 22 days; float of C is 5 days. **Q14.** (a) $A \to D \to G$, 17 days. (b) No effect: B has a float of 6 days. (c) The finish is pushed out by 2 days, to 19 days (A is critical). **Q15.** Critical path $B \to D \to E$; minimum completion 16 days; float of F is 4 days. **Q16.** EST/LFT/float: $A\,0/10/0$, $B\,0/7/1$, $C\,10/15/0$, $D\,6/15/1$, $E\,6/19/9$, $F\,15/22/0$, $G\,10/22/9$. Critical path $A \to C \to F$; minimum completion 22 days; float of E is 9 days. **Q17.** Critical path $B \to E \to G$; minimum completion 16 days; float of D is 5 days and float of F is 2 days. **Q18.** Schedule (EST/LFT/float): $A\,0/8/1$, $B\,0/6/1$, $C\,0/9/0$, $D\,7/12/1$, $E\,5/12/1$, $F\,9/12/0$, $G\,12/17/0$. Critical path $C \to F \to G$; minimum completion 17 days. **Q19.** (a) 15 days; critical activities A, B, C, D, E. (b) $A \to C \to E$ and $B \to D \to E$ (both length 15). (c) Float of F is 5 days. **Q20.** (a) EST/LFT/float: $A\,0/9/5$, $B\,0/6/0$, $C\,4/17/6$, $D\,4/14/5$, $E\,6/14/0$, $F\,11/20/6$, $G\,14/20/0$. (b) $B \to E \to G$, 20 days. (c) C and F each have a float of 6 days (the greatest); either could be delayed, or take up to 6 days longer, without delaying the project.

Fully-worked solutions

Q1. *Forward scan:* A and B start at 0, so $\text{EFT}(A) = 3$ and $\text{EFT}(B) = 5$. Then C follows A (EST = 3, EFT = 9) and D follows B (EST = 5, EFT = 7). E waits for both, so $\text{EST}(E) = max(9, 7) = 9$ and $\text{EFT}(E) = 13$. The minimum completion time is 13 days. *Backward scan* from 13: $\text{LST}(E) = 9$, then $\text{LFT}(C) = \text{LFT}(D) = 9$ giving $\text{LST}(C) = 3$, $\text{LST}(D) = 7$; and $\text{LST}(A) = 3 - 3 = 0$, $\text{LST}(B) = 7 - 5 = 2$. *Floats:* $A\,0$, $B\,2$, $C\,0$, $D\,2$, $E\,0$. The critical path is $A \to C \to E$.

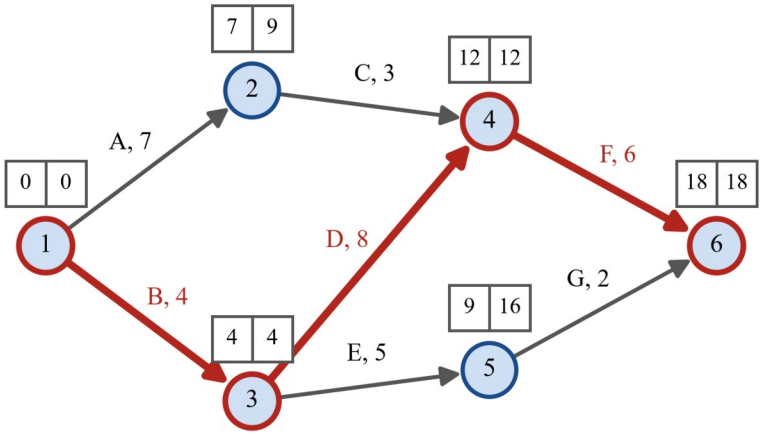
Q2. *Forward scan:* $A\,0/2$, $B\,0/4$; C (after A) $2/7$, D (after B) $4/7$, E (after B) $4/10$; F (after C and D) $\text{EST} = max(7, 7) = 7$, EFT = 11; G (after E) 10/12. Minimum completion 12 days. *Backward scan* from 12: F and G are last, so $\text{LST}(F) = 8$ and $\text{LST}(G) = 10$; then $\text{LFT}(C) = \text{LFT}(D) = 8$ giving $\text{LST}(C) = 3$, $\text{LST}(D) = 5$, and $\text{LFT}(E) = 10$ giving $\text{LST}(E) = 4$; finally $\text{LST}(A) = 3 - 2 = 1$ and, since B feeds D and E, $\text{LFT}(B) = min(5, 4) = 4$ so $\text{LST}(B) = 0$. *Floats:* $A\,1$, $B\,0$, $C\,1$, $D\,1$, $E\,0$, $F\,1$, $G\,0$; the critical path is $B \to E \to G$.



Q3. *Forward scan:* $A\,0/5$, $B\,0/6$; C (after A) $5/7$, D (after A) $5/12$; E follows B and C, so $\text{EST}(E)=max(\text{EFT}(B),\text{EFT}(C))=max(6,7)=7$ and $\text{EFT}(E)=10$. (a) EST values $A\,0$, $B\,0$, $C\,5$, $D\,5$, $E\,7$. (b) The EST of E is set by C, whose EFT (7) exceeds that of B (6); a merge activity cannot start until the *later* predecessor finishes. (c) The largest EFT is $\text{EFT}(D)=12$, so the minimum completion time is 12 days. (d) The longest path is $A \rightarrow D$ ($5+7=12$), which is the critical path.

Q4. *Forward scan:* $A\,0/5$, $B\,0/3$; C (after A) $5/12$, D (after B) $3/7$; E (after C and D) $\text{EST}=max(12,7)=12$, $\text{EFT}=18$. Minimum completion 18 days. *Backward scan* from 18: $\text{LST}(E)=12$; $\text{LFT}(C)=\text{LFT}(D)=12$ so $\text{LST}(C)=5$ and $\text{LST}(D)=8$; $\text{LST}(A)=5-5=0$, $\text{LST}(B)=8-3=5$. *Floats:* $A\,0$, $B\,5$, $C\,0$, $D\,5$, $E\,0$. The critical activities are A, C, E, giving the critical path $A \rightarrow C \rightarrow E$.

Q5. *Forward scan:* $A\,0/7$, $B\,0/4$; C (after A) $7/10$, D (after B) $4/12$, E (after B) $4/9$; F (after C and D) $\text{EST}=max(10,12)=12$, $\text{EFT}=18$; G (after E) $9/11$. (a) The minimum completion time is 18 days. *Backward scan* from 18: $\text{LST}(F)=12$, $\text{LST}(G)=16$; $\text{LFT}(C)=\text{LFT}(D)=12$ so $\text{LST}(C)=9$, $\text{LST}(D)=4$; $\text{LFT}(E)=16$ so $\text{LST}(E)=11$; $\text{LST}(A)=9-7=2$, and B feeds D and E so $\text{LFT}(B)=min(4,11)=4$, $\text{LST}(B)=0$. (b)/(c) *Floats:* $A\,2$, $B\,0$, $C\,2$, $D\,0$, $E\,7$, $F\,0$, $G\,7$. (d) The critical path is $B \rightarrow D \rightarrow F$.

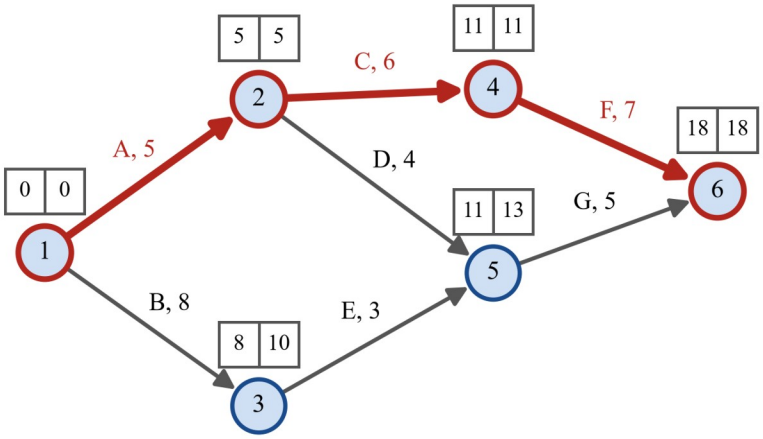


Q6. *Forward scan:* $A\,0/6$, $B\,0/3$; C (after A) $6/8$, D (after B) $3/10$; E (after C and D) $\text{EST}=max(8,10)=10$, $\text{EFT}=15$. *Backward scan* from 15: $\text{LST}(E)=10$; $\text{LFT}(C)=\text{LFT}(D)=10$ so $\text{LST}(C)=8$, $\text{LST}(D)=3$; $\text{LST}(A)=8-6=2$, $\text{LST}(B)=3-3=0$. (a) Schedule EST/LFT/float: $A\,0/8/2$, $B\,0/3/0$, $C\,6/10/2$, $D\,3/10/0$, $E\,10/15/0$. (b) The critical path is $B \rightarrow D \rightarrow E$ and the minimum completion time is 15 days. (c) The two paths into the merge at E are $A \rightarrow C$ (length $6+2=8$) and $B \rightarrow D$ (length $3+7=10$). The critical path is the **longest** path, and $B \rightarrow D$ is longer even though A alone is longer than B; so A carries a float of 2 days and is not critical.

Q7. *Forward scan:* $A\,0/6$, $B\,0/4$; C (after A) $6/9$, D (after A) $6/13$; E follows B and C, so $\text{EST}(E)=max(4,9)=9$, $\text{EFT}(E)=14$. Minimum completion 14 days. *Backward scan* from 14: D is last,

$\text{LST}(D)=7$; $\text{LST}(E)=9$; $\text{LFT}(C)=9$ so $\text{LST}(C)=6$; $\text{LFT}(B)=9$ so $\text{LST}(B)=5$; A feeds C and D, $\text{LFT}(A)=min(6,7)=6$, $\text{LST}(A)=0$. *Floats:* $A\,0$, $B\,5$, $C\,0$, $D\,1$, $E\,0$. The critical path is $A \to C \to E$ (14 days); the non-critical floats are $B\,5$ and $D\,1$.

Q8. *Forward scan:* $A\,0/5$, $B\,0/8$; C (after A) $5/11$, D (after A) $5/9$, E (after B) $8/11$; F (after C) $11/18$; G (after D and E) $\text{EST}=max(9,11)=11$, $\text{EFT}=16$. Minimum completion 18 days (largest EFT is $\text{EFT}(F)=18$). *Backward scan* from 18: $\text{LST}(F)=11$, $\text{LST}(G)=13$; $\text{LFT}(C)=11$ so $\text{LST}(C)=5$; $\text{LFT}(D)=\text{LFT}(E)=13$ so $\text{LST}(D)=9$, $\text{LST}(E)=10$; $\text{LST}(A)=5-5=0$ (via C), $\text{LST}(B)=10-8=2$. *Schedule* (EST/LFT/float): $A\,0/5/0$, $B\,0/10/2$, $C\,5/11/0$, $D\,5/13/4$, $E\,8/13/2$, $F\,11/18/0$, $G\,11/18/2$. The critical path is $A \to C \to F$ and the minimum completion time is 18 days.

