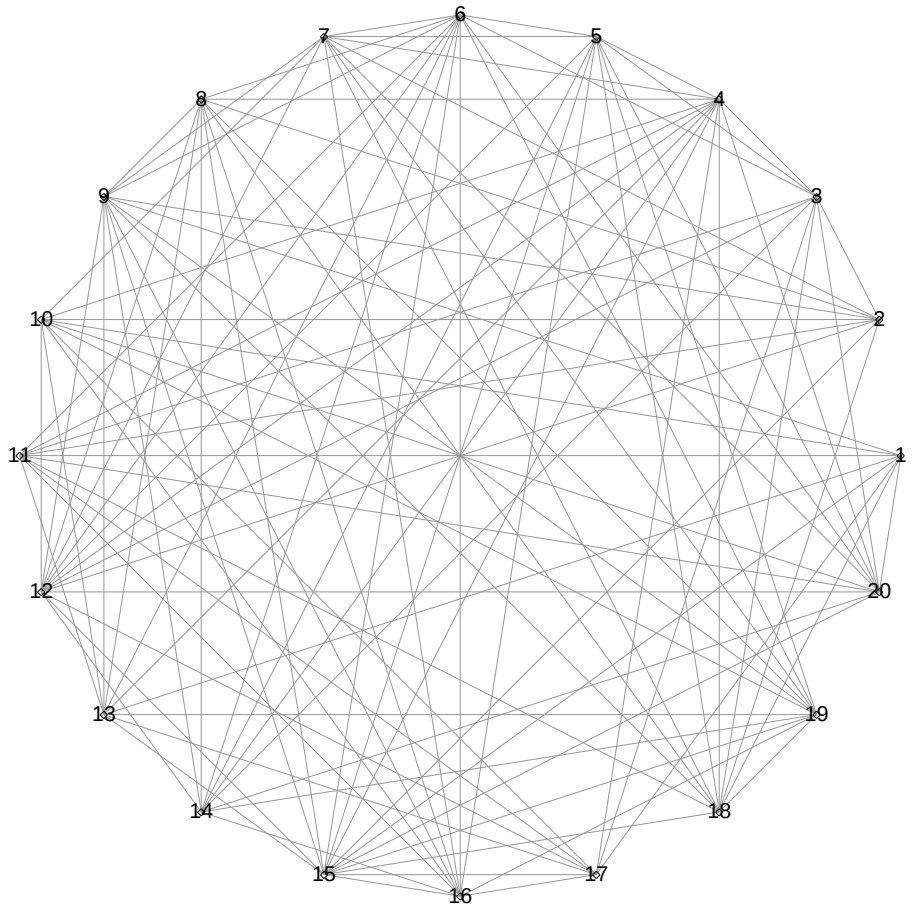
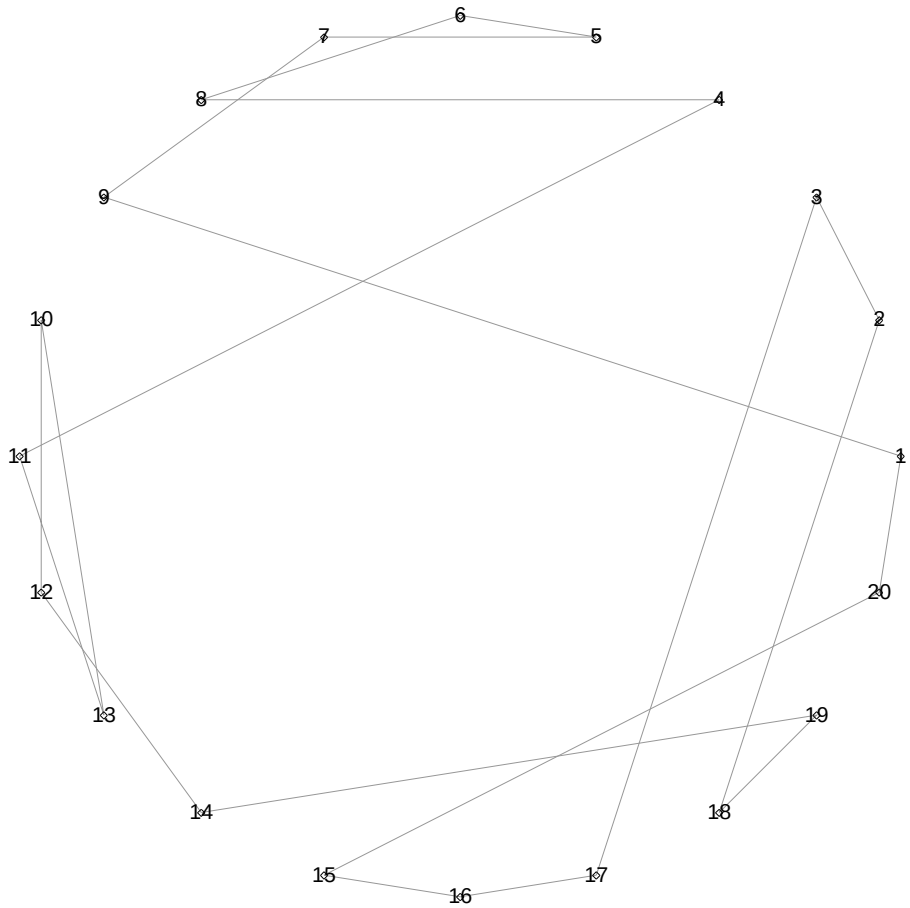
```
> # the networks package contains graph theory functions
> with(networks):
> # this file contains procedures for the travelling salesman
  problem
> read ‘Q:/223/travel_sales.m‘;
> randomize(109877); # this sets the seed so that the
  output can be checked: please use the seed you have been assigned
  - you can find this on the list on the web page for the  course.
> # this creates a graph with n=20 vertices and randomly assigned
  edges and weights.
> R:=random_G(20):
> draw(R);
```




```
> # The graph above is a randomly generated weighted graph with 20
  vertices. To see the weights of edges I could replace the colon on
  the following line with a semicolon.
```

```
>
> eweight(R):
> # The function random_G() may output a graph which is not
  Hamiltonian. If so I just try it
  again. It seems likely that in this case the graph is Hamiltonion. I
think I can see a     Hamiltonian  closed path and I check this out
by typing in the edges of a possible H. c. p.   into the induce
command.

> H:=induce({e5, e65, e87, e66, e13, e79, e75, e85, e63, e58, e50,
  e51, e54, e69, e15, e27, e22, e109, e33, e82},R):
> draw(H);
```




```
> # H above consists of just the edges of the Hamiltonian closed
  path. I'm now satisfied that  this graph is Hamiltonian so I
  continue to look for a lower bound on the weight of Hamil. closed
```

```
  paths. First I check the total weight of the H.c.p. above using
  the function total_eweight().
> total_eweight(H);
```

