Topic: Minimum Spanning Tree **Name:** Solns

Trees

A tree is a graph which:

- is connected with a unique path between every pair of vertices,
- does not contain loops,
- does not contain multiple edges,
- does not contain any cycles, that is, it is acyclic,
- is unidirectional,
- every edge is a bridge.

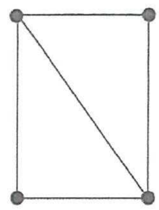
Summing up we can say that a tree is a connected graph with no cycles (acyclic)..

Weighted Graphs

A weighted graph or weighted network is a graph that has a numeric label associated with each edge called the **weight** of the edge. The edge weights can be distances, connection costs, flight times, cost of a fare, electrical capacity of a cable etc.

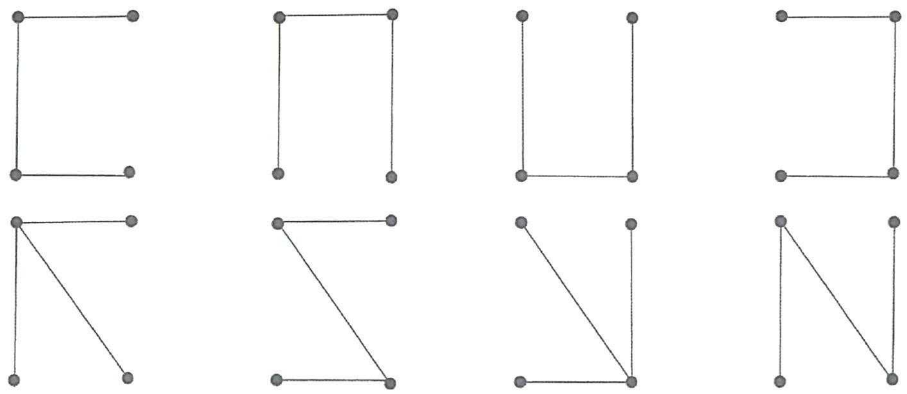
Spanning Trees

A spanning tree of a connected graph is a subgraph of that graph which contains all its vertices and is a tree. Consider the connected graph shown right:



The given connected graph has 4 vertices and 5 edges.

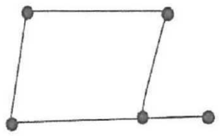
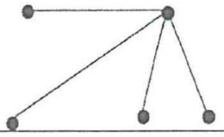
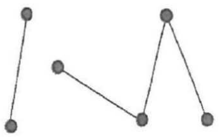
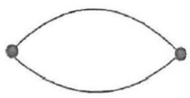
A spanning tree of this graph is a subgraph, which must contain all the vertices of the given graph, that is 4 vertices, and for the subgraph to be a tree it must have no cycles and one edge less than the number of vertices, that is it must have exactly three edges.
To construct spanning trees of the given graph we need to remove exactly two edges and make sure that the remaining three edges are connected and do not form a cycle.
Using a systematic approach, we first can construct all the trees without the diagonal and then all the trees with the diagonal. The result will then be the set of all possible spanning trees of the given graph.



The given connected graph has a total of 8 spanning trees.

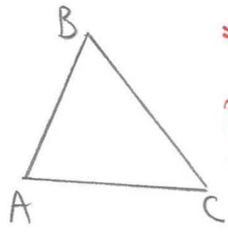
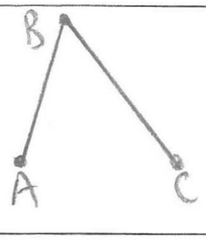
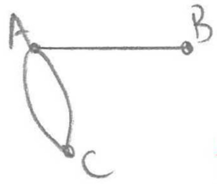
Ex 1.

Consider each of the following graphs. State whether the graph is a tree or not a tree. In each case justify your choice.

(a) 	. Not a tree as trees cannot contain cycles.
(b) 	Yes, It is a tree, as it is connected with <u>no</u> cycles (Acyclic).
(c) 	Not a tree as it is <u>not</u> connected.
(d) 	Not a tree as it contains cycles

Ex 2.

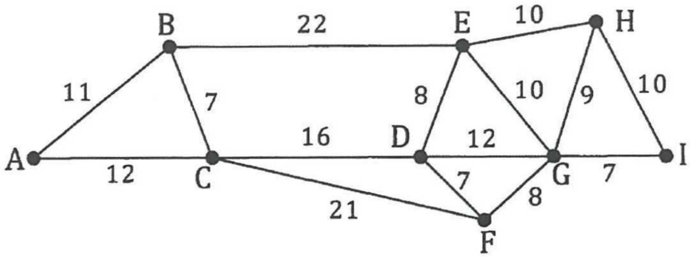
An adjacency matrix is a square matrix used to represent graphs. The elements of the matrix indicate whether pairs of vertices are adjacent or not.
For each of the given adjacency matrices construct the corresponding graph and state if the graph is a tree. If the graph is not at tree state why and what feature of the matrix indicates this.