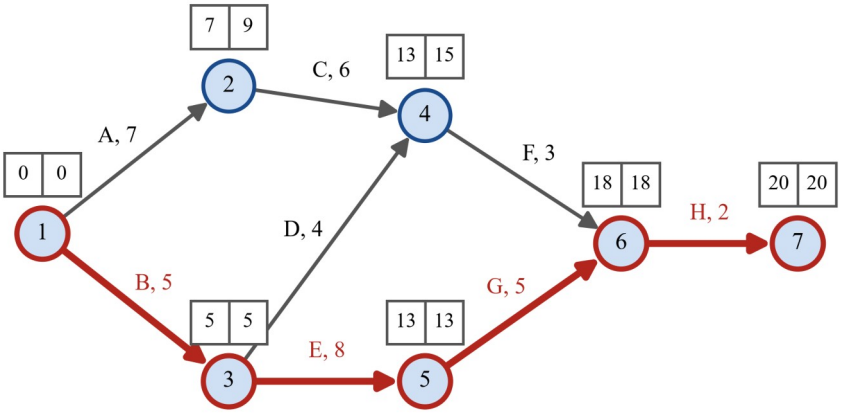


Q9. *Forward scan:* A, B, C start at 0 with EFTs 3, 6, 4; D (after A) $3/10$, E (after B) $6/11$, F (after C) $4/12$; G follows D, E, F, so $\text{EST}(G)=max(10,11,12)=12$ and $\text{EFT}(G)=14$. Minimum completion 14 days. *Backward scan* from 14: $\text{LST}(G)=12$; $\text{LFT}(D)=\text{LFT}(E)=\text{LFT}(F)=12$ so $\text{LST}(D)=5$, $\text{LST}(E)=7$, $\text{LST}(F)=4$; then $\text{LST}(A)=5-3=2$, $\text{LST}(B)=7-6=1$, $\text{LST}(C)=4-4=0$. The critical path is $C \to F \to G$; the float of D is $\text{LST}(D)-\text{EST}(D)=5-3=2$ days.

Q10. *Forward scan:* $A\,0/9$, $B\,0/5$; C (after A) $9/13$, E (after A) $9/15$, D (after B) $5/13$; F (after C and D) $\text{EST}=max(13,13)=13$, $\text{EFT}=16$; G (after E) $15/22$. Minimum completion 22 days. *Backward scan* from 22: $\text{LST}(F)=19$, $\text{LST}(G)=15$; $\text{LFT}(C)=\text{LFT}(D)=19$ so $\text{LST}(C)=15$, $\text{LST}(D)=11$; $\text{LFT}(E)=15$ so $\text{LST}(E)=9$; A feeds C and E, $\text{LFT}(A)=min(15,9)=9$, $\text{LST}(A)=0$; $\text{LST}(B)=11-5=6$. The critical path is $A \to E \to G$. Activity D has float $\text{LST}(D)-\text{EST}(D)=11-5=6$, so it may be delayed up to 6 days.

Q11. *Forward scan:* $A\,0/4$, $B\,0/3$; C (after A) $4/6$, D (after A) $4/10$; E follows B and C, $\text{EST}=max(3,6)=6$, $\text{EFT}=11$; F follows D and E, $\text{EST}=max(10,11)=11$, $\text{EFT}=15$. Minimum completion 15 days. *Backward scan* from 15: $\text{LST}(F)=11$; $\text{LFT}(D)=\text{LFT}(E)=11$ so $\text{LST}(D)=5$, $\text{LST}(E)=6$; $\text{LFT}(C)=6$ so $\text{LST}(C)=4$; $\text{LFT}(B)=6$ so $\text{LST}(B)=3$; A feeds C and D, $\text{LFT}(A)=min(4,5)=4$, $\text{LST}(A)=0$. *Floats:* $A\,0$, $B\,3$, $C\,0$, $D\,1$, $E\,0$, $F\,0$. The critical path is $A \to C \to E \to F$; the minimum completion time is 15 days. Activity D has float 1, so it is **not** critical.

Q12. *Forward scan:* $A\,0/7$, $B\,0/5$; C (after A) $7/13$, D (after B) $5/9$, E (after B) $5/13$; F (after C and D) $\text{EST}=max(13,9)=13$, $\text{EFT}=16$; G (after E) $13/18$; H (after F and G) $\text{EST}=max(16,18)=18$, $\text{EFT}=20$. Minimum completion 20 days. *Backward scan* from 20: $\text{LST}(H)=18$; $\text{LFT}(F)=\text{LFT}(G)=18$ so $\text{LST}(F)=15$, $\text{LST}(G)=13$; $\text{LFT}(C)=15$ so $\text{LST}(C)=9$; $\text{LFT}(D)=15$ so $\text{LST}(D)=11$; $\text{LFT}(E)=13$ so $\text{LST}(E)=5$; $\text{LST}(A)=9-7=2$; B feeds D and E, $\text{LFT}(B)=min(11,5)=5$, $\text{LST}(B)=0$. *Floats:* $A\,2$, $B\,0$, $C\,2$, $D\,6$, $E\,0$, $F\,2$, $G\,0$, $H\,0$. The critical path is $B \to E \to G \to H$ and the minimum completion time is 20 days.

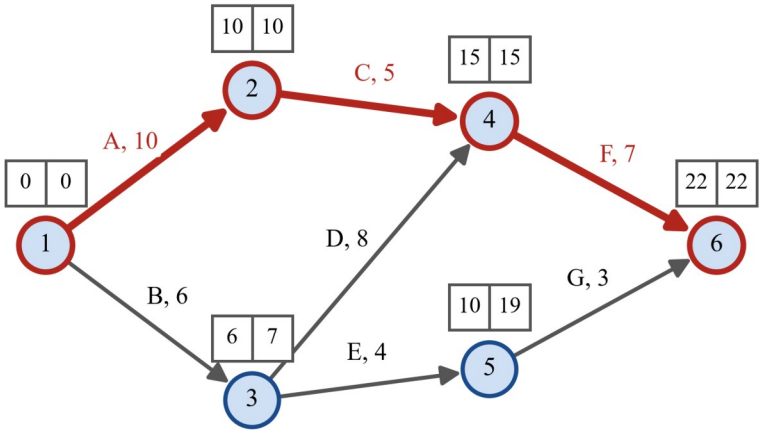


Q13. *Forward scan:* $A\,0/5$, $B\,0/7$; C (after A) $5/13$, D (after A) $5/8$, E (after B) $7/13$; F (after C) $13/17$; G (after D and E) $\text{EST}=max(8,13)=13$, $\text{EFT}=22$. Minimum completion 22 days. *Backward scan* from 22: F is last, $\text{LST}(F)=18$; $\text{LST}(G)=13$; $\text{LFT}(C)=18$ so $\text{LST}(C)=10$; $\text{LFT}(D)=\text{LFT}(E)=13$ so $\text{LST}(D)=10$, $\text{LST}(E)=7$; A feeds C and D, $\text{LFT}(A)=min(10,10)=10$, $\text{LST}(A)=5$; $\text{LST}(B)=7-7=0$. The critical path is $B \to E \to G$; the float of C is $\text{LST}(C)-\text{EST}(C)=10-5=5$ days.

Q14. *Forward scan:* A, B, C start at 0 with EFTs 6, 4, 5; D (after A) $6/13$, E (after B) $4/7$, F (after C) $5/11$; G follows D, E, F, $\text{EST}=max(13,7,11)=13$, $\text{EFT}=17$. *Backward scan* from 17: $\text{LST}(G)=13$; $\text{LFT}(D)=\text{LFT}(E)=\text{LFT}(F)=13$ so $\text{LST}(D)=6$, $\text{LST}(E)=10$, $\text{LST}(F)=7$; $\text{LST}(A)=6-6=0$, $\text{LST}(B)=10-4=6$, $\text{LST}(C)=7-5=2$. (a) The critical path is $A \to D \to G$ and the minimum completion time is 17 days. (b) Activity B has float 6 days, so finishing 3 days late has no effect — the project still finishes in 17 days. (c) Activity A is critical (float 0); finishing 2 days late pushes the whole project out by 2 days, to 19 days.

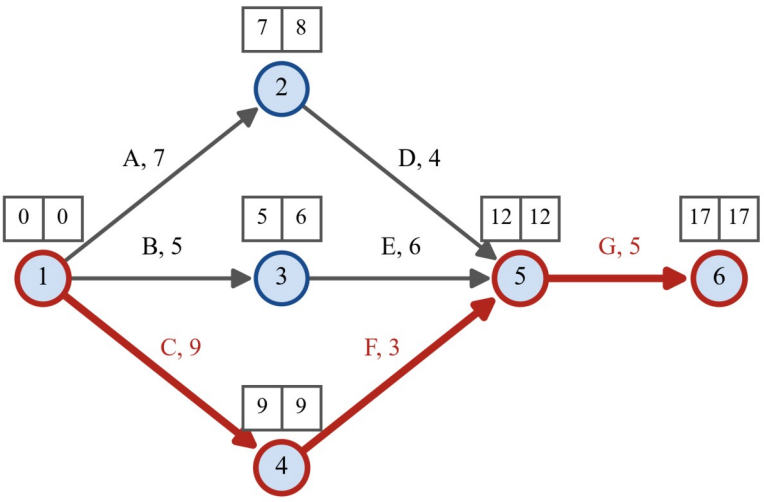
Q15. *Forward scan:* $A\,0/3$, $B\,0/8$; C (after A) $3/8$, D (after B) $8/10$, F (after B) $8/12$; E (after C and D) $\text{EST}=max(8,10)=10$, $\text{EFT}=16$. Minimum completion 16 days. *Backward scan* from 16: F is last, $\text{LST}(F)=12$; $\text{LST}(E)=10$; $\text{LFT}(C)=\text{LFT}(D)=10$ so $\text{LST}(C)=5$, $\text{LST}(D)=8$; $\text{LST}(A)=5-3=2$; B feeds D and F, $\text{LFT}(B)=min(8,12)=8$, $\text{LST}(B)=0$. The critical path is $B \to D \to E$; the float of F is $\text{LST}(F)-\text{EST}(F)=12-8=4$ days.

Q16. *Forward scan:* $A\,0/10$, $B\,0/6$; C (after A) $10/15$, D (after B) $6/14$, E (after B) $6/10$; F (after C and D) $\text{EST}=max(15,14)=15$, $\text{EFT}=22$; G (after E) $10/13$. Minimum completion 22 days. *Backward scan* from 22: $\text{LST}(F)=15$, $\text{LST}(G)=19$; $\text{LFT}(C)=\text{LFT}(D)=15$ so $\text{LST}(C)=10$, $\text{LST}(D)=7$; $\text{LFT}(E)=19$ so $\text{LST}(E)=15$; $\text{LST}(A)=10-10=0$; B feeds D and E, $\text{LFT}(B)=min(7,15)=7$, $\text{LST}(B)=1$. *Schedule* (EST/LFT/float): $A\,0/10/0$, $B\,0/7/1$, $C\,10/15/0$, $D\,6/15/1$, $E\,6/19/9$, $F\,15/22/0$, $G\,10/22/9$. The critical path is $A \to C \to F$, the minimum completion time is 22 days, and the float of E is 9 days.



