Data 620 - Week 2: Assignment 1

Leticia Salazar

February 5, 2023

Overview:

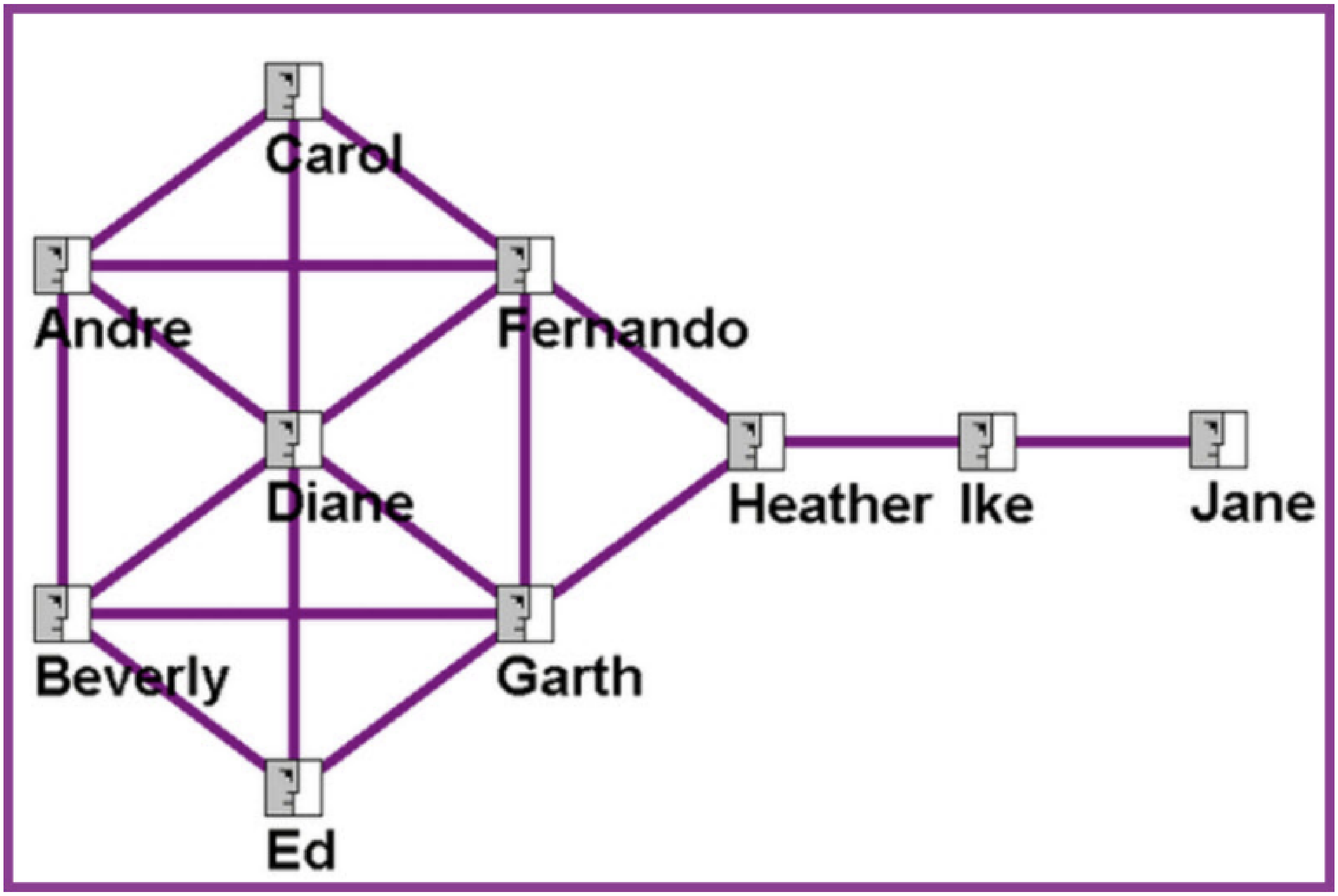
When software developers are starting to work in a new environment, they are encouraged to start with a simple "beachhead" program.

In this assignment, you're asked to create the nodes and edges for a basic graph, such as the Krackhardt kite shown below. (You're welcome to substitute data of your own choosing).

In [1]:

```
from IPython.display import Image
Image(filename = '/Users/letisalba/Desktop/Data-620/Week-2/Assignment1.png',
      width = 900, height = 950)
```

Out[1]:



You may use a package of your own choosing, such as iGraph or NetworkX, or ...? Your code should be delivered in an iPython notebook, and placed into GitHub. In your assignment link, please provide a link to your GitHub repository.

