Enron Network Analysis

In this case study, we will go through and understand the **employee network in Enron** using a subset of the Enron dataset.

Context

The story of **Enron** is a story of a **company that reached immense heights to deepest lows in no time**. Enron Corp. was one of the biggest firms in the United States and was delivering splendid performance on wall street. However, **the company witnessed a sudden drop in prices and declared bankruptcy**. How one of the most powerful businesses in the US, and the world, disintegrated overnight is still a puzzle to many.

The Enron leadership was involved in one of the biggest frauds and this particular fraud has been an area of interest for many researchers and ML practitioners.

In this case study, we have **a subset of 50 senior officials**. The idea is to **build a network from the emails, sent and received by those senior officials, to better understand the connections and highlight the important nodes in this group.**

Steps

- Read the data and understand the structure of the data.
- Put the data into a graph.
- Identify important nodes from the visualization.
- Calculate the centrality measures and quantify the importance.
- Highlight the important nodes through color coding and comment on the roles / importance that can be figured out from this.

References

- Dataset - https://www.cs.cmu.edu/~./enron/

Import the files

```
import pandas as pd

import matplotlib.pyplot as plt

from decorator import decorator

import networkx as nx

from networkx.utils import create_random_state, create_py_random_state
```

Loading the data

```
from google.colab import drive
drive.mount('/content/drive')
```

```
Mounted at /content/drive
```

```
data = pd.read_csv('/content/drive/MyDrive/1.PROGETTI/2024.09.Corso Mit /03.Week - Data Analysis & Visualization/Live Virtual C
```

```
data.shape
```

```
(304, 2)
```

```
data.head()
```

	From	To
0	0	1
1	1	0
2	1	2
3	1	3
4	1	4

Passaggi successivi: Genera codice con data | Visualizza grafici consigliati | New interactive sheet

Generating Graphs

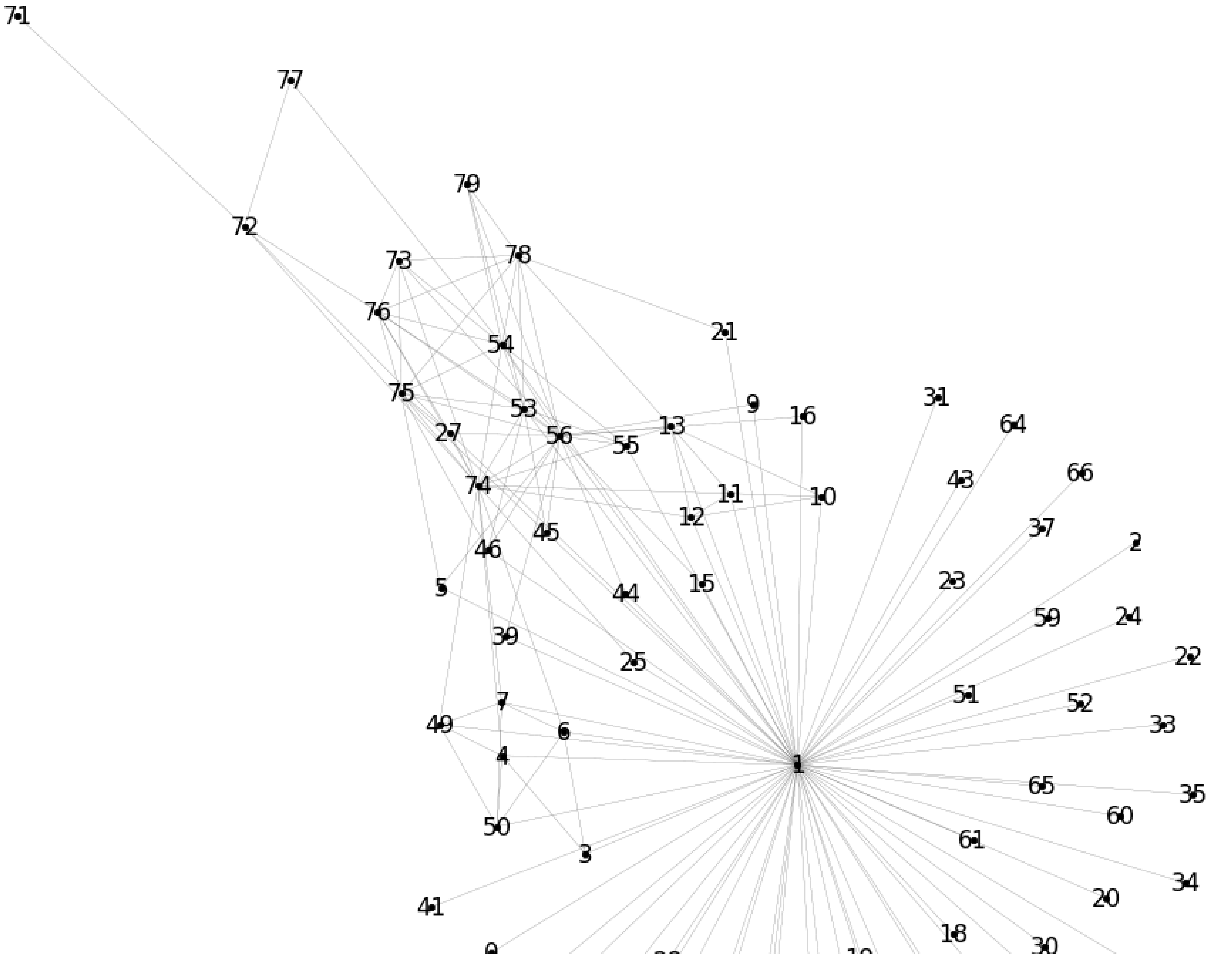
Note: In case you face an error while running the below code, please upgrade the decorator library, by running the following code, to resolve the error.