Q17. *Forward scan:* $A\,0/4$, $B\,0/6$; C (after A) $4/7$, D (after A) $4/9$, E (after B) $6/14$; F (after C) $7/14$; G (after D and E) $\text{EST} = max(9,14) = 14$, $\text{EFT} = 16$. Minimum completion 16 days. *Backward scan* from 16: F is last, $\text{LST}(F) = 9$; $\text{LST}(G) = 14$; $\text{LFT}(C) = 9$ so $\text{LST}(C) = 6$; $\text{LFT}(D) = \text{LFT}(E) = 14$ so $\text{LST}(D) = 9$, $\text{LST}(E) = 6$; A feeds C and D, $\text{LFT}(A) = min(6,9) = 6$, $\text{LST}(A) = 2$; $\text{LST}(B) = 6 - 6 = 0$. The critical path is $B \to E \to G$ (16 days). The float of D is $9 - 4 = 5$ days and the float of F is $9 - 7 = 2$ days.

Q18. *Forward scan:* A, B, C start at 0 with EFTs 7, 5, 9; D (after A) $7/11$, E (after B) $5/11$, F (after C) $9/12$; G follows D, E, F, $\text{EST} = max(11,11,12) = 12$, $\text{EFT} = 17$. Minimum completion 17 days. *Backward scan* from 17: $\text{LST}(G) = 12$; $\text{LFT}(D) = \text{LFT}(E) = \text{LFT}(F) = 12$ so $\text{LST}(D) = 8$, $\text{LST}(E) = 6$, $\text{LST}(F) = 9$; $\text{LST}(A) = 8 - 7 = 1$, $\text{LST}(B) = 6 - 5 = 1$, $\text{LST}(C) = 9 - 9 = 0$. *Schedule* (EST/LFT/float): $A\,0/8/1$, $B\,0/6/1$, $C\,0/9/0$, $D\,7/12/1$, $E\,5/12/1$, $F\,9/12/0$, $G\,12/17/0$. The critical path is $C \to F \to G$ and the minimum completion time is 17 days.



Q19. *Forward scan:* $A\,0/5$, $B\,0/3$; C (after A) $5/11$, D (after B) $3/11$, F (after B) $3/10$; E (after C and D) $\text{EST} = max(11,11) = 11$, $\text{EFT} = 15$. (a) The minimum completion time is 15 days. *Backward scan* from 15: $\text{LST}(E) = 11$; $\text{LFT}(C) = \text{LFT}(D) = 11$ so $\text{LST}(C) = 5$, $\text{LST}(D) = 3$; $\text{LST}(A) = 5 - 5 = 0$; B feeds D and F; $\text{LFT}(F) = 15$ so $\text{LST}(F) = 8$; $\text{LFT}(B) = min(3,8) = 3$, $\text{LST}(B) = 0$. *Floats:* $A\,0$, $B\,0$, $C\,0$, $D\,0$, $E\,0$, $F\,5$, so the critical activities are A, B, C, D, E. (b) Both routes into E have length 11 ($A \to C$: $5+6$; $B \to D$: $3+8$), so there are two critical paths: $A \to C \to E$ and $B \to D \to E$. (c) The float of F is $\text{LST}(F) - \text{EST}(F) = 8 - 3 = 5$ days.

Q20. *Forward scan:* $A\,0/4$, $B\,0/6$; C (after A) $4/11$, D (after A) $4/9$, E (after B) $6/14$; F (after C) $11/14$; G (after D and E) $\text{EST} = max(9,14) = 14$, $\text{EFT} = 20$. Minimum completion 20 days. *Backward scan* from 20: F is last, $\text{LST}(F) = 17$; $\text{LST}(G) = 14$; $\text{LFT}(C) = 17$ so $\text{LST}(C) = 10$; $\text{LFT}(D) = \text{LFT}(E) = 14$ so $\text{LST}(D) = 9$, $\text{LST}(E) = 6$; A feeds C and D, $\text{LFT}(A) = min(10,9) = 9$, $\text{LST}(A) = 5$; $\text{LST}(B) = 6 - 6 = 0$. (a) *Schedule* (EST/LFT/float): $A\,0/9/5$, $B\,0/6/0$, $C\,4/17/6$, $D\,4/14/5$, $E\,6/14/0$, $F\,11/20/6$, $G\,14/20/0$. (b) The critical path is $B \to E \to G$ and the minimum completion time is 20 days. (c) Activities C and F share the greatest float, 6 days. A float of 6 means either activity could start up to 6 days late, or take up to 6 days longer than planned, without delaying the completion of the whole project.

Theory

In an earlier section a project was modelled by an **activity network** built from a precedence table. Forward and backward scanning gave the earliest starting time (EST) and latest starting time (LST) of each activity, and these identified the **critical path** — the longest path through the network — together with the **float** (spare time) of every non-critical activity. The length of the critical path is the shortest possible **completion time** for the whole project.

Often a project must be finished sooner than this. Extra workers, overtime or additional machinery can **shorten** individual activities, but at a price. Reducing the duration of one or more activities in order to reduce the overall completion time is called **crashing** the project.

The cost of crashing.

For a crashable activity we are told:

- its normal duration;
- its **crash cost per unit time** — the cost of shortening it by one day (or week); and
- its **crash limit** — the greatest number of time units by which it can be shortened.

Some activities cannot be crashed at all. The aim is to achieve a required completion time for the **least extra cost**.

Only critical activities matter.

The completion time equals the length of the critical path, so shortening the project means shortening the critical path.

- Crashing an activity that lies **on** the critical path reduces the completion time.
- Crashing a **non-critical** activity (one with float) does **not** change the length of the critical path, so the completion time is unchanged and the money is wasted.

It follows that the cheapest activity overall is **not** necessarily the one to crash: it may have float. Always work with the **critical** activities.

Choosing what to crash. To reduce the completion time as cheaply as possible:

1. Find the critical path and the completion time.
2. Among the activities **on the critical path**, crash the one with the **smallest crash cost per unit time**.
3. Keep crashing that activity until **either**
    - it reaches its crash limit, **or**
    - a **second path becomes critical** — whichever happens first.
4. Re-draw the network with the reduced durations and find the **new** critical path and completion time. If a further reduction is required, repeat from step 2.

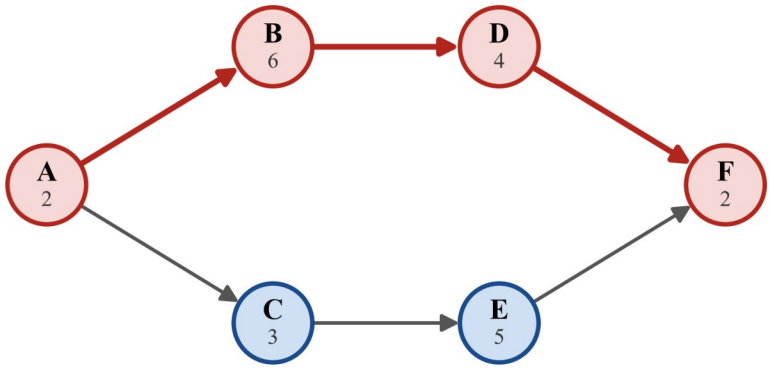
How far can one activity help? A single critical activity can only shorten the completion time until the next-longest path catches up. If the critical path is currently L and the next-longest path is L', then crashing activities on that one path can reduce the completion time by at most $L - L'$ before a **new critical path** appears. Once two or more paths are critical, **every** critical path must be shortened at the same time: crash an activity that is **common to all** the critical paths, or crash one activity **on each** path — whichever is cheaper.

Extra cost and new time.

$$\text{extra cost} = \sum(\text{units crashed}) \times (\text{crash cost per unit time}),$$

and the **new completion time** is the length of the new critical path. Because every activity has a crash limit, there is a shortest achievable completion time; beyond it no amount of money will help.

The activity network below has activities labelled with their durations (in days). Forward and backward scanning give the critical path A–B–D–F, highlighted, of length $2+6+4+2=14$ days; the path A–C–E–F is $2+3+5+2=12$ days, so C and E each have 2 days of float. Suppose activity D is on the critical path and can be crashed by up to 2 days at \$200 per day. Crashing D by 2 days shortens the critical path to 12 days for an extra cost of $2\times\$200=\$400$; the path A–C–E–F is now also 12 days, so both paths are critical and no further saving is possible by crashing D alone.

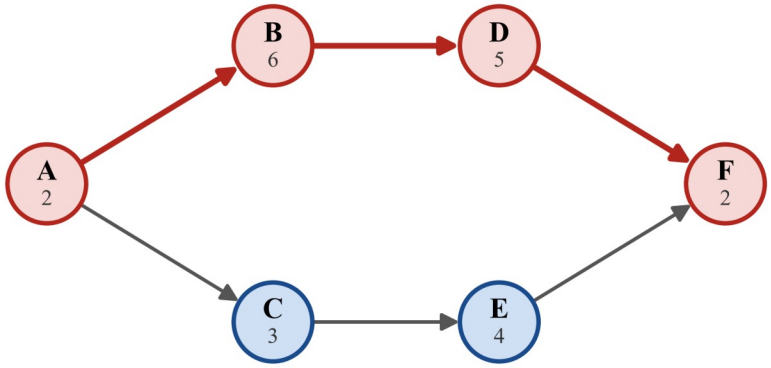


Worked examples

Example 1 — scaffolded

A project has the activities, durations (in days) and crashing information shown. An em dash means the activity cannot be crashed.

Activity	Duration	Immediate predecessor(s)	Crash cost per day	Crash limit (days)
A	2	—	\$400	1
B	6	A	\$300	3
C	3	A	\$200	2
D	5	B	\$250	2
E	4	C	\$250	2
F	2	D, E	\$350	1



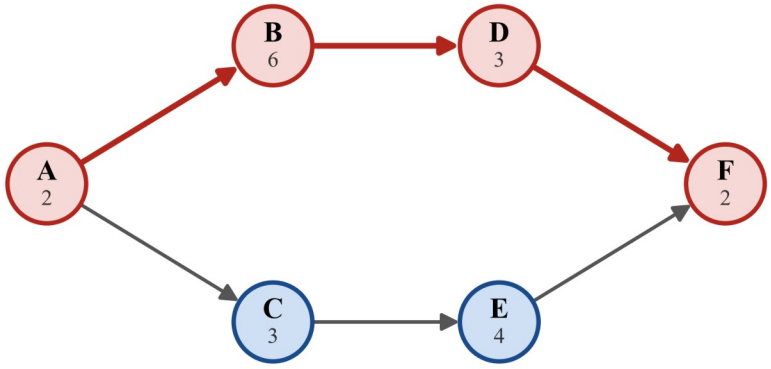
(a) State the critical path and the completion time.

(b) Which single activity should be crashed first to reduce the completion time most cheaply?

(c) Crash that activity as far as possible. State the new completion time and the extra cost.

(d) Explain why crashing activity C would **not** reduce the completion time.