258

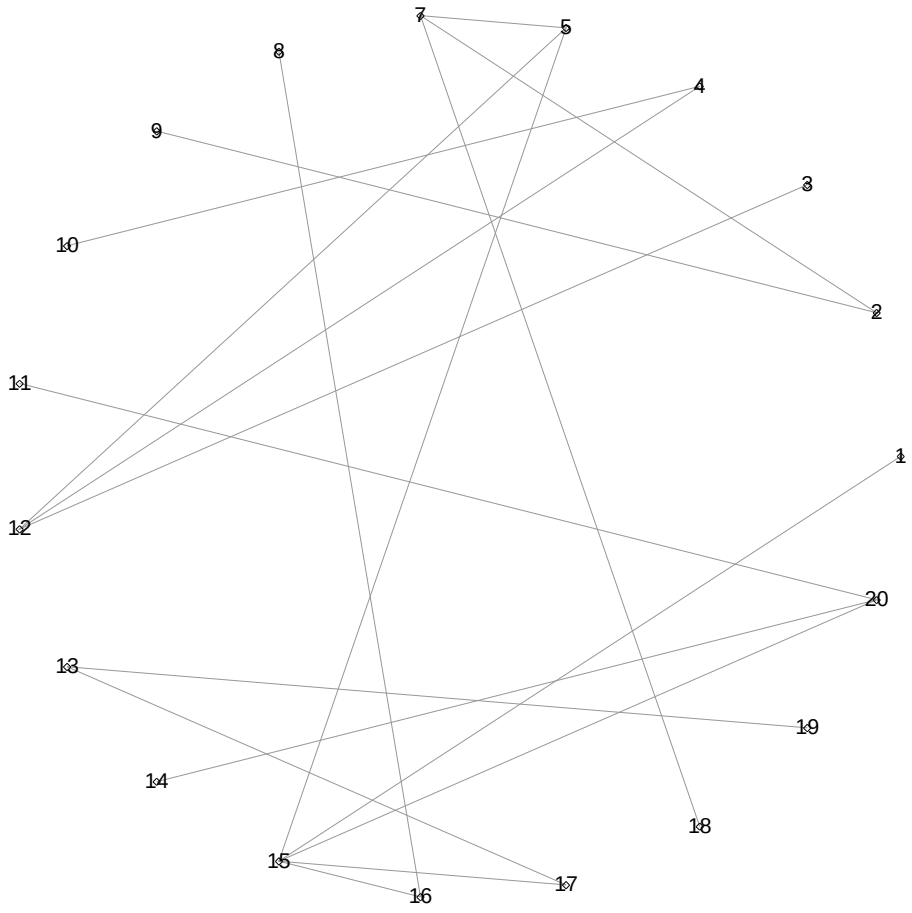
```
> # The function del_span(G) will take each vertex v of a graph G in
  turn, delete it and  find a minimal weight spanning tree of G-v.
  The output is an  array each entry of which a pair [T(n),w(n)].
  Here T(n) is a  minimal spanning tree of G -(vertex n) and w(n)
  is its weight. In the example below the array is output as a
  column with weights down the RHS. To see the output I must use the
  eval() function.
> X:=del_span(R):
> eval(X);
```