(a) $\begin{array}{cc} & \begin{array}{ccc}A & B & C\end{array}\\ \begin{array}{c}A\\B\\C\end{array} & \begin{bmatrix}0 & 1 & 1\\1 & 0 & 1\\1 & 1 & 0\end{bmatrix}\end{array}$		⇒ <u>Not</u> a tree because it contains a cycle. The matrix shows that all of the vertices are connected and hence form a cycle.
(b) $\begin{array}{cc} & \begin{array}{ccc}A & B & C\end{array}\\ \begin{array}{c}A\\B\\C\end{array} & \begin{bmatrix}0 & 1 & 0\\1 & 0 & 1\\0 & 1 & 0\end{bmatrix}\end{array}$		A tree as it is connected and acyclic (No cycles). The matrix shows that there is no edge between A and C and hence acyclic.
(c) $\begin{array}{cc} & \begin{array}{ccc}A & B & C\end{array}\\ \begin{array}{c}A\\B\\C\end{array} & \begin{bmatrix}0 & 1 & 2\\1 & 0 & 0\\2 & 0 & 0\end{bmatrix}\end{array}$		<u>Not</u> a tree because it contains a cycle. The matrix shows that there are 2 edges between A and C, which form a cycle.

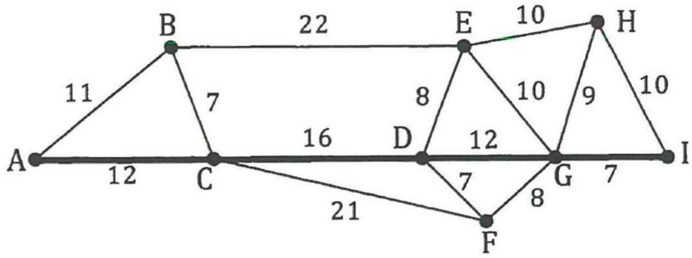
Complete Ex 5A

Spanning trees.

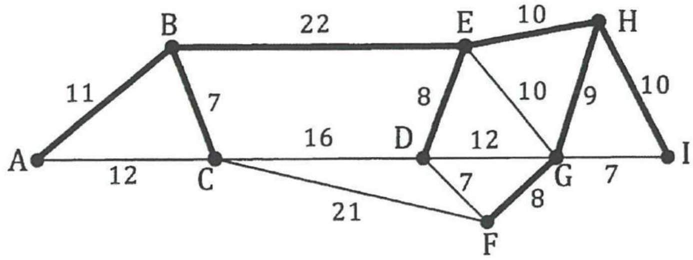
The network on the right shows the roads that exist between nine farms, A to I, with the numbers indicating distances in kilometres. The water authority wish to connect these farms to the mains water supply by laying water pipes alongside existing roads.



This is not the same as finding the shortest route from A to I because this shortest route, shown on the right, does not connect up all the farms (B, E, F and H are not "on line").



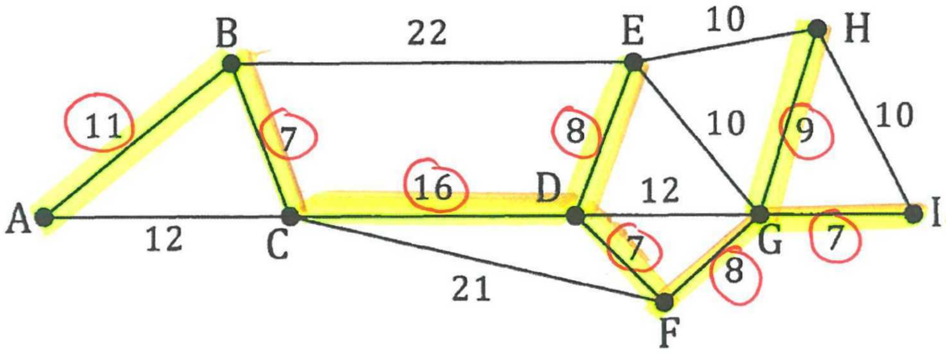
Instead we need to find a system that includes all the farms but which has no unnecessary connections. What we need is a **minimum spanning tree** for this network.



Can you find a shorter spanning tree?
Can you find the shortest spanning tree?
How long is it?

- Have a go:

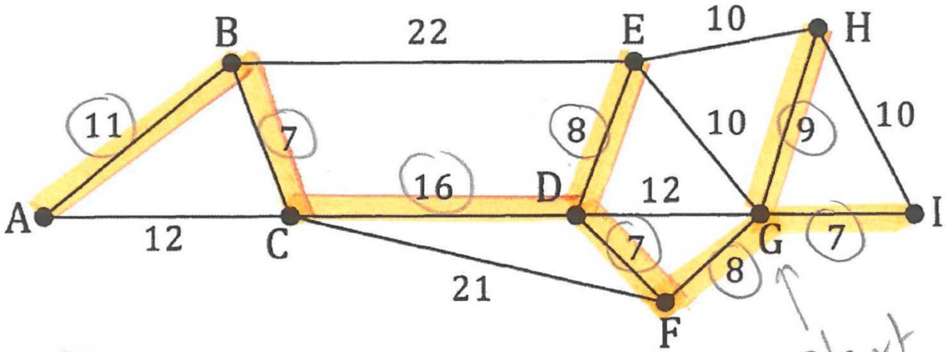
start



11
7
16
8
7
8
9
7

Total = 73 units

- Have a go at a different vertex:

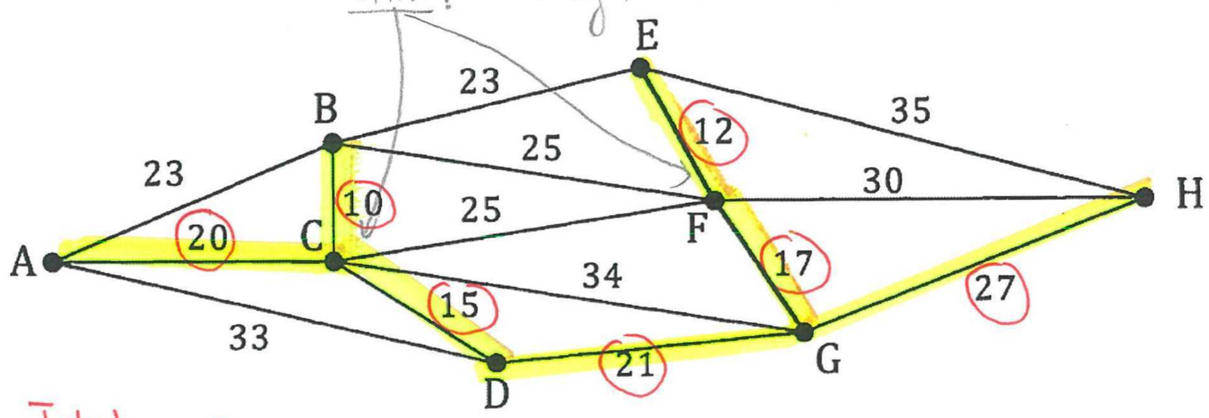


start

Note: You can start at any vertex.

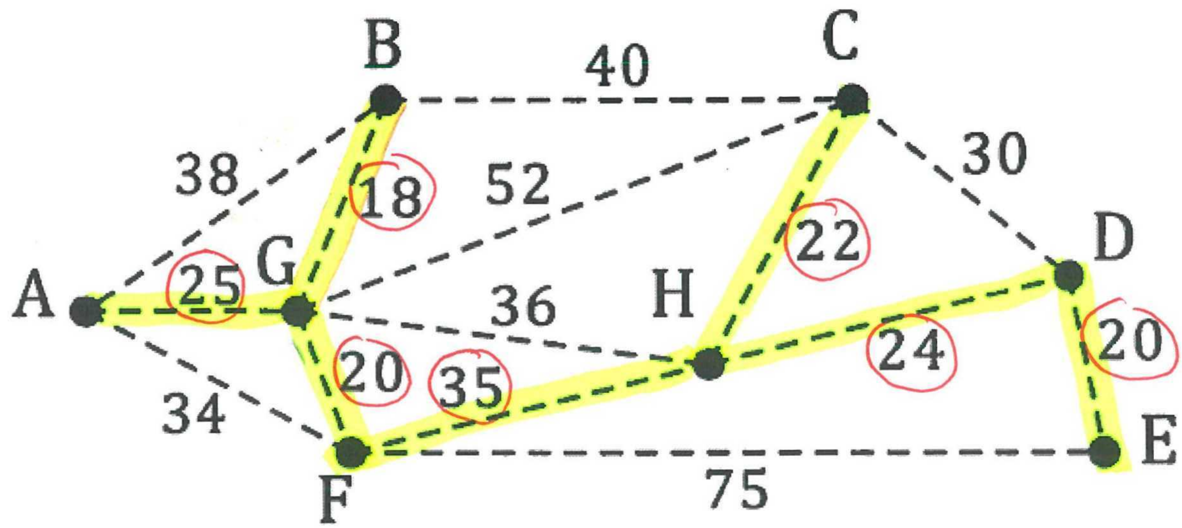
If you think you have found the minimum spanning tree for the network shown above then try to find it for the network shown below.

Hint: starting from C or F



Total = 20 + 10 + 15 + 21 + 17 + 12 + 27

= 122 units



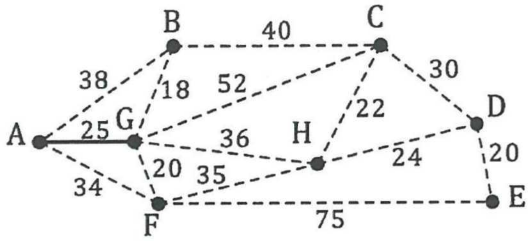
Hint: starting from G

Total = 25 + 18 + 20 + 35 + 22 + 24 + 20

= 164 units

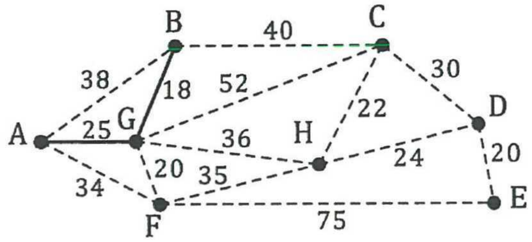
Minimum spanning trees – a systematic approach.

Consider the network on the right.

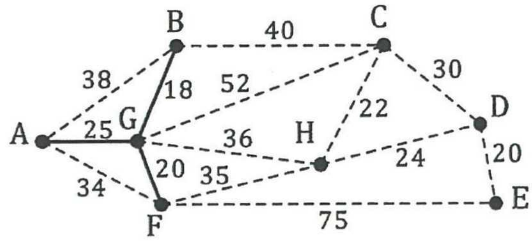


To determine the minimum spanning tree systematically we can in fact start at any vertex but we will start at A as that seems a logical place to start.