Working	Comment
Path A–B–D–F: $2+6+5+2=15$. Path A–C–E–F: $2+3+4+2=11$.	List each path from start to finish and add the durations.
Critical path A–B–D–F; completion time 15 days.	The critical path is the longest path.
Critical activities are A, B, D, F, costing \$400, \$300, \$250, \$350 per day.	Only critical activities can shorten the project.
The cheapest is D at \$250 per day, so crash D first.	C is cheaper (\$200) but is **not** on the critical path.
D has crash limit 2 days. Reducing D by 2: path A–B–D–F becomes $2+6+3+2=13$.	Path A–C–E–F is still 11, so A–B–D–F stays critical.
New completion time 13 days; extra cost $2\times\$250=\$500$.	New time = new critical-path length.
C has float 4 days, so shortening C leaves the critical path A–B–D–F unchanged at 15 days.	Crashing a non-critical activity does not help.

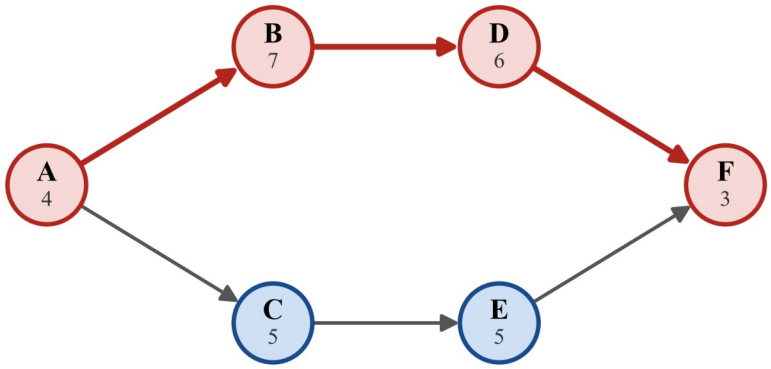


The reduced network confirms the critical path is still A–B–D–F, now 13 days.

Example 2 — scaffolded

A project is described by the table below (durations in weeks).

Activity	Duration	Immediate predecessor(s)	Crash cost per week	Crash limit (weeks)
A	4	—	\$500	2
B	7	A	\$200	5
C	5	A	\$250	3
D	6	B	\$450	3
E	5	C	\$280	3
F	3	D, E	\$300	1



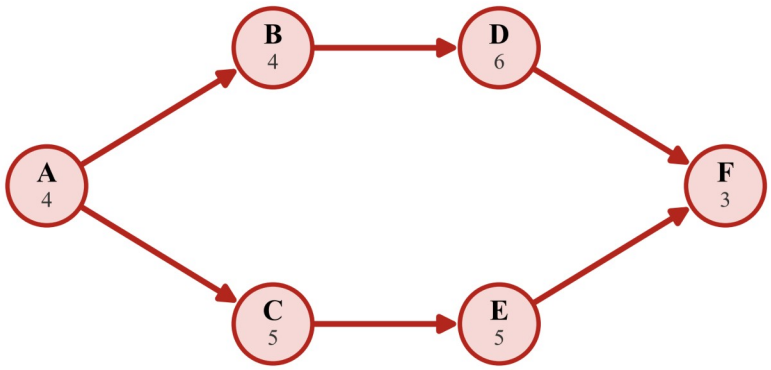
(a) Find the critical path and completion time.

(b) The cheapest critical activity is to be crashed. By how many weeks can it usefully be crashed, and why does crashing stop there?

(c) State the new completion time and the extra cost.

(d) Describe the cheapest way to reduce the completion time by one more week.

Working	Comment
Path A–B–D–F: $4+7+6+3=20$. Path A–C–E–F: $4+5+5+3=17$.	Two paths from start to finish.
Critical path A–B–D–F; completion time 20 weeks.	The longer path.
Cheapest critical activity: B at \$200 per week.	Critical costs are A \$500, B \$200, D \$450, F \$300.
Crashing B shortens A–B–D–F; it can only be reduced to 17 before A–C–E–F (also 17) becomes critical.	Useful reduction $= 20 - 17 = 3$ weeks (less than B's limit of 5).
Crash B by 3 weeks: B becomes 4, path A–B–D–$F = 4+4+6+3=17$.	A **new** critical path A–C–E–F has appeared.
New completion time 17 weeks; extra cost $3 \times \$200 = \$600$.	Both paths are now critical.
To save one more week, shorten **both** critical paths. Activity F (on both paths) costs \$300 per week — cheaper than crashing one activity on each path.	Crash F by 1 week: completion 16 weeks, further cost \$300.



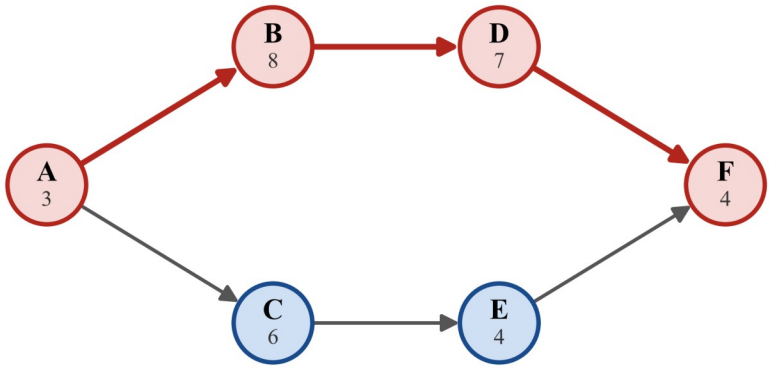
After crashing B, every activity lies on a critical path, so both routes through the network now take 17 weeks.

Example 3 — unscaffolded

A project has the following activities and crashing data (durations in days).

Activity	Duration	Immediate predecessor(s)	Crash cost per day	Crash limit (days)
A	3	—	\$500	2
B	8	A	\$150	4
C	6	A	\$180	3
D	7	B	\$200	3
E	4	C	\$220	2
F	4	D, E	\$450	2

Find the least cost of reducing the completion time to 17 days.



The two paths are A–B–D–$F = 3+8+7+4 = 22$ days and A–C–E–$F = 3+6+4+4 = 17$ days, so the critical path is A–B–D–F at 22 days and the project must be shortened by 5 days.

Only the critical activities A (\$500), B (\$150), D (\$200) and F (\$450) can help. The cheapest is B at \$150 per day. Crashing B by its full limit of 4 days reduces A–B–D–F to $3+4+7+4 = 18$ days, which is still critical (the other path is 17), at a cost of $4 \times \$150 = \$600$.

One more day is needed. The next cheapest critical activity is D at \$200 per day. Crashing D by 1 day reduces A–B–D–F to $3+4+6+4 = 17$ days, at a cost of $1 \times \$200 = \$200$. The path A–C–E–F is also 17 days, so both are now critical and the target is met.

The extra cost is $\$600 + \$200 = \$800$.

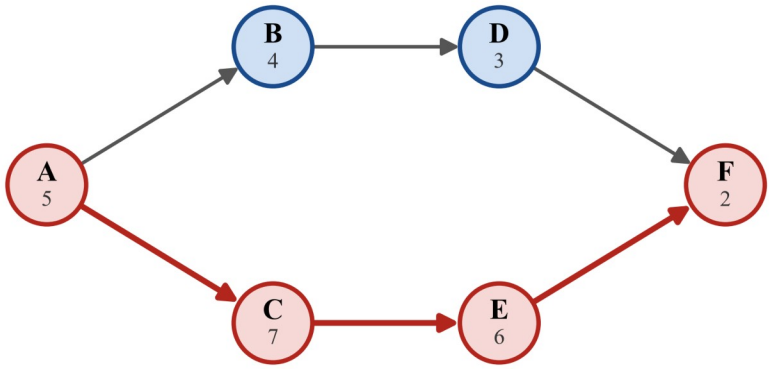
∴ the completion time can be reduced from 22 to 17 days for a least extra cost of \$800.

Example 4 — unscaffolded

A project is described by the table below (durations in days).

Activity	Duration	Immediate predecessor(s)	Crash cost per day	Crash limit (days)
A	5	—	\$600	1
B	4	A	\$250	2
C	7	A	\$400	3
D	3	B	\$300	2
E	6	C	\$350	2
F	2	D, E	\$450	1

A manager notices that activity B is the cheapest to crash and proposes crashing it to bring the project in sooner. Explain why this will not work, and find the greatest reduction in completion time that crashing activity E can achieve, together with its cost.



The paths are A–B–D–$F = 5+4+3+2 = 14$ days and A–C–E–$F = 5+7+6+2 = 20$ days, so the critical path is A–C–E–F at 20 days. Activity B lies on the path A–B–D–F, which has $20 - 14 = 6$ days of float. Because B is **not** on the critical path, shortening it does not shorten the critical path, so the completion time stays at 20 days no matter how much B is crashed — the money is wasted.

Activity E **is** on the critical path, with crash limit 2 days. Crashing E by 2 days reduces A–C–E–F to $5+7+4+2 = 18$ days; the other path is only 14 days, so A–C–E–F stays critical. The extra cost is $2 \times \$350 = \$700$.

$\therefore$ crashing B achieves nothing, whereas crashing E by 2 days reduces the completion time to 18 days at a cost of \$700.

Practice questions

Scaffolded

Q1. A project has the activities below (durations in days).

Activity	Duration	Immediate predecessor(s)	Crash cost per day	Crash limit (days)
A	3	—	\$600	1
B	5	A	\$400	3
C	2	A	—	—
D	4	B	\$300	2
E	4	C	—	—
F	3	D, E	\$500	1

(a) Find the two paths through the network and their lengths.

(b) State the critical path, the completion time, and which activities are non-critical.

(c) Which critical activity is the cheapest to crash?

(d) Crash that activity to its limit. State the new completion time and the extra cost.

Q2. A project is described by the table below (durations in weeks).

Activity	Duration	Immediate predecessor(s)	Crash cost per week	Crash limit (weeks)
A	4	—	\$500	1
B	6	A	\$250	3
C	3	A	\$300	2
D	5	B	\$400	2
E	3	C	—	—

Activity	Duration	Immediate predecessor(s)	Crash cost per week	Crash limit (weeks)
F	2	D, E	\$450	1

(a) State the critical path and completion time.

(b) Crash the cheapest critical activity to its limit; state the new completion time and cost.

(c) Show that crashing activity C instead would not reduce the completion time.

Q3. A project has the activities below (durations in days).

Activity	Duration	Immediate predecessor(s)	Crash cost per day	Crash limit (days)
A	2	—	—	—
B	8	A	\$200	5
C	5	A	—	—
D	4	B	\$350	3
E	4	C	—	—
F	3	D, E	\$400	1

(a) State the critical path and completion time.

(b) The cheapest critical activity is B. Crashing B, how many days can the completion time be reduced before a new critical path appears? State the extra cost and the new completion time.

(c) List the two critical paths after the crash.

Q4. A project is described by the table below (durations in days).

Activity	Duration	Immediate predecessor(s)	Crash cost per day	Crash limit (days)
A	3	—	\$500	2
B	6	A	\$150	3
C	4	A	\$250	2
D	6	B	\$250	3
E	4	C	—	—
F	3	D, E	\$400	1

(a) State the critical path and completion time.

(b) Crash activity B as far as it is useful, then continue crashing the critical path until a new critical path forms.

(c) State the smallest completion time reached in part (b) and the total extra cost.

Q5. A small project has activities A–E (durations in days).

Activity	Duration	Immediate predecessor(s)	Crash cost per day	Crash limit (days)
A	4	—	\$400	2
B	5	A	\$300	2
C	3	B	\$200	1
D	6	C	\$250	3
E	10	A	—	—

(a) State the critical path and completion time.