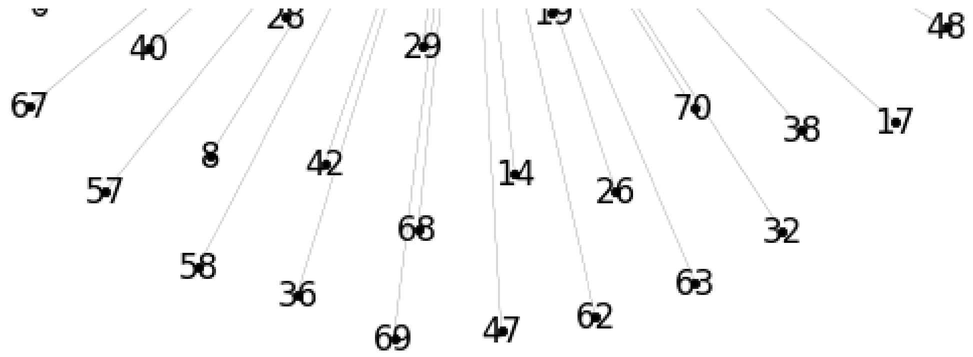
!pip install --upgrade decorator

```
G = nx.Graph()
```

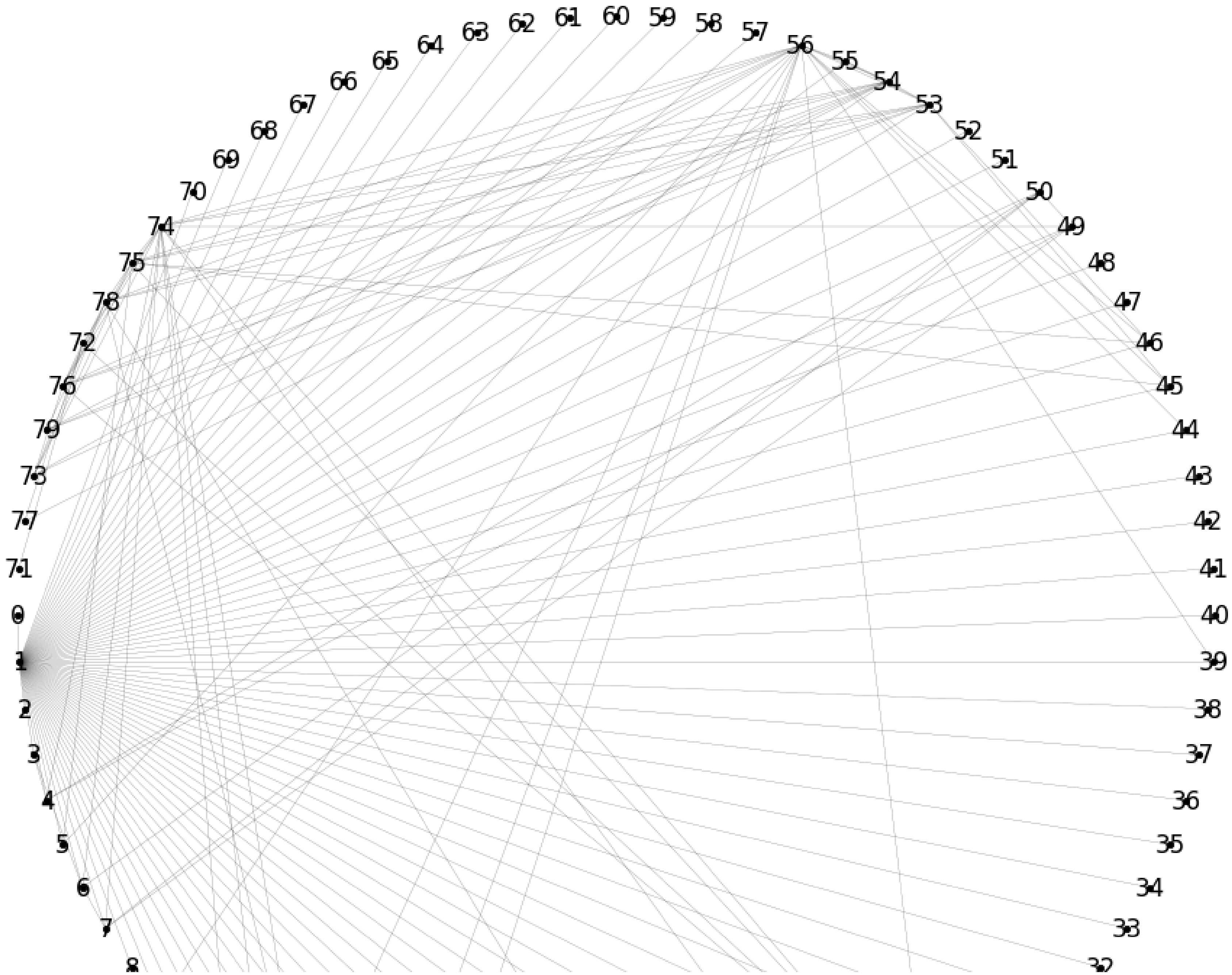
```
G = nx.from_pandas_edgelist(data, 'From', 'To')

plt.figure(figsize = (10, 10))

options = {
    "node_color": "black",
    "node_size": 10,
    "linewidths": 0.5,
    "width": 0.1,
}

nx.draw(G, with_labels = True, **options)
```

