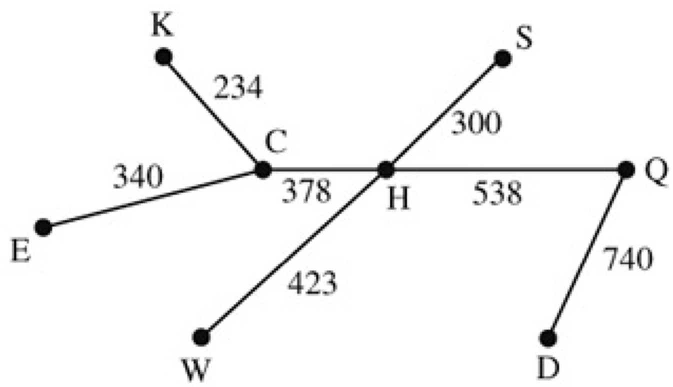
## Algorithms on graphs Mixed Exercise

**1 a i** Arcs are labelled with initial letters of the nodes.
CK add to tree
SH add to tree
CE add to tree
EK reject
CH add to tree
HW add to tree
CS reject
HQ add to tree
QS reject
QD add to tree
KS reject
DW reject
EW reject

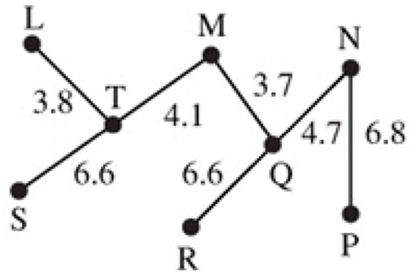
**ii** EC
CK
CH
HS
HW
HQ
QD

**b**



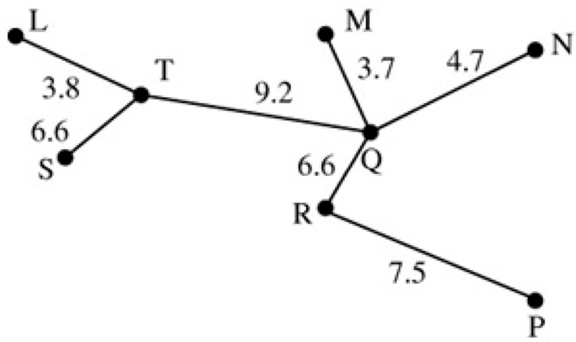

weight: 2953

**2 a i** LT
MT
MQ
NQ
ST } 
QR }
NP




**ii** MQ (3.7) add to tree
LT (3.8) add to tree
MT (4.1) add to tree
NQ (4.7) add to tree
MN (5.3) add to tree
ST (6.6) add to tree
QR (6.6) add to tree
NP (6.8) add to tree
reject remaining arcs

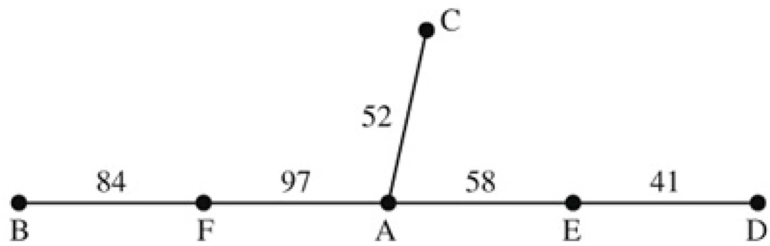
**b** Start off the tree with QT and PR then apply Kruskal's algorithm.
Prim's algorithm requires the 'growing' tree to be connected at all times. When using Kruskal's algorithm the tree can be built from non-connected sub-trees.




**3 a** Arcs in order:
AC
AE
DE
AF
BF

| | ↓1 A | ↓6 B | ↓2 C | ↓4 D | ↓3 E | ↓5 F |
|---|---|---|---|---|---|---|
| A | - | 124 | 52 | 87 | 58 | 97 |
| B | 124 | - | 114 | 111 | 115 | (84) |
| C | (52) | 114 | - | 67 | 103 | 98 |
| D | 87 | 111 | 67 | - | (41) | 117 |
| E | (58) | 115 | 103 | 41 | - | 121 |
| F | (97) | 84 | 98 | 117 | 121 | - |

**b**




Length 332 mm

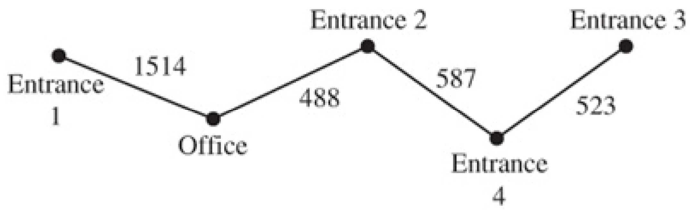
**3 c** $0.02 \times \left(\frac{240}{80}\right)^3 = 0.54$ seconds