(b) Which activities can, when crashed, reduce the completion time?

(c) Find the least cost of reducing the completion time by 3 days, and state the new time.

Q6. Answer the following. (a) Explain why crashing a non-critical activity does not reduce the completion time of a project. (b) A critical activity G can be crashed by up to 4 days at \$500 per day, but the second-longest path is only 2 days shorter than the critical path. By how many days can crashing G alone reduce the completion time, and at what cost? (c) Two activities could be crashed for the same cost: one has float 0 and the other has float 3. Which should be crashed to shorten the project, and why?

Unscaffolded

Q7. A project has the activities below (durations in days).

Activity	Duration	Immediate predecessor(s)	Crash cost per day	Crash limit (days)
A	5	—	\$500	1
B	4	A	—	—
C	8	A	\$300	3
D	6	C	\$350	2
E	3	B	—	—
F	2	D, E	\$400	1

Find the critical path and completion time, then crash the cheapest critical activity to its limit and state the new completion time and the extra cost.

Q8. A project is described by the table below (durations in weeks).

Activity	Duration	Immediate predecessor(s)	Crash cost per week	Crash limit (weeks)
A	3	—	—	—
B	9	A	\$180	6
C	6	A	—	—
D	4	B	\$300	3
E	4	C	—	—
F	2	D, E	\$350	1

By crashing the cheapest critical activity, reduce the completion time as far as possible before a new critical path appears. State the new completion time and the extra cost, and list the critical paths after the crash.

Q9. A project has the activities below (durations in weeks).

Activity	Duration	Immediate predecessor(s)	Crash cost per week	Crash limit (weeks)
A	4	—	\$600	2
B	8	A	\$120	5
C	5	A	\$250	2
D	5	B	\$300	3
E	4	C	—	—
F	4	D, E	\$400	2

(a) State the critical path and completion time.

(b) Crash activity B as far as it is useful; state the completion time and cost so far.

(c) Find the cheapest way to reduce the completion time by one further week, and state the total extra cost.

Q10. A project has completion time set by the table below (durations in days). The client requires the project finished within 16 days and will charge a penalty of \$400 for each day the project runs beyond 16 days.

Activity	Duration	Immediate predecessor(s)	Crash cost per day	Crash limit (days)
A	3	—	\$400	1
B	6	A	\$180	3
C	4	A	\$250	2
D	5	B	\$300	2
E	4	C	—	—
F	4	D, E	\$350	1

Determine the completion time, then decide whether it is cheaper to crash the project to 16 days or to pay the penalty, justifying your answer.

Q11. Answer the following. (a) Explain what is meant by *crashing* a project. (b) State the two conditions that determine how far a single critical activity can usefully be crashed. (c) Explain why crashing a project cannot reduce the completion time indefinitely.

Q12. A project is described by the table below (durations in days).

Activity	Duration	Immediate predecessor(s)	Crash cost per day	Crash limit (days)
A	2	—	\$450	1
B	6	A	\$200	2
C	6	A	\$200	2
D	5	B	\$220	2
E	5	C	\$220	2
F	3	D, E	\$300	2

(a) Show that the network has two critical paths and state the completion time.

(b) Find the cheapest way to reduce the completion time by 2 days, and state the new time and cost.

Q13. A project has seven activities (durations in days).

Activity	Duration	Immediate predecessor(s)	Crash cost per day	Crash limit (days)
A	6	—	\$500	2
B	4	A	\$300	2
C	5	A	—	—
D	7	B	\$250	3
E	3	C	—	—
F	4	D	\$350	2
G	5	E, F	\$400	2

Find the critical path and completion time, then crash the cheapest critical activity to its limit and state the new completion time and the extra cost.

Q14. A project is described by the table below (durations in weeks).

Activity	Duration	Immediate predecessor(s)	Crash cost per week	Crash limit (weeks)
A	4	—	\$400	2
B	7	A	—	—

Activity	Duration	Immediate predecessor(s)	Crash cost per week	Crash limit (weeks)
C	6	A	\$300	2
D	5	B	—	—
E	8	C	\$220	4
F	3	D, E	\$350	1

Crash the cheapest critical activity until a new critical path appears. State the new completion time, the extra cost, and the two critical paths after the crash.

Q15. A project has the activities below (durations in days).

Activity	Duration	Immediate predecessor(s)	Crash cost per day	Crash limit (days)
A	3	—	\$500	2
B	7	A	\$300	3
C	4	A	\$250	2
D	8	B	\$200	4
E	5	C	\$220	2
F	3	D, E	\$450	2

Find the least cost of reducing the completion time to 15 days, showing which activities are crashed and by how much.

Q16. A project is described by the table below (durations in weeks). A client will pay a bonus of \$1500 if the project is completed in 17 weeks or less.

Activity	Duration	Immediate predecessor(s)	Crash cost per week	Crash limit (weeks)
A	4	—	\$600	2
B	9	A	\$400	5
C	6	A	—	—
D	4	B	—	—
E	4	C	—	—
F	3	D, E	\$500	1

Determine the completion time, the least cost of qualifying for the bonus, and whether it is worth crashing the project to earn it.

Q17. A project has the activities below (durations in days).

Activity	Duration	Immediate predecessor(s)	Crash cost per day	Crash limit (days)
A	2	—	\$600	1
B	7	A	\$250	2
C	7	A	\$250	2
D	4	B	\$200	2
E	4	C	\$200	2
F	3	D, E	\$350	1

(a) State the two critical paths and the completion time.

(b) Explain why reducing the completion time by one day requires crashing an activity on each of the two critical paths, unless a shared activity is crashed.

(c) Find the cheapest way to reduce the completion time by one day.

Q18. A project has a critical path of length 24 days and a second-longest path of length 19 days. (a) By crashing activities on the critical path only, what is the greatest number of days by which the completion time can be reduced before a new critical path appears? (b) To reduce the completion time further, which activities must then be crashed? (c) Explain why the completion time can never be reduced below the length of the longest chain of activities that cannot be crashed.

Q19. A project is described by the table below (durations in days).

Activity	Duration	Immediate predecessor(s)	Crash cost per day	Crash limit (days)
A	4	—	\$400	2
B	6	*A*	—	—
C	9	*A*	\$300	4
D	5	*B*	—	—
E	4	*C*	\$350	2
F	2	*D*, *E*	\$450	1

(a) Find the critical path and completion time.

(b) Crash the cheapest critical activity until a new critical path appears; state the new completion time and the extra cost.

Q20. A project has the activities below (durations in days).

Activity	Duration	Immediate predecessor(s)	Crash cost per day	Crash limit (days)
A	3	—	\$300	1
B	6	*A*	\$150	3
C	3	*A*	—	—
D	6	*B*	\$200	2
E	3	*C*	—	—
F	3	*D*, *E*	\$250	1

Find the shortest completion time achievable by crashing, and the total extra cost of reaching it. (Crash every useful activity to its limit.)

Answers

Q1. (a) A–B–D–$F = 15$, A–C–E–$F = 12$; (b) critical path A–B–D–F, 15 days; non-critical C, E; (c) D ($300/day); (d) crash D by 2: 13 days, extra cost $600. **Q2.** (a) A–B–D–F, 17 weeks; (b) crash B by 3: 14 weeks, cost $750; (c) C has float, so crashing it leaves the completion time at 17 weeks. **Q3.** (a) A–B–D–F, 17 days; (b) 3 days, cost $600, new time 14 days; (c) A–B–D–F and A–C–E–F, both 14 days. **Q4.** (a) A–B–D–F, 18 days; (b) crash B by 3 (15 days) then D by 1; (c) 14 days, total extra cost $700. **Q5.** (a) A–B–C–D, 18 days; (b) A, B, C, D (the critical activities); (c) crash C by 1 and D by 2: cost $700, new time 15 days. **Q6.** (a) it has float, so shortening it does not change the length of the critical path; (b) 2 days, cost $1000; (c) the float-0 (critical) activity, since only critical activities shorten the project. **Q7.** A–C–D–F, 21 days; crash C by 3: new time 18 days, extra cost $900. **Q8.** Crash B by 3 weeks: new time 15 weeks, extra cost $540; critical paths A–B–D–F and A–C–E–F, both 15 weeks. **Q9.** (a) A–B–D–F, 21 weeks; (b) crash B by 4: 17 weeks, cost $480; (c) crash B by 1 more ($120) and C by 1 ($250), one on each critical path: 16 weeks, total $850. **Q10.** Completion time 18 days; crashing to 16 days costs $2 \times \$180 = \$360$ (crash B by 2), which is cheaper than the penalty $2 \times \$400 = \$800$, so crash the project. **Q11.** (a) shortening one or more activities, at a cost, to reduce the overall completion time; (b) the activity's crash limit, and the point at which a second path becomes critical; (c) every activity has a crash limit (and some cannot be crashed), so a shortest achievable time exists. **Q12.** (a) A–B–D–$F = A$–C–E–$F = 16$ days; (b) crash F by 2 (on both paths): 14 days, cost $600. **Q13.** A–B–D–F–G, 26 days; crash D by 3: new time 23 days, extra cost $750. **Q14.** Crash E by 2 weeks: new time 19 weeks, extra cost $440; critical paths A–C–E–F and A–B–D–F, both 19 weeks. **Q15.** Crash D by 4 ($800) and B by 2 ($600): completion 15 days, least cost $1400. **Q16.** Completion time 20 weeks; crash B by 3 to reach 17 weeks at cost $1200. Since $1200 < $1500, the bonus is worth earning (net gain $300). **Q17.** (a) A–B–D–F and A–C–E–F, both 16 days; (b) both paths must be shortened together, or a shared activity crashed; (c) crash F by 1 day ($350), cheaper than crashing one activity on each path ($\$200 + \$200 = \$400$); new time 15 days. **Q18.** (a) 5 days; (b) activities common to all critical paths, or one activity on each critical path; (c) those activities cannot be shortened, so their total length is a fixed lower bound. **Q19.** (a) A–C–E–F, 19 days; (b) crash C by 2: new time 17 days, extra cost $600. **Q20.** Crash B by 3, D by 2, A by 1 and F by 1: shortest time 11 days, total extra cost $1400.

Fully-worked solutions

Q1. The paths are A–B–D–$F = 3 + 5 + 4 + 3 = 15$ and A–C–E–$F = 3 + 2 + 4 + 3 = 12$, so the critical path is A–B–D–F with completion time 15 days; C and E (float 3 each) are non-critical. The critical activities that can be crashed are A ($600), B ($400), D ($300) and F ($500), so D is the cheapest. D has crash limit 2; crashing D by 2 days gives A–B–D–$F = 3 + 5 + 2 + 3 = 13$ days, while A–C–E–F stays at 12, so the new completion time is 13 days for an extra cost of $2 \times \$300 = \$600$.