From A we next bring "on line" the vertex that is A's *nearest neighbour*. Point G, being just 25 units from A is the nearest neighbour in this example.

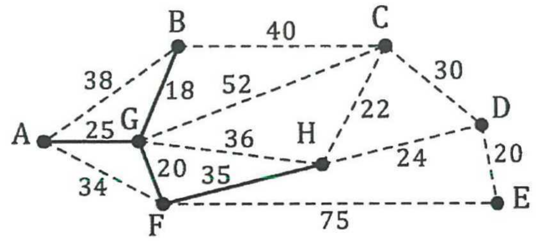


Now that we have A and G on line we next connect to the vertex that is closest to one of these two vertices. In our example B is the next nearest neighbour to one of our on line points A and G.

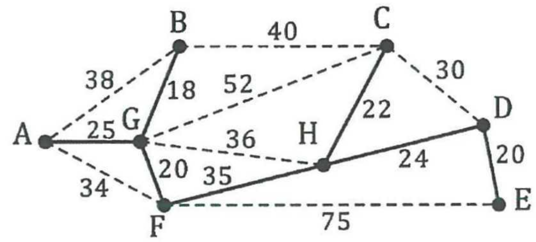


Now that we have A, G and B on line we now look for the vertex that is not yet on line but that is the nearest neighbour to one of our on line points.

Thus point F, being just 20 units from G, is the next nearest neighbour.



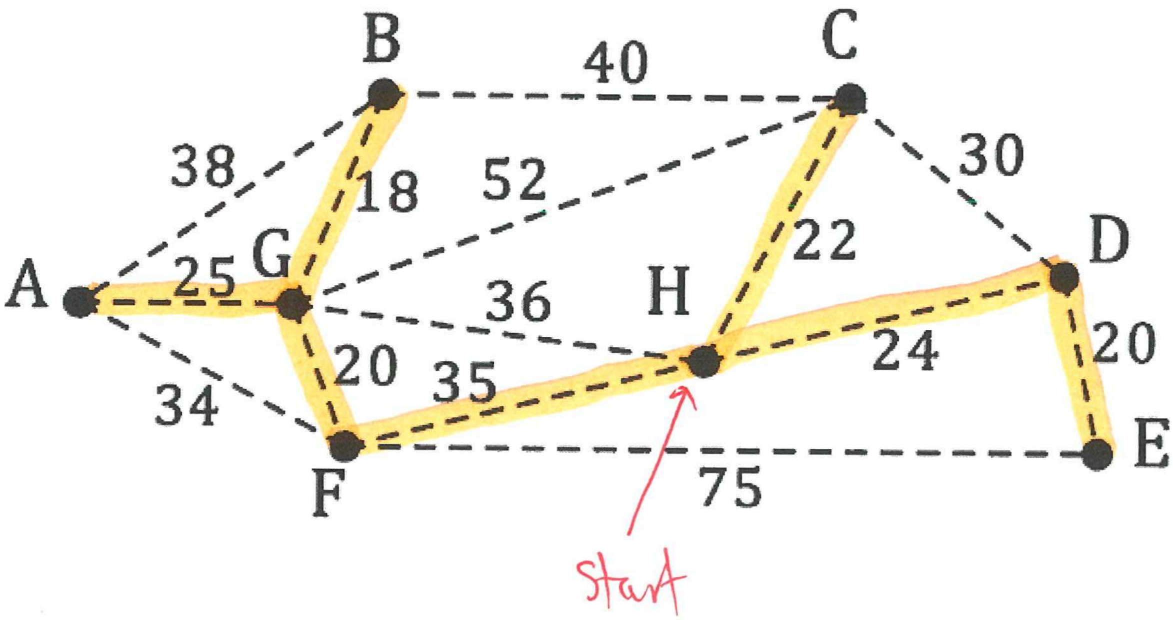
We now have A, G, B and F on line and again look for the next nearest neighbour. In this case point A is only 34 units from F but we already have A on line so this connection would be pointless. (It would give an unwanted *cycle* in our connection.) Thus we choose to bring H on line as it is the nearest neighbour not yet on line.