**d** The time required is not directly proportional to $n^3$ but this is used as an approximation.

**4 a** Arcs in order:
Entrance 2 – Office
Entrance 2 – Entrance 4
Entrance 4 – Entrance 3
Office – Entrance 1

| | ↓2 | ↓5 | ↓1 | ↓4 | ↓3 |
|---|---|---|---|---|---|
| | Office | Entrance 1 | Entrance 2 | Entrance 3 | Entrance 4 |
| Office | - | 1514 | 488 | 980 | 945 |
| Entrance 1 | 1514 | - | 1724 | 2446 | 2125 |
| Entrance 2 | 488 | 1724 | - | 884 | 587 |
| Entrance 3 | 980 | 2446 | 884 | - | 523 |
| Entrance 4 | 945 | 2125 | 587 | 523 | - |

**b**



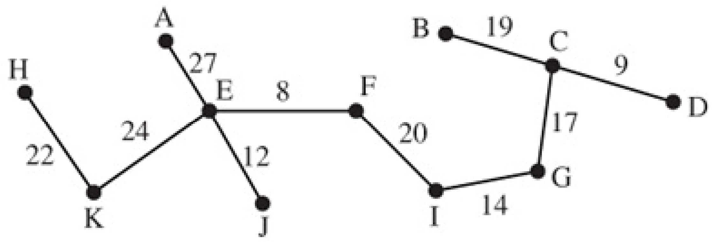

Length: 3112 m

**5 a**

| | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 29 | 27 | 30 | 19 | 9 | 26 | 17 | 18 | 8 | 12 | 24 | 20 | 14 | 14 | 22 | 26 |
| 8 | 29 | 27 | 30 | 19 | 9 | 26 | 17 | 18 | 12 | 24 | 20 | 14 | 14 | 22 | 26 |
| 8 | 9 | 17 | 12 | 14 | 14 | 18 | 29 | 27 | 30 | 19 | 26 | 24 | 20 | 22 | 26 |
| 8 | 9 | 12 | 17 | 14 | 14 | 18 | 19 | 24 | 20 | 22 | 26 | 29 | 27 | 30 | 26 |
| 8 | 9 | 12 | 14 | 17 | 14 | 18 | 19 | 20 | 24 | 22 | 26 | 29 | 27 | 26 | 30 |
| 8 | 9 | 12 | 14 | 14 | 17 | 18 | 19 | 20 | 22 | 24 | 26 | 26 | 27 | 29 | 30 |
| EF | CD | EJ | FJ | GI | CG | DG | BC | FI | HK | EK | CF | JK | AE | AB | AH |

**b** Order arcs into ascending order of weight and select the arc of least weight to start the tree: *EF*
Consider the next arc of least weight, if it would form a cycle with the arcs already selected, reject it. Continue to select an arc of least weight until all vertices are connected to give a minimum spanning tree.