Q2. The paths are A–B–D–$F = 4 + 6 + 5 + 2 = 17$ and A–C–E–$F = 4 + 3 + 3 + 2 = 12$, so the critical path is A–B–D–F, completion time 17 weeks. The cheapest critical activity is B at $250 per week (others: A $500, D $400, F $450). Crashing B by its limit of 3 weeks gives A–B–D–$F = 4 + 3 + 5 + 2 = 14$ weeks; the other path is only 12, so A–B–D–F stays critical. New completion time 14 weeks, extra cost $3 \times \$250 = \$750$. Activity C lies on A–C–E–F, which has $17 - 12 = 5$ weeks of float, so crashing C does not shorten the critical path and the completion time remains 17 weeks.

Q3. The paths are A–B–D–$F = 2 + 8 + 4 + 3 = 17$ and A–C–E–$F = 2 + 5 + 4 + 3 = 14$, so the critical path is A–B–D–F, completion time 17 days. Crashing B shortens this path, but only until A–C–E–F (length 14) becomes critical, that is by $17 - 14 = 3$ days (less than B's limit of 5). Crashing B by 3 days costs $3 \times \$200 = \$600$ and gives a new completion time of 14 days. Both A–B–D–$F = 2 + 5 + 4 + 3 = 14$ and A–C–E–$F = 14$ are now critical.

Q4. The paths are A–B–D–$F = 3 + 6 + 6 + 3 = 18$ and A–C–E–$F = 3 + 4 + 4 + 3 = 14$, so the critical path is A–B–D–F, completion time 18 days. The cheapest critical activity is B at $150 per day. Crashing B by its limit of 3 days gives A–B–D–$F = 3 + 3 + 6 + 3 = 15$ days (cost $3 \times \$150 = \$450$); the other path is 14, so the critical path is still A–B–D–F. The next cheapest critical activity is D ($250). Crashing D by 1 day gives A–B–D–$F = 3 + 3 + 5 + 3 = 14$ days (cost $250), and now A–C–E–F is also 14, so a new critical path has formed. The smallest completion time is 14 days, for a total extra cost of $\$450 + \$250 = \$700$.

Q5. The paths are A–B–C–$D = 4+5+3+6 = 18$ and A–$E = 4+10 = 14$, so the critical path is A–B–C–D, completion time 18 days. Only the critical activities A, B, C, D can reduce the completion time; E has float and cannot help. To reduce by 3 days at least cost, crash the cheapest critical activities first: C at \$200 per day (limit 1) then D at \$250 per day. Crash C by 1 day and D by 2 days, reducing the path to $4+5+2+4 = 15$ days (the other path is still 14). The least cost is $1 \times \$200 + 2 \times \$250 = \$700$, with a new completion time of 15 days.

Q6. (a) The completion time equals the length of the critical path. A non-critical activity has float, so it is not on the critical path; reducing its duration leaves the critical path, and therefore the completion time, unchanged. (b) Crashing G can shorten the critical path only until the second-longest path becomes critical, that is by 2 days, even though its crash limit is 4. The cost is $2 \times \$500 = \$1000$. (c) The activity with float 0 is on the critical path, so crashing it shortens the project; the activity with float 3 is non-critical and crashing it would not help. Crash the float-0 activity.

Q7. The paths are A–B–E–$F = 5+4+3+2 = 14$ and A–C–D–$F = 5+8+6+2 = 21$, so the critical path is A–C–D–F, completion time 21 days. The critical activities that can be crashed are A (\$500), C (\$300), D (\$350) and F (\$400), so C is the cheapest. Crashing C by its limit of 3 days gives A–C–D–$F = 5+5+6+2 = 18$ days; the other path is only 14, so A–C–D–F stays critical. The new completion time is 18 days for an extra cost of $3 \times \$300 = \$900$.

Q8. The paths are A–B–D–$F = 3+9+4+2 = 18$ and A–C–E–$F = 3+6+4+2 = 15$, so the critical path is A–B–D–F, completion time 18 weeks. The crashable critical activities are B (\$180), D (\$300) and F (\$350) — A cannot be crashed — so crash B. Crashing B shortens the path only until A–C–E–F (length 15) becomes critical, that is by $18 - 15 = 3$ weeks (less than B's limit of 6). The extra cost is $3 \times \$180 = \$540$ and the new completion time is 15 weeks. Now A–B–D–$F = 3+6+4+2 = 15$ and A–C–E–$F = 15$ are both critical.

Q9. (a) The paths are A–B–D–$F = 4+8+5+4 = 21$ and A–C–E–$F = 4+5+4+4 = 17$, so the critical path is A–B–D–F, completion time 21 weeks. (b) B is the cheapest critical activity (\$120 per week). Crashing B is useful only until A–C–E–F (length 17) becomes critical, that is by $21 - 17 = 4$ weeks (within B's limit of 5). This gives a completion time of 17 weeks at a cost of $4 \times \$120 = \$480$. (c) Both paths are now critical, so a further reduction must shorten both: either crash an activity common to both paths, or crash one activity on each path. The common activities are A (\$600) and F (\$400), so the cheapest shared option costs \$400. For one activity on each path, note that B has been crashed by only 4 of its limit of 5, so it still has 1 week available at \$120; on A–C–E–F the only crashable activity is C (\$250). That pair costs $\$120 + \$250 = \$370$, which is cheaper than \$400. Crash B by 1 more week and C by 1 week: A–B–D–$F = 4+3+5+4 = 16$ and A–C–E–$F = 4+4+4+4 = 16$ weeks. The total extra cost is $\$480 + \$370 = \$850$.

Q10. The paths are A–B–D–$F = 3+6+5+4 = 18$ and A–C–E–$F = 3+4+4+4 = 15$, so the critical path is A–B–D–F, completion time 18 days. To finish in 16 days the project must be shortened by 2 days. The cheapest critical activity is B at \$180 per day, with limit 3; crashing B by 2 days gives A–B–D–$F = 3+4+5+4 = 16$ days (the other path is 15), at a cost of $2 \times \$180 = \$360$. Paying the penalty instead would cost $2 \times \$400 = \$800$. Since $\$360 < \$800$, it is cheaper to crash the project to 16 days.

Q11. (a) Crashing a project means deliberately shortening the durations of one or more activities, at a stated cost per unit time, so that the overall completion time is reduced. (b) A single critical activity can usefully be crashed until either (i) it reaches its crash limit, or (ii) a second path becomes critical (the float of the next-longest path runs out) — whichever occurs first. (c) Every activity has a crash limit, and some activities cannot be crashed at all, so there is a shortest possible completion time. Once that is reached, no further reduction is possible at any cost.

Q12. (a) The paths are A–B–D–$F = 2+6+5+3 = 16$ and A–C–E–$F = 2+6+5+3 = 16$, so both are critical and the completion time is 16 days. (b) Reducing the completion time requires shortening both critical paths. Activity F lies on both paths, so crashing F by 1 day shortens both at once for \$300 — cheaper than crashing one activity on each path (at least $\$200 + \$200 = \$400$ per day). Crashing F by its limit of 2 days gives a completion time of $2+6+5+1 = 14$ days for an extra cost of $2 \times \$300 = \$600$.

Q13. The paths are A–B–D–F–$G = 6+4+7+4+5 = 26$ and A–C–E–$G = 6+5+3+5 = 19$, so the critical path is A–B–D–F–G, completion time 26 days. The crashable critical activities are A (\$500), B (\$300), D (\$250), F (\$350) and G (\$400), so D is the cheapest. Crashing D by its limit of 3 days gives A–B–D–F–$G = 6+4+4+4+5 = 23$

days; the other path is only 19, so the critical path is unchanged. The new completion time is 23 days for an extra cost of $3 \times \$250 = \$750$.

Q14. The paths are $A–B–D–F = 4+7+5+3 = 19$ and $A–C–E–F = 4+6+8+3 = 21$, so the critical path is $A–C–E–F$, completion time 21 weeks. The crashable critical activities are A (\$400), C (\$300), E (\$220) and F (\$350), so E is the cheapest. Crashing E shortens the path only until $A–B–D–F$ (length 19) becomes critical, that is by $21 - 19 = 2$ weeks (within E's limit of 4). The extra cost is $2 \times \$220 = \$440$ and the new completion time is 19 weeks. Now $A–C–E–F = 4+6+6+3 = 19$ and $A–B–D–F = 19$ are both critical.

Q15. The paths are $A–B–D–F = 3+7+8+3 = 21$ and $A–C–E–F = 3+4+5+3 = 15$, so the critical path is $A–B–D–F$, completion time 21 days; the target requires a reduction of 6 days. The cheapest critical activity is D at \$200 per day (limit 4). Crashing D by 4 days gives $A–B–D–F = 3+7+4+3 = 17$ days (cost $4 \times \$200 = \$800$); the other path is 15, so $A–B–D–F$ stays critical. The next cheapest critical activity is B (\$300). Crashing B by 2 days gives $A–B–D–F = 3+5+4+3 = 15$ days (cost $2 \times \$300 = \$600$), and $A–C–E–F$ is also 15, so the target is reached. The least cost is $\$800 + \$600 = \$1400$.

Q16. The paths are $A–B–D–F = 4+9+4+3 = 20$ and $A–C–E–F = 4+6+4+3 = 17$, so the critical path is $A–B–D–F$, completion time 20 weeks. To qualify for the bonus the project must be shortened by 3 weeks. The cheapest crashable critical activity is B at \$400 per week (limit 5; the others are A \$600 and F \$500). Crashing B by 3 weeks gives $A–B–D–F = 4+6+4+3 = 17$ weeks — exactly meeting the bonus — at a cost of $3 \times \$400 = \$1200$. Since the bonus is \$1500 and $\$1200 < \$1500$, crashing the project is worthwhile, giving a net gain of $\$1500 - \$1200 = \$300$.

Q17. (a) The paths are $A–B–D–F = 2+7+4+3 = 16$ and $A–C–E–F = 2+7+4+3 = 16$, so both are critical and the completion time is 16 days. (b) The completion time is the length of the longest path, and here two paths share that length. Shortening only one of them leaves the other at 16 days, so the completion time would not change; both critical paths must be shortened, either by crashing an activity on each or by crashing one activity common to both. (c) The only activities on both paths are A (\$600) and F (\$350). Crashing F by 1 day shortens both paths for \$350, cheaper than crashing one activity on each path (the cheapest such pair is D and E at $\$200 + \$200 = \$400$). The new completion time is 15 days.

Q18. (a) Crashing activities on the current critical path shortens only that path, until the second-longest path becomes critical. That happens after a reduction of $24 - 19 = 5$ days. (b) After that, two paths are critical, so any further reduction must shorten **both**: crash an activity common to all critical paths, or crash one activity on each critical path. (c) An activity that cannot be crashed keeps its full duration whatever is spent. A chain of such activities therefore has a fixed total length that no crashing can reduce, so the completion time can never fall below the length of the longest such chain.

Q19. (a) The paths are $A–B–D–F = 4+6+5+2 = 17$ and $A–C–E–F = 4+9+4+2 = 19$, so the critical path is $A–C–E–F$, completion time 19 days. (b) The crashable critical activities are A (\$400), C (\$300), E (\$350) and F (\$450), so C is the cheapest. Crashing C shortens the path only until $A–B–D–F$ (length 17) becomes critical, that is by $19 - 17 = 2$ days (within C's limit of 4). The extra cost is $2 \times \$300 = \$600$ and the new completion time is 17 days.