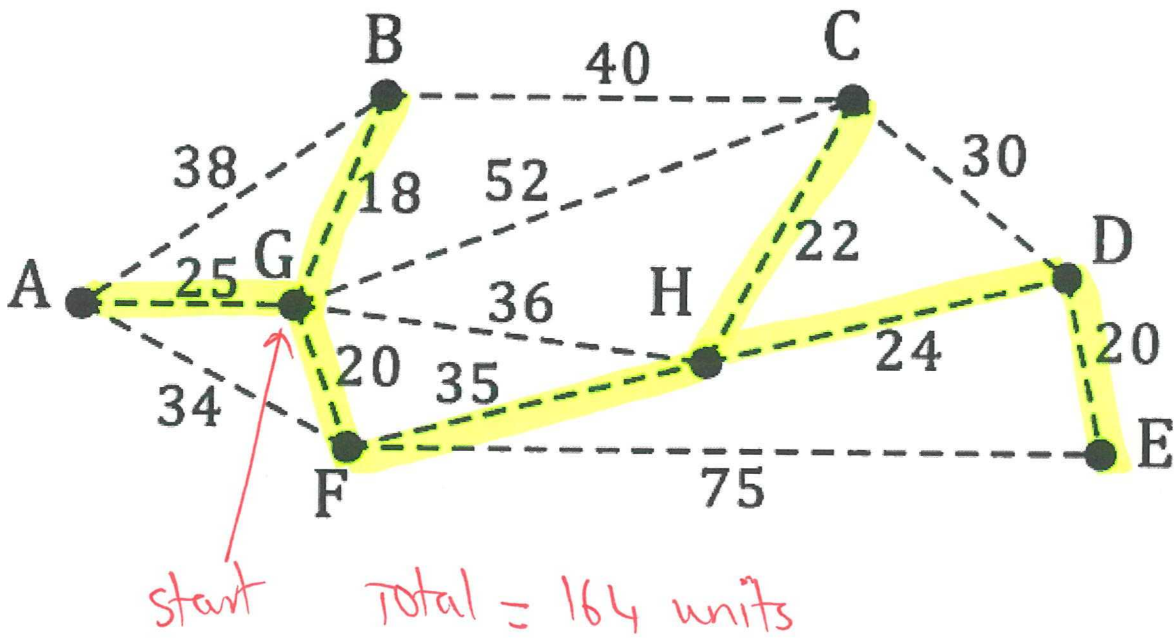
Continuing in this way until all points are on line gives the minimum spanning tree shown on the right.

Of course this process would usually be carried out on one copy of the network. The five copies used here are purely to illustrate the steps of the method.

Question: Can we start at **any vertex** other than A?



Total = 164 units



Total = 164 units

Hint: Starting from the vertex which has the most connections.

The method explained on the previous page, in which we:

> *choose any vertex as our initial "on line" vertex and then build up the spanning tree by connecting online vertices to the "nearest neighbour", whilst always making sure that no cycles are introduced,*

is called **Prim's algorithm.**

Prim's Algorithm

Prim's algorithm is a systematic method for finding a minimum spanning tree for a weighted, connected undirected graph or network.

Prim's Algorithm for Constructing a Minimum Spanning Tree from a Graph or Network

(i) Choose a starting node. This can be any node in the network because all nodes will need to be connected to construct a minimum spanning tree.

(ii) Examine all arcs which are connected to this node and choose the one with the lowest value. Join your starting node and the node at the end of the least value edge. This is the start of your spanning tree.

(iii) Examine all arcs connected to your spanning tree, select the one with the lowest value and add it to your spanning tree.

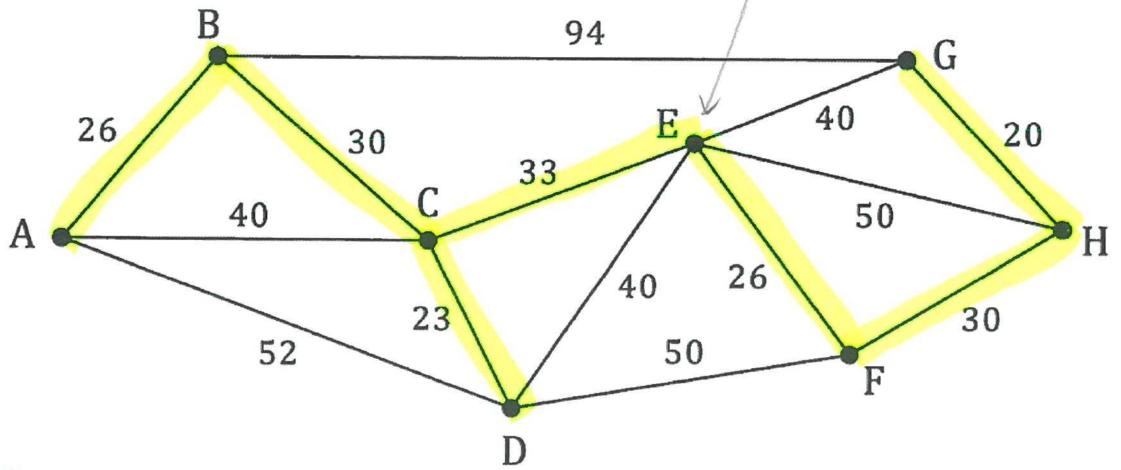
(iv) Repeat the above step until all nodes have been included in the minimum spanning tree making sure that no cycles are introduced.

NOTE: If there is more than one edge with the lowest value, then choose either.

Ex 3. Find the **minimum spanning tree** and state its **length**.

[Using **Prim's Algorithm**]

Starting node



Length = 26 + 30 + 23 + 33 + 26 + 30 + 20

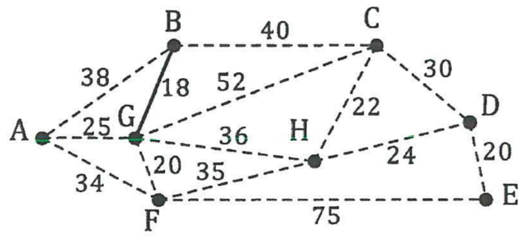
= 188 units.