**c**



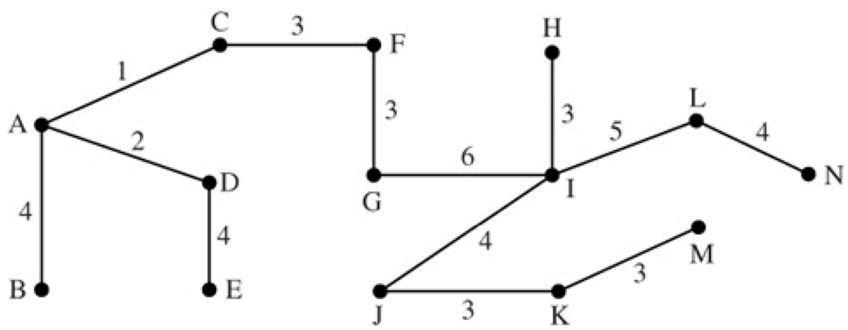

weight: 172

**5 d** $e = v - 1$

**6 a** Order of arcs

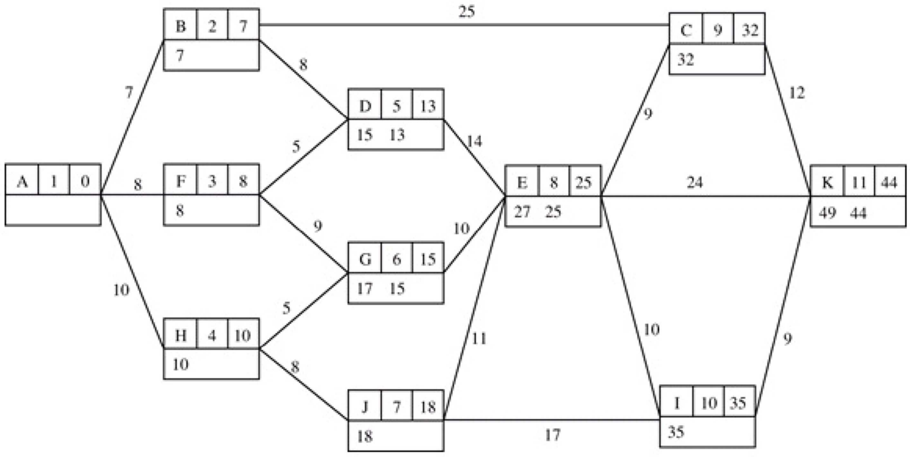
AC (1) add to tree
AD (2) add to tree
CD (2) reject
CF (3) add to tree
FG (3) add to tree
HI (3) add to tree
KM (3) add to tree
JK (3) add to tree
AB (4) add to tree
DE (4) add to tree
IJ (4) add to tree
LN (4) add to tree
DG (5) reject
BE (5) reject
IL (5) add to tree
MN (5) reject
EG (6) reject
GI (6) add to tree
IM (6)
FH (7)
HL (7)
EJ (7)
} reject remaining arcs

weight = 45 so 4500 m needed




**b** Remove FG (7) and replace with DG (5) weight = 47 so 4700 m

**7**




**a** **i** Possible paths are A – H – G – E – I – K
and A – H – J – I – K
and A – B – C – K

} Any one accepted

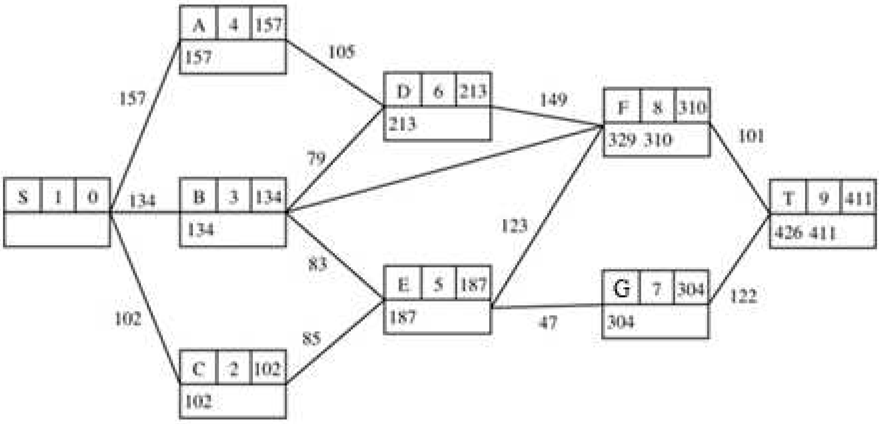
**ii**

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| $44-9=35$ | IK | | $44-9=35$ | IK | | $44-12=32$ | CK |
| $35-10=25$ | EI | | $35-17=18$ | JI | | $32-25=7$ | BC |
| $25-10=15$ | GE | or | $18-8=10$ | MJ | or | $7-7=0$ | AB |
| $15-5=10$ | HG | | $10-10=0$ | AH | | | |
| $10-10=0$ | AH | | | | | | |

**b** A – H – G – E – I – K and A – H – J – I – K and A – B – C – K

**c** The arcs could be roads.
The nodes could be junctions.
The number on each arc could be the distance in km.
The network, together with Dijkstra's algorithm, could be used to find the shortest route from A to K.

**8**

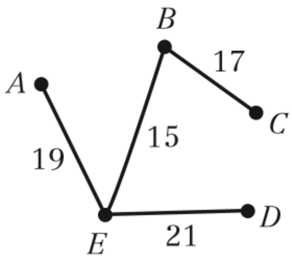



Order of vertex labelling: S C B A E D G F T

Route: S – C – E – F – T

$411 - 101 = 310$ FT
$310 - 123 = 187$ EF
$187 - 85 = 102$ CE
$102 - 102 = 0$ SC

**9 a i, ii**




Total weight = 72

**b** Prim's algorithm grows a minimum spanning tree by adding one vertex at a time. The next vertex chosen to be added is always the shortest edge from the vertex already on the graph.
Kruskal's algorithm grows a minimum spanning tree by adding one edge at a time. The edge with the least weight is always the next to be added only if it does not create a cycle.