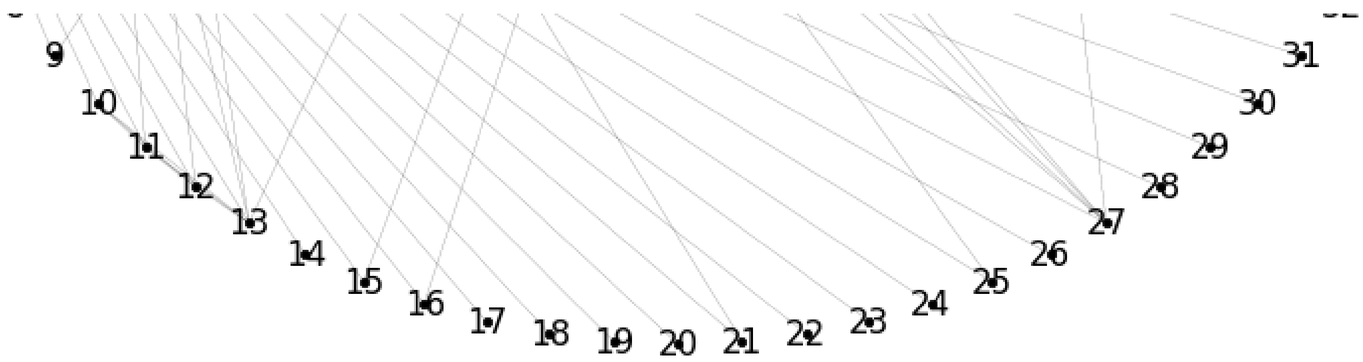
```

plt.show()
```


71
77
79
72
73
78
76
54
21
75
53
9
16
31
56
13
64
27
55
74
11
10
43
66
12
37
2
45
46
23
5
44
15
59
24
39
25
22
7
51
52
49
6
33
1
4
65
35
60
50
61
3
34
41
20
18
30