Extension:

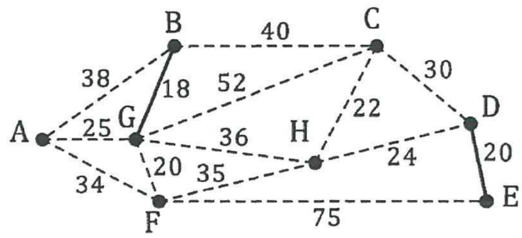
An alternative approach is to use **Kruskal's algorithm**. In this method we:

start with the "shortest" edge, then choose the next" shortest" edge, even if it is not connected to the previously selected edge, and then continue this process, always making sure that no cycles are introduced. When all vertices are "on line" we have the minimum spanning tree.

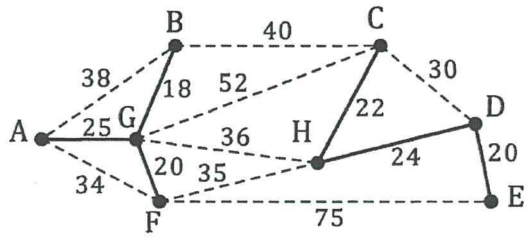
First we choose the "shortest" edge in the network, in this case GB which has a weighting of just 18.



Then we choose the next "shortest" edge. In this case there are two, GF and DE, each with a weighting of 20. It does not matter which of these two we choose. We will choose DE, as shown.



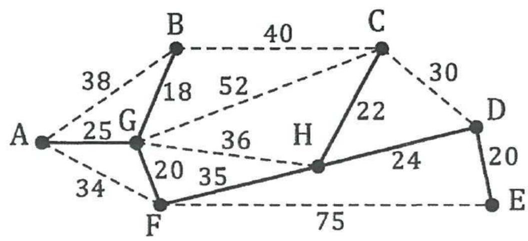
Continuing this process sees us choose GF (20) then HC (22) then HD (24) and then AG (25) as our next four edges, as shown on the right.



The next "shortest" edge will be CD (30). However this will not bring any vertex on line that is not already on line. (It will give an unnecessary cycle.)

The next "shortest" after CD is AF (34) which again introduces an unnecessary cycle.

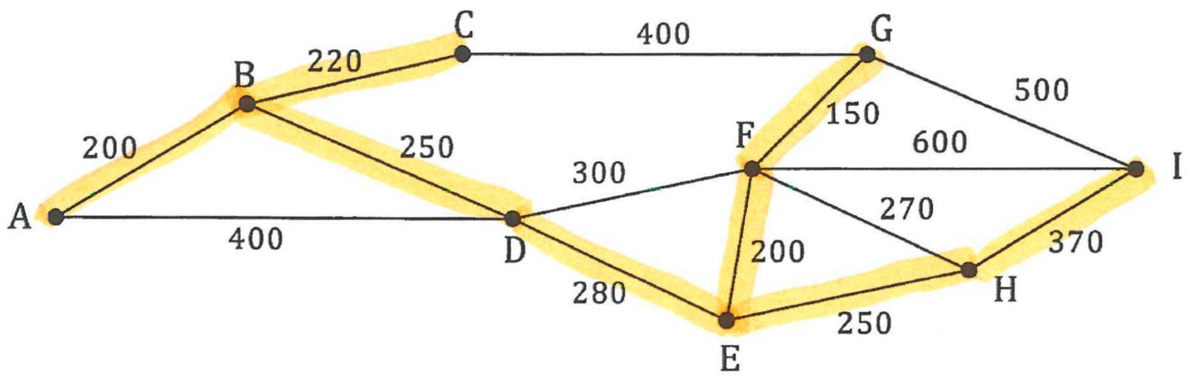
Choosing the next "shortest" which does not introduce a cycle we choose FH (35), and so the minimum spanning tree is completed.



Extension

Ex 4. Find the **minimum spanning tree** and state its **length**.

[Using **Kruskal's Algorithm**]



Total = 200 + 220 + 250 + 280 + 200 + 150 + 250 + 370

= 1920 units.

** Prim's Algorithm is a preferred method.

Ex 5.

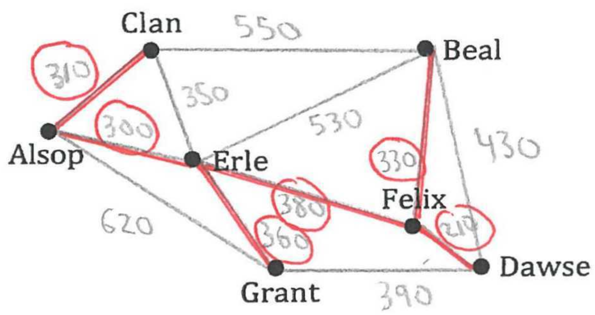
The table on the right shows the distances, in km, along the roads that exist between seven towns:
Alsop, Beal, Clan, Dawse, Erle, Felix, and Grant.

	Alsop	Beal	Clan	Dawse	Erle	Felix	Grant
Alsop	–	–	310	–	300	–	620
Beal	–	–	550	430	530	330	–
Clan	310	550	–	–	350	–	–
Dawse	–	430	–	–	–	210	390
Erle	300	530	350	–	–	380	360
Felix	–	330	–	210	380	–	–
Grant	620	–	–	390	360	–	–

(A "–" indicates that there is no direct road between those towns.)

The sketch on the right shows the approximate positions of the seven towns with respect to each other.

(a) Make a copy of this sketch and show on your copy the roads that exist between the towns and the lengths of these roads.