Q20. The paths are $A–B–D–F = 3+6+6+3 = 18$ and $A–C–E–F = 3+3+3+3 = 12$, so the critical path is $A–B–D–F$, completion time 18 days. Crash the cheapest critical activities in turn. First B (\$150) by its limit of 3 days, giving 15 days; then D (\$200) by its limit of 2 days, giving 13 days — the other path is still 12. The remaining crashable activities A and F lie on **both** paths, so crashing them shortens the whole network: crash A by 1 day (to 12 days) and F by 1 day (to 11 days). Every crashable activity is now at its limit, so 11 days is the shortest achievable completion time. The total extra cost is

$$3 \times \$150 + 2 \times \$200 + 1 \times \$300 + 1 \times \$250 = 450 + 400 + 300 + 250 = \$1400.$$

Topic 4 Test A — Multiple-choice

*One bound reference and one approved technology (CAS calculator or software) are permitted; a scientific calculator may also be used. Reading time: 3 minutes. Writing time: 20 minutes. 12 multiple-choice questions, 12 marks. Choose the response that is **correct** for the question.*

Question 1

A graph has five vertices with degrees $2, 3, 3, 4$ and 4. The number of edges in the graph is

A. 8

B. 10

C. 12

D. 16

Question 2

A connected planar graph has 7 vertices and 10 edges. By Euler's formula, the number of faces (regions), including the outer region, is

A. 2

B. 3

C. 4

D. 5

Question 3

The adjacency matrix of a graph, with the vertices taken in the order A, B, C, D, is

$$\begin{bmatrix} 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 1 \\ 1 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}.$$

The degree of vertex C is

A. 2

B. 3

C. 4

D. 6

Question 4

A graph has the adjacency matrix

$$A=\begin{bmatrix}0 & 1 & 1 & 0\\ 1 & 0 & 1 & 1\\ 1 & 1 & 0 & 1\\ 0 & 1 & 1 & 0\end{bmatrix},$$

with the vertices taken in the order A, B, C, D. The number of walks of length 2 from A to D is

A. 1

B. 2

C. 3

D. 4

Question 5

A connected graph has five vertices with degrees $2, 3, 3, 4$ and 4. Which one of the following statements is true?

A. The graph has an Eulerian circuit.

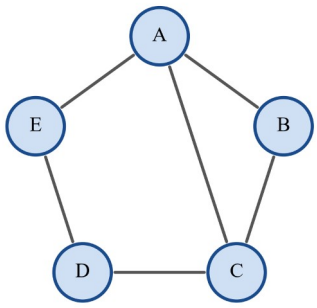
B. The graph must be a complete graph.

C. The graph has neither an Eulerian trail nor an Eulerian circuit.

D. The graph has an Eulerian trail but not an Eulerian circuit.

Question 6

For the graph shown below, which one of the following is a Hamiltonian cycle?



A. $A-B-D-C-E-A$

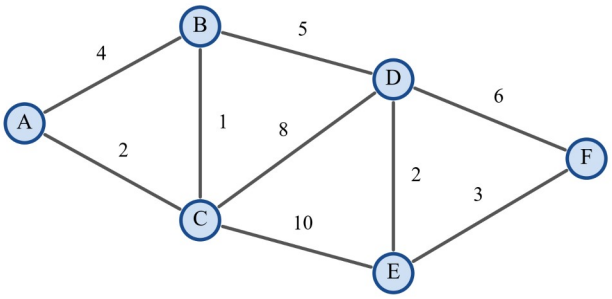
B. $A-C-D-B-E-A$

C. $A-C-B-D-E-A$

D. $A-B-C-D-E-A$

Question 7

For the weighted graph shown below, the length of the shortest path from A to F is



A. 11

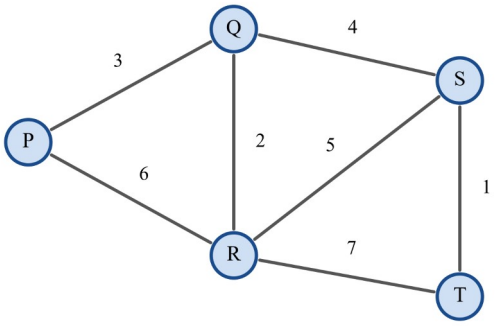
B. 12

C. 13

D. 15

Question 8

The network below shows the possible connections between five sites, with the number on each edge its weight. The total weight of the minimum spanning tree of this network is



A. 10

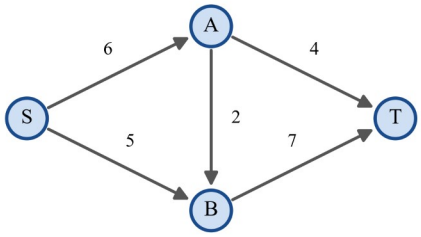
B. 12

C. 15

D. 16

Question 9

For the directed network shown below, the number on each arc is the capacity of that arc. The maximum flow from the source S to the sink T is



A. 10

B. 11

C. 12

D. 13

Question 10

The table shows the cost, in dollars, for each of three workers X, Y, Z to complete each of three tasks. Each worker is assigned exactly one task. The minimum total cost of completing all three tasks is

Worker	Task 1	Task 2	Task 3
X	9	2	7
Y	6	4	3
Z	5	8	1

A. \$9

B. \$10

C. \$12

D. \$14

Question 11

The precedence table for a project is shown below. The minimum completion time for the project, in hours, is

Activity	Duration (hours)	Immediate predecessor(s)
A	6	—
B	3	—
C	5	A
D	7	B
E	4	C, D

A. 13

B. 14

C. 15

D. 17

Question 12

A project has critical path $A - C - E$ of length 15 hours; its only other path, $B - D - E$, has length 14 hours. Activities on the critical path can be shortened (crashed) at these costs: A at \$100 per hour, C at \$150 per hour and E at \$200 per hour. The minimum cost of reducing the completion time of the project by one hour is

A. \$0

B. \$100

C. \$150

D. \$200

End of Topic 4 Test A

Test A — answers and solutions

Q	1	2	3	4	5	6	7	8	9	10	11	12
Ans	A	D	C	B	D	D	C	A	B	A	C	B

Q1. A — the sum of the degrees is $2+3+3+4+4=16$, and this equals twice the number of edges, so the number of edges is $16\div 2=8$.

Q2. D — Euler's formula for a connected planar graph is $v-e+f=2$, so $f=2-v+e=2-7+10=5$.

Q3. C — the entry on the leading diagonal for C is 1, so C carries a loop. The row sum for C is $1+1+1+0=3$; a loop adds 2 to the degree while written as 1, so $\text{degree}=(\text{row sum})+(\text{loops})=3+1=4$.

Q4. B — the number of walks of length 2 from A to D is the (A, D) entry of A^2. Here $A^2=\begin{bmatrix}2&1&1&2\\1&3&2&1\\1&2&3&1\\2&1&1&2\end{bmatrix}$, and the (A, D) entry is 2 (the walks $A-B-D$ and $A-C-D$).

Q5. D — the graph has exactly two vertices of odd degree (the two of degree 3). A connected graph has an Eulerian trail but not an Eulerian circuit precisely when it has exactly two odd-degree vertices.

Q6. D — the available edges are AB, BC, CD, DE, EA and AC. Only $A-B-C-D-E-A$ uses edges that all exist; each of the other options requires the edge $B-D$, which is not in the graph.

Q7. C — applying Dijkstra's algorithm from A gives shortest distances $A\,0, C\,2, B\,3, D\,8, E\,10, F\,13$. The shortest path is $A-C-B-D-E-F$, of length $2+1+5+2+3=13$.

Q8. A — the minimum spanning tree uses edges $S-T\,(1), Q-R\,(2), P-Q\,(3), Q-S\,(4)$, with total weight $1+2+3+4=10$.

Q9. B — the cut separating S from the rest has capacity $6+5=11$, and a flow of 11 is achievable ($S-A-T$ carries 4, $S-B-T$ carries 5, $S-A-B-T$ carries 2), so the maximum flow equals the minimum cut, 11.

Q10. A — assigning X to Task 2 ($2), Y to Task 1 ($6) and Z to Task 3 ($1) gives the least total, $2+6+1=9$; no other allocation is cheaper.

Q11. C — the earliest finish times are $A\,6, B\,3, C\,11, D\,10, E\,15$. Activity E cannot start until both C (finishes at 11) and D (finishes at 10) are complete, so the project finishes at $11+4=15$ hours (critical path $A-C-E$).

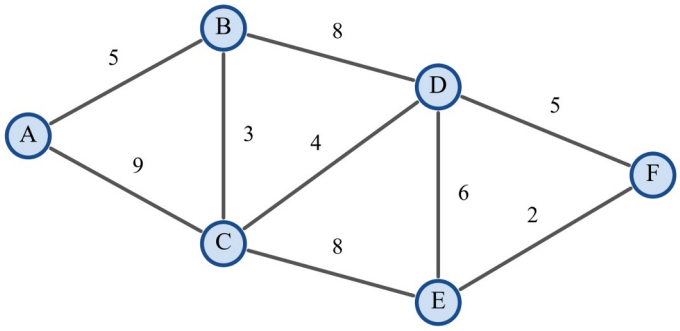
Q12. B — only crashing an activity on the critical path $A-C-E$ shortens the project. The cheapest such activity is A at $100 per hour; crashing A by one hour reduces $A-C-E$ to 14 hours (equal to $B-D-E$), so a one-hour reduction is achieved at a cost of $100.

Topic 4 Test B — Short-answer

One bound reference and one approved technology (CAS calculator or software) are permitted; a scientific calculator may also be used. Reading time: 5 minutes. Writing time: 40 minutes. Total 40 marks. In all questions where a numerical answer is required, an exact value must be given unless otherwise specified. Where more than one mark is available, appropriate working must be shown.

Question 1 (9 marks)

A national park has six lookout points A, B, C, D, E and F joined by walking tracks. In the weighted graph below, each number is the length of that track, in kilometres.



a. State the degree of lookout D and the total number of tracks in the network. 2 marks

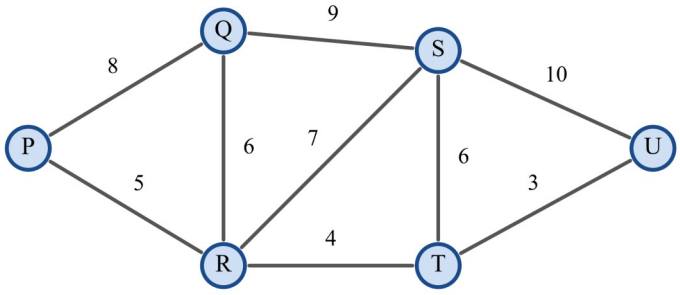
b. A ranger wants to walk along every track exactly once. Using the degrees of the vertices, explain why this is possible, and state the two lookouts at which such a walk must begin and end. 2 marks

c. Find the shortest path from A to F, and state its length. 4 marks

d. Write down the length of the shortest path from A to E. 1 mark

Question 2 (7 marks)

A water authority must lay pipes to connect six new estates P, Q, R, S, T and U. The network below shows the possible pipe connections; each number is the cost of that pipe, in thousands of dollars.