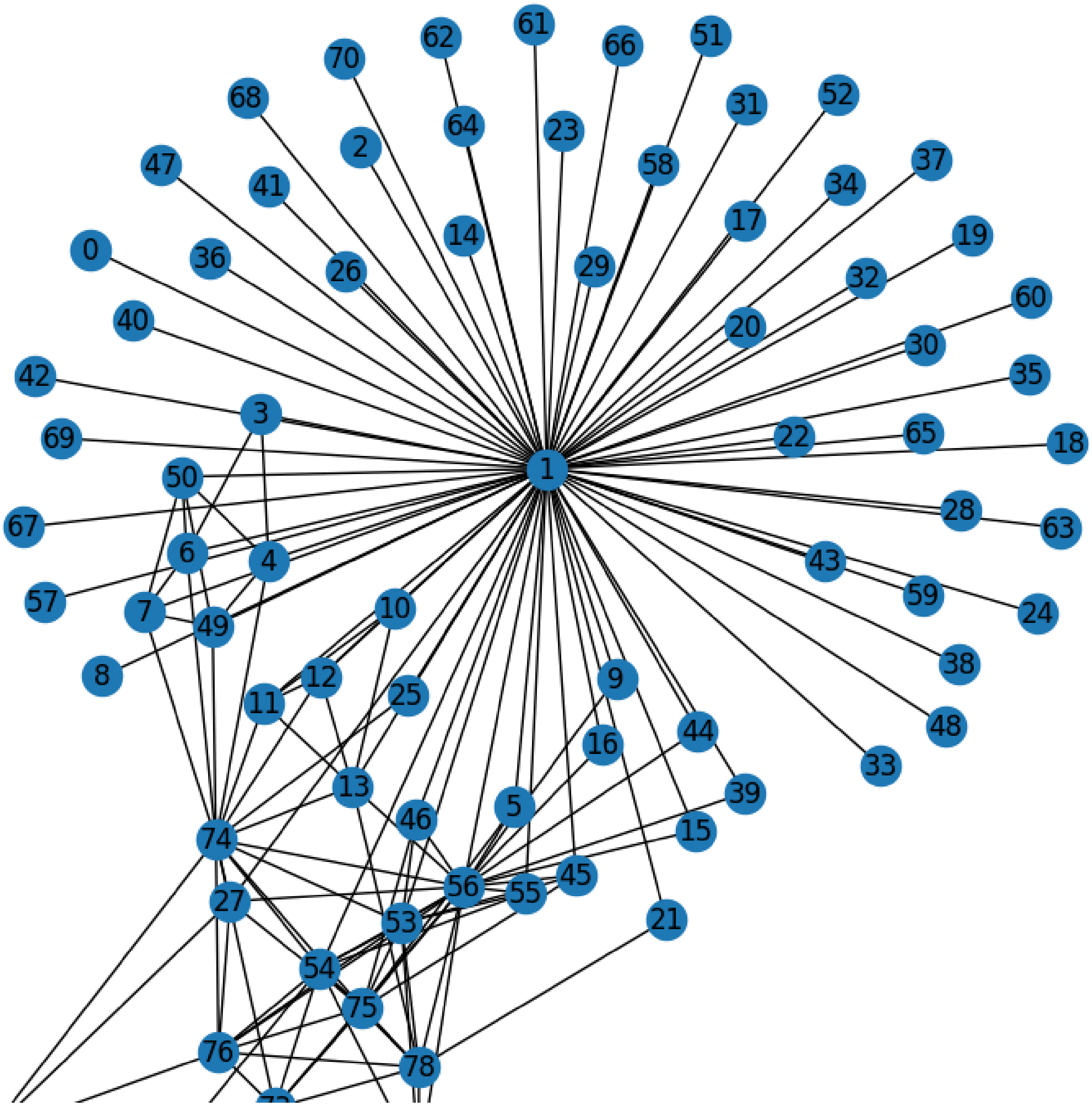


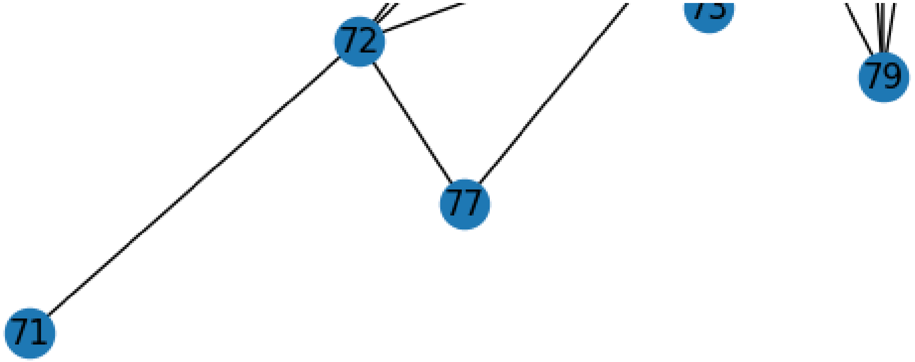
```
plt.figure(figsize = (10, 10))

nx.draw_shell(G, with_labels = True, **options)
```


61
60
59
58
57
56
55
54
53
52
51
50
49
48
47
46
45
44
43
42
41
40
39
38
37
36
35
34
33
32
62
63
64
65
66
67
68
69
70
74
75
78
72
76
79
73
77
71
0
1
2
3
4
5
6
7
8



```
plt.figure(figsize = (10, 10))

# With the default parameters
nx.draw_spring(G, with_labels = True)
```


61
51
62
66
70
52
31
68
64
23
2
58
37
47
41
34
17
19
0
36
26
14
29
32
60
40
20
30
35
42
3
22
65
18
69
1
50
28
63
67
6
4
43
57
7
49
10
59
24
8
12
25
9
38
11
44
48
16
13
33
39
5
46
74
15
56
45
55
27
53
21
54
75
76
78



Observations:

- Out of the 80 nodes in the dataset, 1 appears to be the most important node as it is connected with all the other nodes. We can interpret this official, perhaps as the CEO.
- Other important nodes are also highlighted in the visualization - 56, 54, 74, 53, 50. The circular visualization is a better visualization approach to highlight the important nodes.
- There are internal team structures that appear from the visualization but are not very clear as to which nodes are part of which teams.

```
# Let us quickly look at the degree of the nodes
for i in G.degree():
    print(i)
```

```
(33, 1)
(34, 1)
(35, 1)
(36, 1)
(37, 1)
(38, 1)
(39, 2)
(40, 1)
(41, 1)
(42, 1)
(43, 1)
(44, 2)
(45, 4)
(46, 4)
(47, 1)
(48, 1)
(49, 5)
(50, 5)
(51, 1)
(52, 1)
(53, 11)
```

```
(79, 4)
(73, 6)
(77, 2)
```