(b) Water pipes are to be laid along some of these roads so that the seven towns are, either directly or indirectly, connected to each other.

Determine the minimum length of piping required.

Minimum length = 310 + 300 + 360 + 380 + 330 + 210

= 1890 km

Note: Minimum spanning tree from the "distance table" to be discussed later!! ⇒ different technique.

Complete Ex 5B

Minimum spanning tree from the distances table.

In the last question of the previous exercise it was reasonably easy to transfer the details given in the table onto a network diagram because we were given a diagram showing the approximate locations of the towns with respect to each other. We then used the network diagram to determine the minimum spanning tree. However, if the network involved had been larger, and/or the relative locations of the vertices not have been known, the task of creating the network diagram with all of the weightings in place would have been more difficult. Fortunately it is possible to determine the minimum spanning tree directly from the table of distances without having to create the network diagram first. This process is demonstrated on the next page. However, as you read through it you should notice that the method is actually Prim's algorithm ....

* *choose any vertex as our initial "on line" vertex and then build up the spanning tree by connecting online vertices to the "nearest neighbour" whilst always making sure that no cycles are introduced* *

applied to the table of distances rather than to the network drawing.

Prim's Algorithm

Prim's algorithm is the most suitable method for finding the minimum spanning tree when given a "distance type" table especially when the number of nodes is large and hence the drawing of a network would be difficult as well as time-consuming.

** **Prim's Algorithm for determining a Minimum Spanning Tree from a Distance Table**

(i) Choose a starting vertex or node. This can be any node in the network because all nodes will need to be connected to determine a minimum spanning tree. It is common practice to choose that column with the minimum vertex or table entry.

(ii) Cross out the row in the table that corresponds to this node.

(iii) Label the column corresponding to this node 1, look down this column for the minimum number and circle it.

(iv) Cross out the row of this minimum value and label the corresponding column 2.

(v) Look down columns labelled 1 and 2 for the minimum number, circle it, cross out the row, label the column 3 and carry on in this manner until all the columns have been numbered.

(vi) The length of the minimum spanning tree is the sum of the circled numbers.

Ex 6. Prim's Algorithm - Different Approach (from Ex 3).

Consider the table of distances shown on the right, which is actually the one from the last question of the previous exercise.

	A	B	C	D	E	F	G
A	–	–	310	–	300	–	620
B	–	–	550	430	530	330	–
C	310	550	–	–	350	–	–
D	–	430	–	–	–	210	390
E	300	530	350	–	–	380	360
F	–	330	–	210	380	–	–
G	620	–	–	390	360	–	–

Whilst we can start at any vertex we will choose to start at A.
With A *on line* we look down the A column to see which is the shortest connection that can be made from A.
In this case it is the 300 km connection to town E.

	A	B	C	D	E	F	G
A	–	–	310	–	300	–	620
B	–	–	550	430	530	330	–
C	310	550	–	–	350	–	–
D	–	430	–	–	–	210	390
E	300	530	350	–	–	380	360
F	–	330	–	210	380	–	–
G	620	–	–	390	360	–	–

Having chosen to look down the *A column* for a connection from A we then rule a line through the *A row* to ensure that we do not, at some later stage, use this line to make an unnecessary connection back to A.
(Alternatively, we could have chosen to look across *row* A for the shortest connection, and then rule a line through *column* A.)

	A	B	C	D	E	F	G
~~A~~			~~310~~		~~300~~		~~620~~
B	–	–	550	430	530	330	–
C	310	550	–	–	350	–	–
D	–	430	–	–	–	210	390
E	300	530	350	–	–	380	360
F	–	330	–	210	380	–	–
G	620	–	–	390	360	–	–

Having selected the 300 km connection to vertex E we now have both A and E *on line* and indicate this by the arrows at the top of these two columns.
Again, to avoid making unnecessary connections at a later stage we rule a line through the E *row*.
We now look down our two *on line* columns, A and E, and choose the minimum connection that can be made from one of these two vertices.

	A	B	C	D	E	F	G
~~A~~			~~310~~		~~300~~		~~620~~
B	–	–	550	430	530	330	–
C	310	550	–	–	350	–	–
D	–	430	–	–	–	210	390
~~E~~	~~300~~	~~530~~	~~350~~			~~380~~	~~360~~
F	–	330	–	210	380	–	–
G	620	–	–	390	360	–	–

This gives us the 310 km road from A to C as our next connection.

This brings vertex C on line and we rule a line through the C row.

	A	B	C	D	E	F	G
~~A~~			~~310~~		~~300~~		~~620~~
B	–	–	550	430	530	330	–
~~C~~	310	~~550~~			~~350~~		
D	–	430	–	–	–	210	390
~~E~~	300	~~530~~	~~350~~			~~380~~	~~360~~
F	–	330	–	210	380	–	–
G	620	–	–	390	360	–	–

We now look down our three *on line* columns, A, C and E, and choose the minimum connection that can be made from one of these three vertices.

This gives us the 360 km road from E to G as our next connection.

	A	B	C	D	E	F	G
~~A~~			~~310~~		~~300~~		~~620~~
B	–	–	550	430	530	330	–
~~C~~	310	~~550~~			~~350~~		
D	–	430	–	–	–	210	390
~~E~~	300	~~530~~	~~350~~			~~380~~	~~360~~
F	–	330	–	210	380	–	–
G	620	–	–	390	360	–	–

This brings town G on line and we rule a line through the G row.