**c** Prim's algorithm may be quicker on a graph with a large number of arcs, such as a complete network, as Kruskal's algorithm would require arcs to be sorted by weight.

**10 a** 1st iteration

| | *P* | *Q* | *R* | *S* |
|---|---|---|---|---|
| *P* | – | 12 | ∞ | 16 |
| *Q* | 12 | – | 15 | 28 |
| *R* | ∞ | 15 | – | 10 |
| *S* | 16 | 20 | 10 | – |

| | *P* | *Q* | *R* | *S* |
|---|---|---|---|---|
| *P* | *P* | *Q* | *R* | *S* |
| *Q* | *P* | *Q* | *R* | *P* |
| *R* | *P* | *Q* | *R* | *S* |
| *S* | *P* | *Q* | *R* | *S* |

2nd iteration

| | *P* | *Q* | *R* | *S* |
|---|---|---|---|---|
| *P* | – | 12 | 27 | 16 |
| *Q* | 12 | – | 15 | 28 |
| *R* | 27 | 15 | – | 10 |
| *S* | 16 | 20 | 10 | – |

| | *P* | *Q* | *R* | *S* |
|---|---|---|---|---|
| *P* | *P* | *Q* | *Q* | *S* |
| *Q* | *P* | *Q* | *R* | *P* |
| *R* | *Q* | *Q* | *R* | *S* |
| *S* | *P* | *Q* | *R* | *S* |

3rd iteration

| | *P* | *Q* | *R* | *S* |
|---|---|---|---|---|
| *P* | – | 12 | 27 | 16 |
| *Q* | 12 | – | 15 | 28 |
| *R* | 27 | 15 | – | 10 |
| *S* | 16 | 20 | 10 | – |

| | *P* | *Q* | *R* | *S* |
|---|---|---|---|---|
| *P* | *P* | *Q* | *Q* | *S* |
| *Q* | *P* | *Q* | *R* | *R* |
| *R* | *Q* | *Q* | *R* | *S* |
| *S* | *P* | *Q* | *R* | *S* |

4th iteration

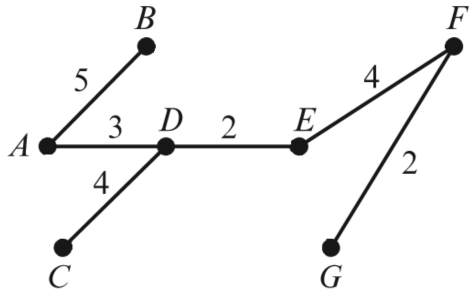
| | *P* | *Q* | *R* | *S* |
|---|---|---|---|---|
| *P* | – | 12 | 26 | 16 |
| *Q* | 12 | – | 15 | 25 |
| *R* | 26 | 15 | – | 10 |
| *S* | 16 | 20 | 10 | – |

| | *P* | *Q* | *R* | *S* |
|---|---|---|---|---|
| *P* | *P* | *Q* | *S* | *S* |
| *Q* | *P* | *Q* | *R* | *R* |
| *R* | *S* | *Q* | *R* | *S* |
| *S* | *P* | *Q* | *R* | *S* |

**b** From route table: shortest distance from *R* to *P* is via *S*.
Shortest distance from *R* to *S* is direct. Shortest distance from *S* to *P* is direct.
So shortest route from *R* to *P* is *RSP* = 26

**11 a** Prim's algorithm or Kruskal's algorithm

**b**




Total length is 20 miles

**c** Dijkstra's algorithm

**d** By inspection, shortest distance = 11 miles
Route: *ADEFG*

**e** Floyd's algorithm

**Challenge**

For a network of $n$ vertices, after the $r$th vertex has been selected you need to compare $(n-r)$ values of min($Y$) with $XY$, where $X$ is the most recently selected vertex. You then need to choose the smallest value of min($Y$), which requires a further $(n-r-1)$ comparisons. The number of comparisons at each step is $(n-1)+(n-r-1)=2n-2r-1$

So the total number of comparisons is:

$$\sum_{r=1}^{n-1}(2n-2r-1) = 2\sum_{r=1}^{n-1} n - 2\sum_{r=1}^{n-1} r - \sum_{r=1}^{n-1} 1$$
$$= 2n(n-1) - n(n-1) - (n-1)$$
$$= n^2 - 2n + 1$$

Which has order $n^2$