$$\begin{bmatrix} ?_{1,1} & 65 \\ ?_{2,1} & 69 \\ ?_{3,1} & 70 \\ ?_{4,1} & 71 \\ ?_{5,1} & 71 \\ ?_{6,1} & 74 \\ ?_{7,1} & 64 \\ ?_{8,1} & 63 \\ ?_{9,1} & 71 \\ ?_{10,1} & 67 \\ ?_{11,1} & 73 \\ ?_{12,1} & 71 \\ ?_{13,1} & 64 \\ ?_{14,1} & 62 \\ ?_{15,1} & 82 \\ ?_{16,1} & 69 \\ ?_{17,1} & 66 \\ ?_{18,1} & 62 \\ ?_{19,1} & 64 \\ ?_{20,1} & 69 \end{bmatrix}$$

```
> M:=X[6,1];
```

$$M := min\_span_{6,1}$$

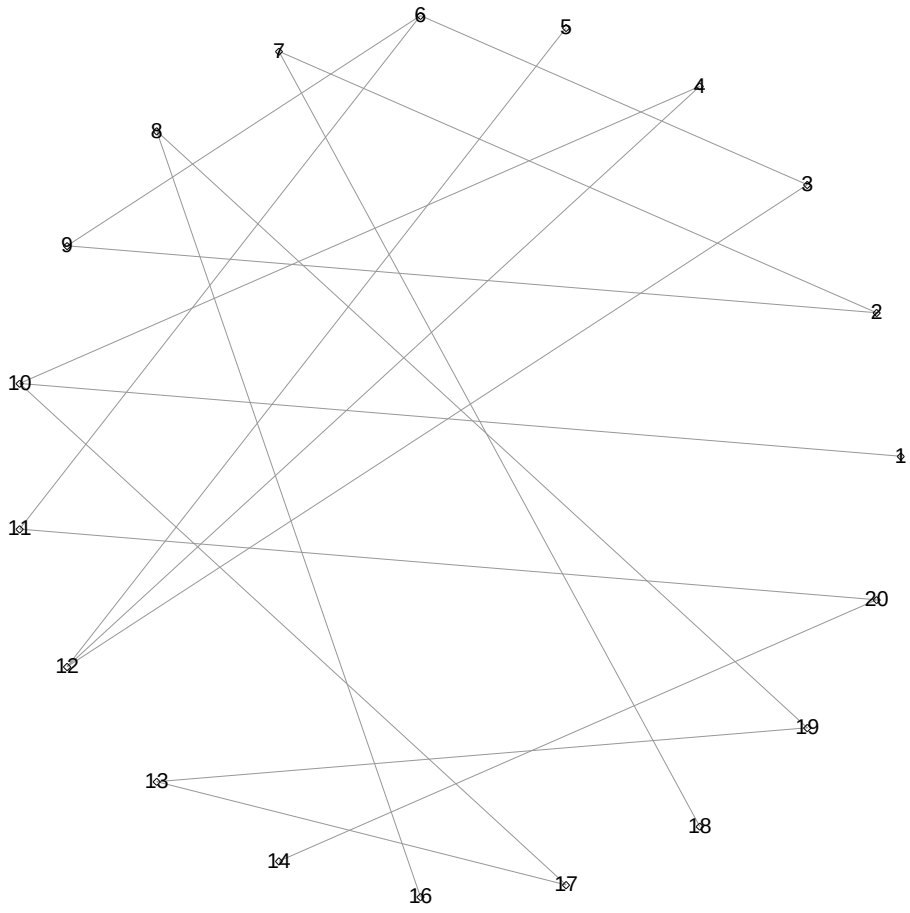
```
> # M is the minimal spanning tree obtained by removing vertex 6
  from R. From the array X above I can see that the weight of M,
  being entry [6,1] is 74 i.e. entry [6,2].
> draw(M);
```




```
> # Similarly N is the minimal spanning tree of the graph obtained
  from R by removing vertex 15 and N has weight 82.
> N:=X[15,1];
```

$$N := min\_span_{15,\,1}$$

```
> draw(N);
```




```
> # The edges incident to vertex 15 are found using the incident()
  function.
> incident(15,R);
```