We continue this process until all of the towns are on line. The circled items in the table indicate the direct connections that together form the minimum spanning tree.

	A	B	C	D	E	F	G
~~A~~			~~310~~		~~300~~		~~620~~
B	–	–	550	430	530	330	–
~~C~~	310	~~550~~			~~350~~		
D	–	430	–	–	–	210	390
~~E~~	300	~~530~~	~~350~~			~~380~~	~~360~~
F	–	330	–	210	380	–	–
~~G~~	~~620~~			~~390~~	360		

From the completed table on the right we see that the roads AC, AE, EF, EG, FB and FD form the minimum spanning tree, with a total length of 1890 km.

	A	B	C	D	E	F	G
~~A~~			~~310~~		~~300~~		~~620~~
~~B~~			~~550~~	~~430~~	~~530~~	330	
~~C~~	310	~~550~~			~~350~~		
~~D~~		~~430~~				210	390
~~E~~	300	~~530~~	~~350~~			~~380~~	~~360~~
~~F~~		~~330~~		~~210~~	380		
~~G~~	~~620~~			~~390~~	360		

Ex 7. Determine the length of the minimum spanning trees for the networks below.

a)

	A	B	C	D
A	–	–	21	20
B	–	–	39	35
C	21	39	–	24
D	20	35	24	–

∴ Minimum length = 21 + 20 + 35

= 76 units.

b)

	A	B	C	D	E
A	-	2.7	2.3	-	2.5
B	2.7	-	2.4	2.1	2.9
C	2.3	2.4	-	1.8	1.9
D	-	2.1	1.8	-	2.2
E	2.5	2.9	1.9	2.2	-

Minimum length = 2.3 + 1.8 + 1.9 + 2.1

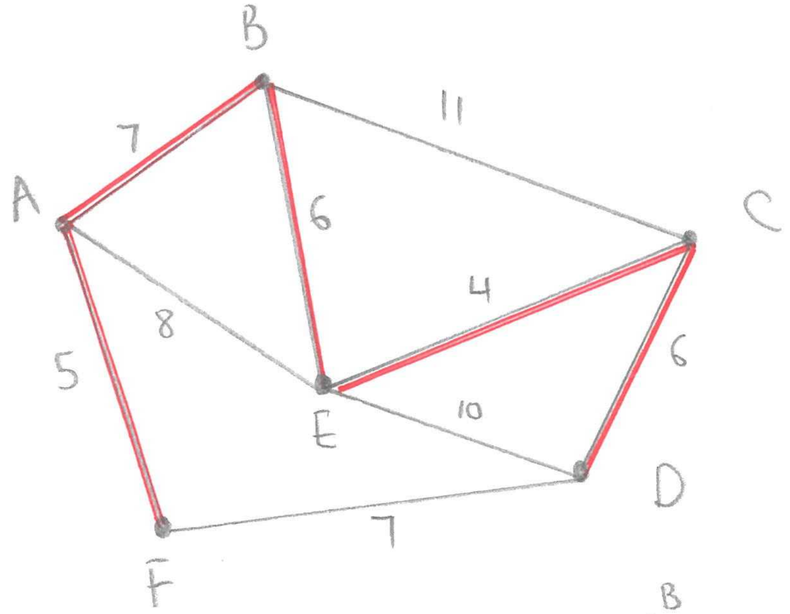
= 8.1 units

Ex 8. "Application of minimum spanning tree"

A water treatment plant has six storage tanks which need to be connected by pipelines in order to transfer water from one tank to another either directly or indirectly. The estimated cost, in thousands of dollars, of installing a pipeline between any two tanks is given in the table opposite. The dash " - " entries indicate that a pipeline cannot be installed.

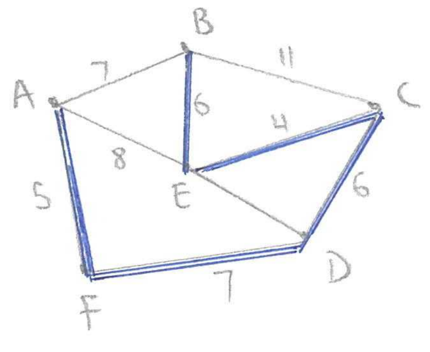
	A	B	C	D	E	F
A	-	7	-	-	8	5
B	7	-	11	-	6	-
C	-	11	-	6	4	-
D	-	-	6	-	10	7
E	8	6	4	10	-	-
F	5	-	-	7	-	-

(a) Construct a network which shows all of the cost estimates of the pipelines between the tanks.



Solution 1

Hint: start from E.



Solution 2

(b) On your network (a) draw the minimal cost spanning tree.

There are <u>2</u> possible solutions.

(c) Calculate an estimate of the minimum cost of connecting these thanks.

Minimum cost $= 5+7+6+4+6 = \$28$ thousand

$= \$28{,}000$

(d) It was discovered that the connection between tanks A and E could not proceed. How would this affect the minimum cost? Explain.

Since the connection between tanks A and E is <u>not</u> on the minimum spanning tree, there is <u>no</u> affect on the minimum cost.

Complete Ex 5C