Centrality Measures

```
deg_cen = nx.degree_centrality(G)

eig_cen = nx.eigenvector_centrality(G)

clo_cen = nx.closeness_centrality(G)

bet_cen = nx.betweenness_centrality(G)
```

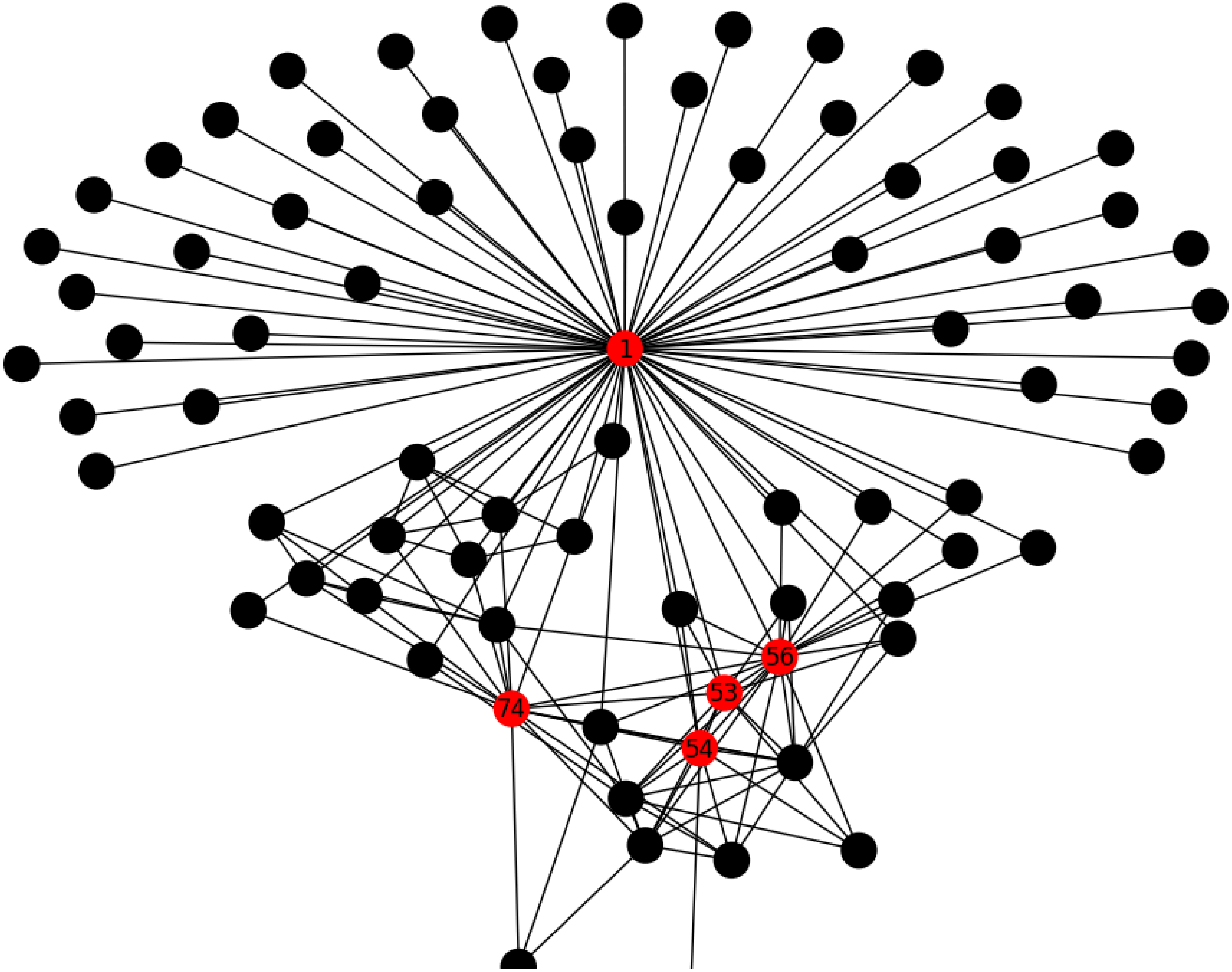
a. Degree Centrality

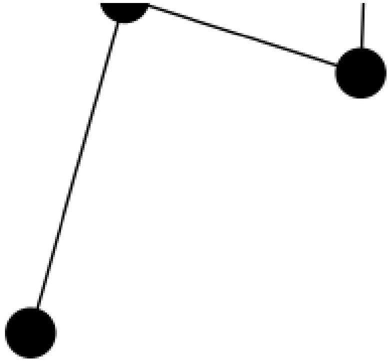
```
temp = {}

for w in sorted(deg_cen, key = deg_cen.get, reverse = True):
    temp[w] = deg_cen[w]

print("Sorted Importance of nodes in terms of deg_cen for Phase {} is {}".format(w + 1, list(temp.keys())[:5]))

print()
```

```
Sorted Importance of nodes in terms of deg_cen for Phase 72 is [1, 56, 74, 53, 54]
```

```
 # Let us color these nodes and visualize the network again

 color = []

 for node in G:

     if (node == 1 or node == 56 or node == 74 or node==53 or node==54):
         color.append('red')
```

```
    else:
        color.append('black')

plt.figure(figsize = (10, 10))

nx.draw(G, node_color = color, with_labels = True)
```