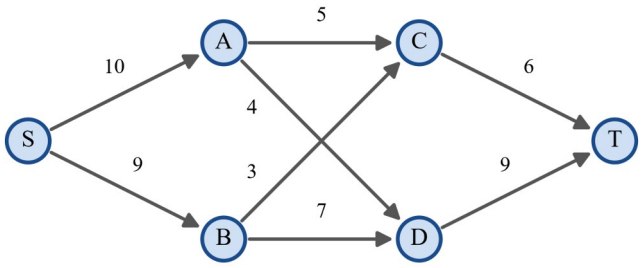
a. The authority will connect all six estates as cheaply as possible using a minimum spanning tree. How many pipes will the minimum spanning tree contain? 1 mark

b. Determine the minimum spanning tree. List the pipes it uses and state the minimum total cost. 4 marks

c. The pipe between R and T cannot be built. Find the new minimum total cost of connecting all six estates. 2 marks

Question 3 (8 marks)

The directed network below shows a data network carrying information from server S to client T. The number on each arc is the capacity of that arc, in gigabytes per second (GB/s).



a. Find the capacity of the cut that separates $\{S, A, B\}$ from $\{C, D, T\}$. 2 marks

b. Explain why the maximum flow from S to T cannot exceed 15 GB/s. 1 mark

c. Determine the maximum flow from S to T, in GB/s. 3 marks

d. State the minimum cut and explain how it confirms your answer to part **c.** 2 marks

Question 4 (8 marks)

A café must assign four baristas $W1, W2, W3$ and $W4$ to four stations. The table shows the time, in minutes, each barista takes to run each station. Each barista is assigned to exactly one station.

Barista	Station 1	Station 2	Station 3	Station 4
$W1$	12	9	15	11
$W2$	8	13	10	14
$W3$	14	11	9	12
$W4$	10	8	13	15

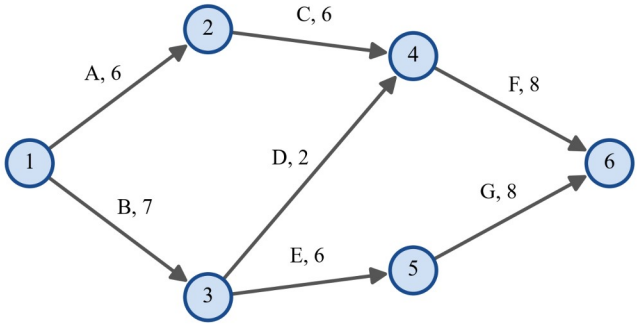
a. The manager wants to minimise the total time. Explain why it may not be possible simply to give each barista the station at which they are individually fastest, and describe the first step of the Hungarian algorithm. 2 marks

b. Determine the allocation of baristas to stations that minimises the total time, and state that minimum total time. 4 marks

c. Barista $W2$ is replaced by a trainee who takes 3 minutes longer than $W2$ at every station. Explain why the optimal allocation does not change, and state the new minimum total time. 2 marks

Question 5 (8 marks)

The activity network below shows the activities needed to set up a large outdoor event. Each arc is an activity, labelled with its letter and duration in hours; the numbered circles are events.



a. Determine the minimum completion time for the set-up and write down the critical path. 3 marks

b. Find the float (slack) of activity C and of activity D. 2 marks

Each activity except D can be crashed (shortened) by at most 2 hours. The cost of crashing, per hour, is shown below.

Activity	A	B	C	E	F	G
Cost per hour ($)	40	70	55	60	80	50

c. The organisers want to reduce the completion time by 2 hours. Determine which activities should be crashed, the new completion time, and the minimum extra cost. 3 marks

End of Topic 4 Test B

Test B — answers and solutions

Q1. (a) The degrees are $A\,2, B\,3, C\,4, D\,4, E\,3, F\,2$, so the degree of D is 4. The sum of the degrees is $2+3+4+4+3+2=18$, which is twice the number of tracks, so there are $18\div 2=9$ tracks.

(b) A walk along every track exactly once (an Eulerian trail) exists in a connected graph precisely when the number of odd-degree vertices is 0 or 2. Here exactly two vertices are of odd degree, B and E (each of degree 3), so such a walk is possible; it must begin at one of B, E and end at the other.

(c) Applying Dijkstra's algorithm from A:

Step	A	B	C	D	E	F
start	0	5	9	—	—	—
fix $B=5$			8	13	—	—
fix $C=8$				12	16	—
fix $D=12$					16	17
fix $E=16$						17

The shortest distance from A to F is 17 km, along the path $A-B-C-D-F$ ($5+3+4+5=17$).

(d) The shortest distance from A to E is 16 km (path $A-B-C-E$, $5+3+8=16$).

Q2. (a) A spanning tree of a network with 6 vertices has $6-1=5$ edges, so the minimum spanning tree contains 5 pipes.

(b) Using Prim's algorithm, start at P and repeatedly add the cheapest pipe joining an estate already connected to one that is not: $P-R(5), R-T(4), T-U(3), Q-R(6), S-T(6)$. These five pipes connect all six estates (Q is joined by $Q-R$ at 6 rather than $P-Q$ at 8, and S by $S-T$ at 6 rather than $R-S$ at 7 or $Q-S$ at 9). The minimum total cost is

$$5+4+3+6+6=24,$$

that is, \$24,000.

(c) Without the pipe $R-T$, Prim's algorithm from P selects $P-R(5), Q-R(6)$, and then the only remaining links from $\{P, Q, R\}$ are $R-S(7)$ and $Q-S(9)$, so $R-S(7)$ is used; from there $S-T(6)$ and $T-U(3)$ complete the tree. The new minimum total cost is

$$5+6+7+6+3=27,$$

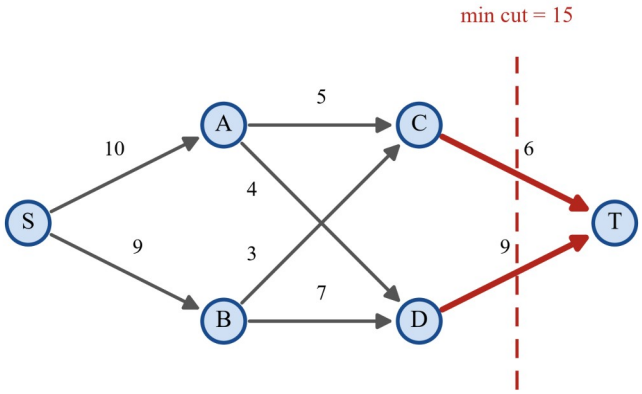
that is, \$27,000.

Q3. (a) The cut separating $\{S, A, B\}$ from $\{C, D, T\}$ cuts the forward arcs $A-C(5), A-D(4), B-C(3)$ and $B-D(7)$, so its capacity is $5+4+3+7=19$.

(b) Every unit of flow reaching T must travel along one of the only two arcs into T, namely $C-T$ and $D-T$. Their combined capacity is $6+9=15$, so the maximum flow cannot exceed 15 GB/s.

(c) A flow of 15 GB/s is achievable, for example: $S-A$ carries 9 (split as $A-C$ 5 and $A-D$ 4) and $S-B$ carries 6 (split as $B-C$ 1 and $B-D$ 5). Then $C-T$ carries $5+1=6$ and $D-T$ carries $4+5=9$, giving a total flow into T of $6+9=15$. Since the flow cannot exceed 15 (part **b**), the maximum flow is 15 GB/s.

(d) The minimum cut is $\{S, A, B, C, D\}\,|\,\{T\}$, cutting the arcs $C-T$ and $D-T$, with capacity $6+9=15$. By the max-flow/min-cut theorem the maximum flow equals the capacity of the minimum cut, and both are 15 GB/s, confirming the answer to part **c**.



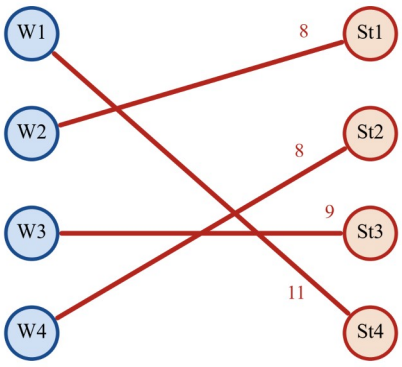
Q4. (a) The station at which each barista is individually fastest is: $W1 \rightarrow$ Station 2 (9), $W2 \rightarrow$ Station 1 (8), $W3 \rightarrow$ Station 3 (9), $W4 \rightarrow$ Station 2 (8). Both $W1$ and $W4$ are fastest at Station 2, but each station takes exactly one barista, so these individual choices clash and cannot all be used. The first step of the Hungarian algorithm is to subtract the smallest entry in each row from every entry in that row (row reduction).

(b) Row reduction (subtract $9, 8, 9, 8$) then column reduction (the only column without a zero is Station 4; subtract its smallest entry, 2) gives

$$\begin{bmatrix} 3 & 0 & 6 & 0 \\ 0 & 5 & 2 & 4 \\ 5 & 2 & 0 & 1 \\ 2 & 0 & 5 & 5 \end{bmatrix}.$$

The zeros allow one complete allocation with a zero in every row and column: $W1 \rightarrow$ Station 4, $W2 \rightarrow$ Station 1, $W3 \rightarrow$ Station 3, $W4 \rightarrow$ Station 2. The minimum total time is

$$11 + 8 + 9 + 8 = 36 \text{ minutes.}$$



(c) Adding 3 minutes to every entry of the $W2$ row adds 3 to the total time of *every* possible allocation, because each allocation uses exactly one entry from the $W2$ row. Adding the same constant to all totals does not change which allocation is smallest, so the optimal allocation is unchanged. The new minimum total time is $36 + 3 = 39$ minutes.

Q5. (a) A forward scan gives the earliest event times $E_1 = 0, E_2 = 6, E_3 = 7, E_4 = 12, E_5 = 13, E_6 = 21$, so the minimum completion time is 21 hours. Tracing the longest path back from event 6 gives the critical path $B - E - G$ ($7 + 6 + 8 = 21$).

(b) The full schedule (durations, earliest and latest start/finish, and float) is:

Activity	Dur	EST	EFT	LST	LFT	Float
A	6	0	6	1	7	1
B	7	0	7	0	7	0

Activity	Dur	EST	EFT	LST	LFT	Float
C	6	6	12	7	13	1
D	2	7	9	11	13	4
E	6	7	13	7	13	0
F	8	12	20	13	21	1
G	8	13	21	13	21	0

The float of activity C is 1 hour and the float of activity D is 4 hours.

(c) The critical path $B-E-G$ has length 21 hours; the next-longest path is $A-C-F$ at 20 hours (the third path, $B-D-F$, is only 17 hours). To finish in 19 hours, both $B-E-G$ and $A-C-F$ must be brought down to 19 hours.

- $B-E-G$ must lose 2 hours. The cheapest activity on it is G at \$50 per hour, so crash G by 2 hours: cost $2\times\$50=\$100$. Then $B-E-G=7+6+6=19$.
- $A-C-F$ must lose 1 hour. The cheapest activity on it is A at \$40 per hour, so crash A by 1 hour: cost $1\times\$40=\$40$. Then $A-C-F=5+6+8=19$.

Crash G by 2 hours and A by 1 hour. The new completion time is 19 hours and the minimum extra cost is $\$100+\$40=\$140$.