$$\{e2, e18, e24, e30, e43, e48, e59, e60, e64, e65, e76, e87, e88, e89\}$$

```
> # Their weights can be seen using eweight(); (copy and paste the
  list of edges from above)
> eweight([e2, e18, e24, e30, e43, e48, e59, e60, e64, e65, e76,
  e87, e88, e89],R);
```

$$[4, 18, 1, 17, 8, 17, 10, 11, 19, 7, 15, 1, 3, 9]$$

```
> # This can all be done in one if you prefer:
> eweight([op(incident(15,R))],R);
```

$$[4, 18, 1, 17, 8, 17, 10, 11, 19, 7, 15, 1, 3, 9]$$

```
> # A lower bound for my travelling salesman problem is therefore
```

```
> X[15,2]+1+1;
```

84

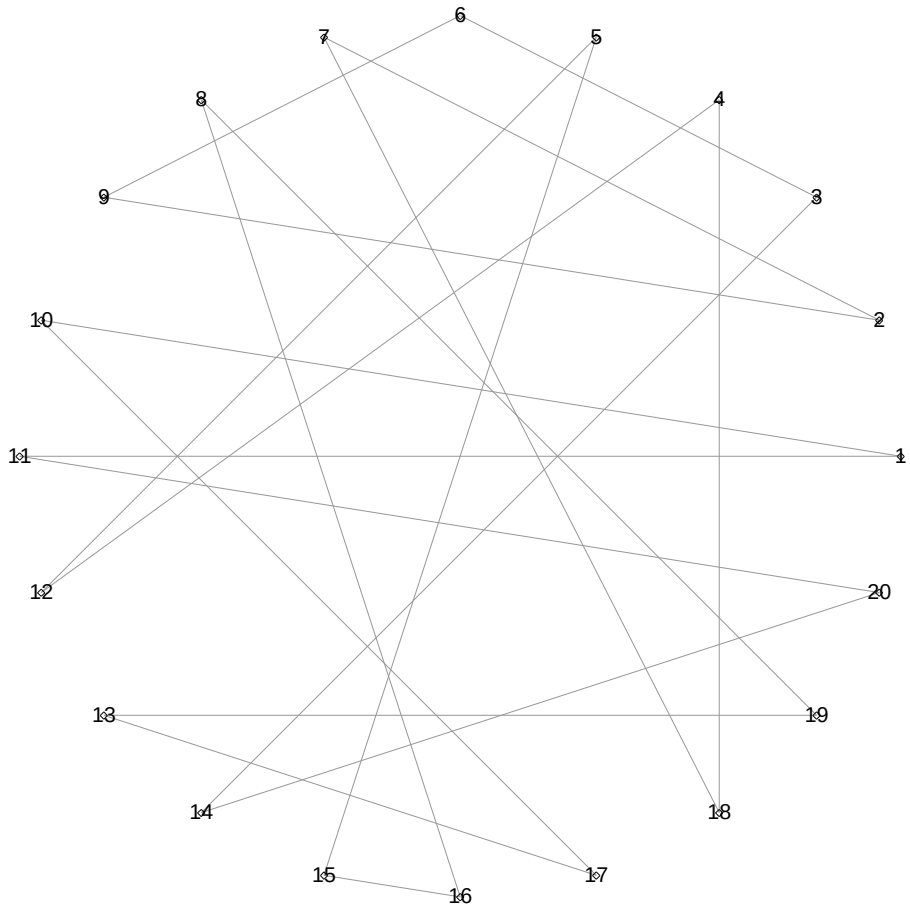
```
> # This is alot smaller than the weight of the H.c.p. I found above
  (weight 258). I can see if I can increase this bound by looking at
  removal of different vertices - but that doesn't look likely. Next
  I try to find an H.c.p. of lower weight. At least I know I need go
  no lower than 84. I need the weights of edges of R, a list of ends
  of edges and their corresponding names - use the function
  list_eweights(R) for this. I also need a list of edges of N
  (replace the colons by a semicolons below and remove the # in
  front of list_eweights)

> #list_eweights(R);
>
> edges(N):
```

$$\{e1, e6, e9, e11, e23, e28, e106, e36, e45, e61, e68, e70, e80, e86, e90, e94, e95, e99\}$$

```
> # on inspection I find the following Hamil. closed path and type
  in the edges.
> HCP:=induce({e1,e6,e9,e12,e23,e83,e106,e36,e45,e61,e20,e70,e80,e86
  ,e90,e94,e95,e99,e87,e24},R):
> # calculate its weight and draw it.
> total_eweight(HCP);
```

120

```
> draw(HCP);
```


6
7
5
8
4
9
3
2
10
1
11
12
20
13
19
14
18
15
17
16