1
56
53
74
54

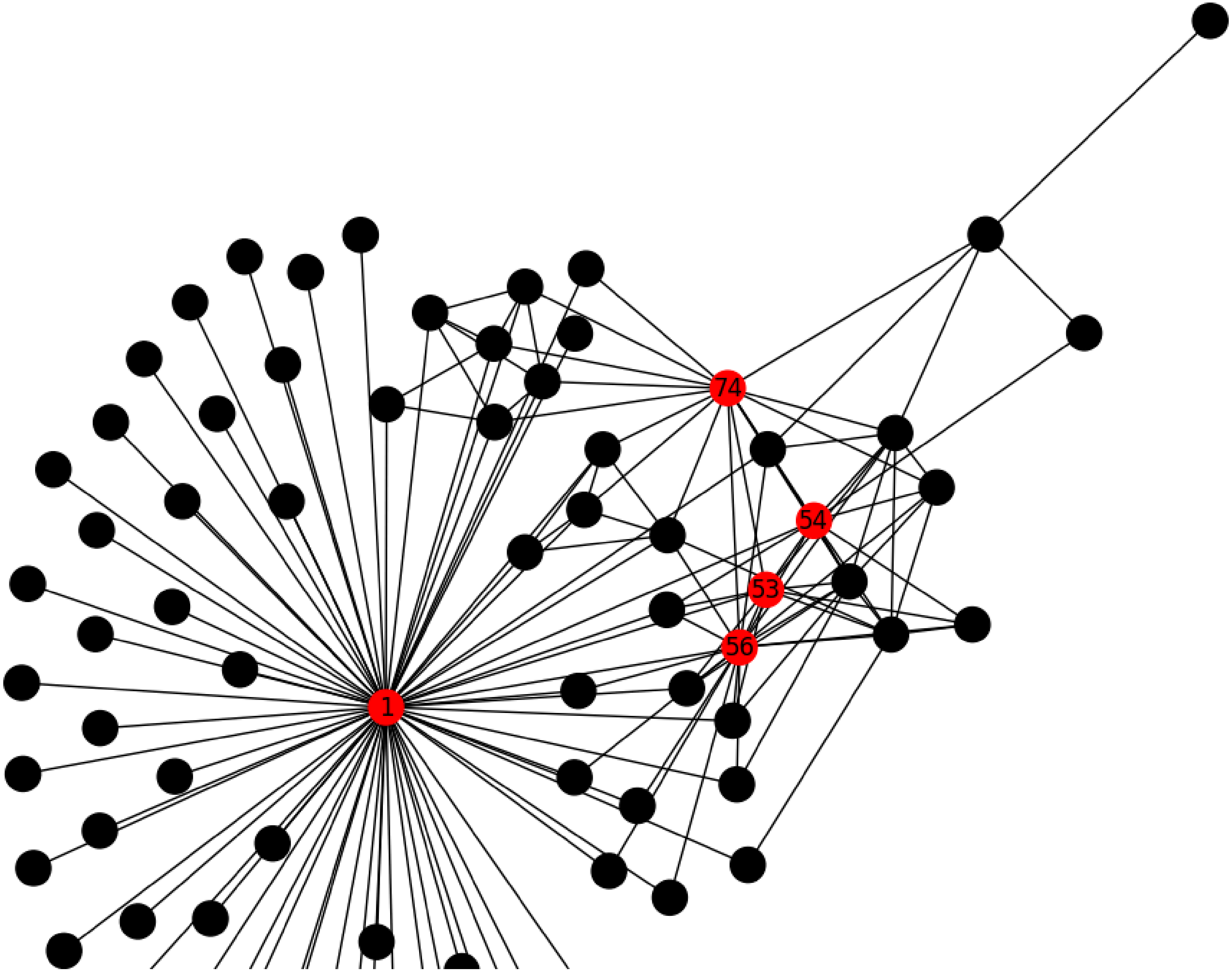


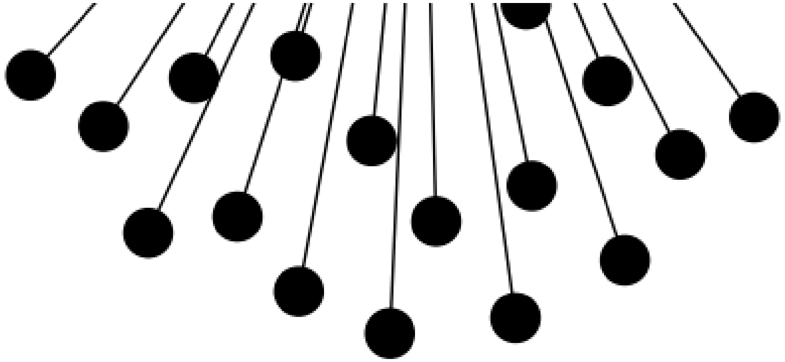
b. Eigenvector Centrality

```python
temp = {}

for w in sorted(eig_cen, key = eig_cen.get, reverse = True):
    temp[w] = eig_cen[w]

print("Sorted Importance of nodes in terms of eig_cen for Phase {} is {}".format(w + 1, list(temp.keys())[:5]))

print()
```

```
Sorted Importance of nodes in terms of eig_cen for Phase 72 is [1, 56, 74, 53, 54]
```

```python
# Let us color these nodes and visualize the network again

color = []

for node in G:

    if (node == 1 or node == 56  or node == 74 or node==53 or node==54):
        color.append('red')

    else:
        color.append('black')

plt.figure(figsize = (10, 10))

nx.draw(G, node_color = color, with_labels = True)
```


74
54
53
56
1



c. Betweenness Centrality

```
temp = {}

for w in sorted(bet_cen, key = bet_cen.get, reverse = True):
    temp[w] = bet_cen[w]

print("Sorted Importance of nodes in terms of bet_cen is {}".format(list(temp.keys())[:5]))

print()
```

```
Sorted Importance of nodes in terms of bet_cen is [1, 56, 54, 27, 74]
```

```
color = []

for node in G:

    if (node == 1 or node == 56  or node == 54 or node==27 or node==74):
        color.append('red')

    else:
        color.append('black')
```

```

plt.figure(figsize = (10, 10))

nx.draw(G, node_color = color, with_labels = True)
```