Import libraries:

In [2]:

```
import networkx as nx
import matplotlib.pyplot as plt
```

Create the graph

`G` will stand to reference the graphs we create

In [3]:

```
# create an empty graph
G = nx.Graph()
```

In [4]:

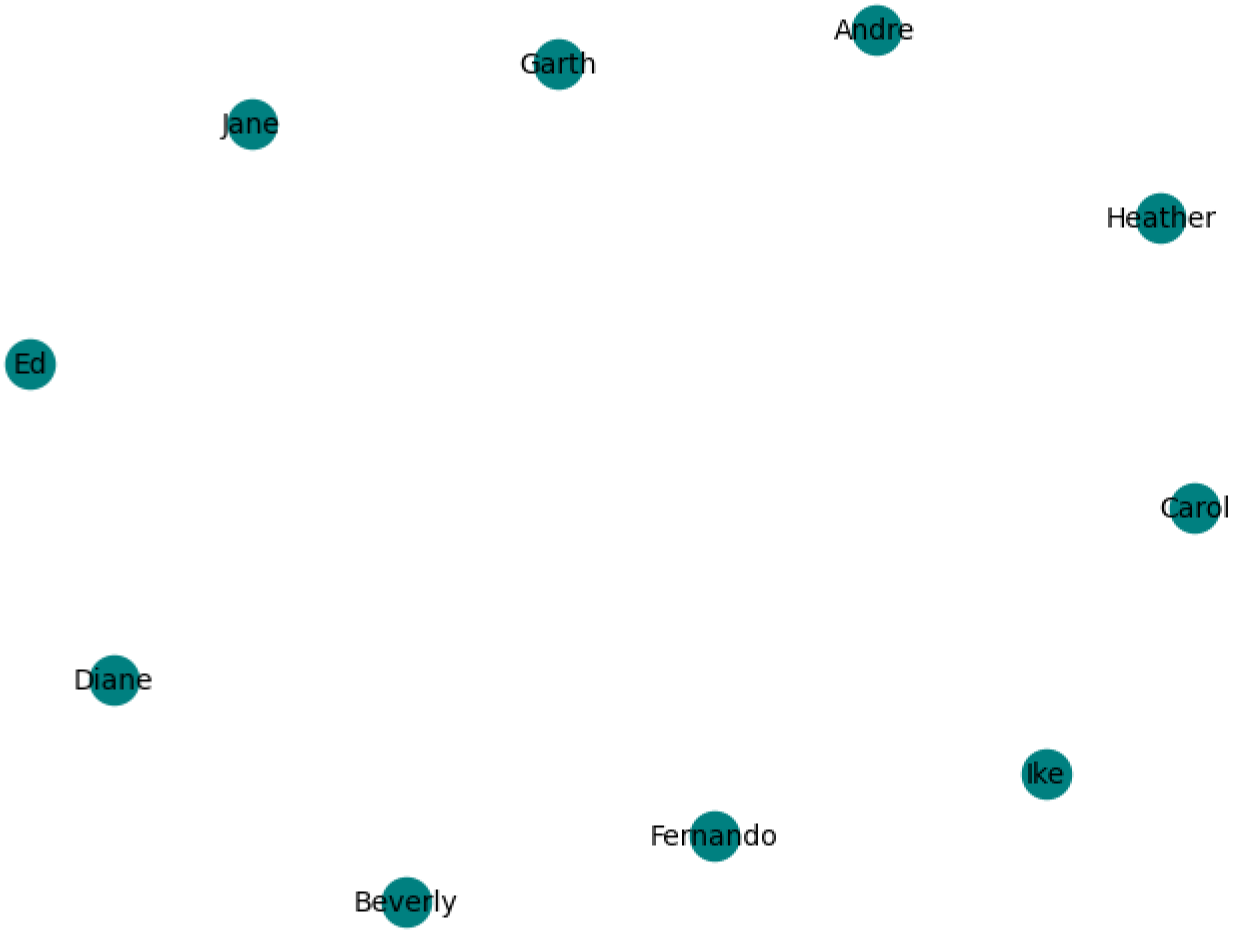
```
# Add the nodes to graph
G.add_nodes_from(['Andre', 'Carol', 'Fernando', 'Diane', 'Beverly', 'Ed',
                  'Garth', 'Heather', 'Ike', 'Jane'])
```

Quick explanation of the parameters used for the graphs:

- `with_labels` : set to True when using labels parameter
- `node_color` : color of the node
- `font_size` : size of font used to label the nodes
- `font_color` : color of the font used
- `edge_color` : color of the line used to make the connections
- `width` : how thick the edge should be

In [5]:

```
# Draw the graph
nx.draw(G, with_labels=True, node_color="teal", font_size = 10)
```



Based on the chart above I will have to make connections for each person. While I worked it out on paper I decided to start with Jane and work my way to making the connection with everyone.

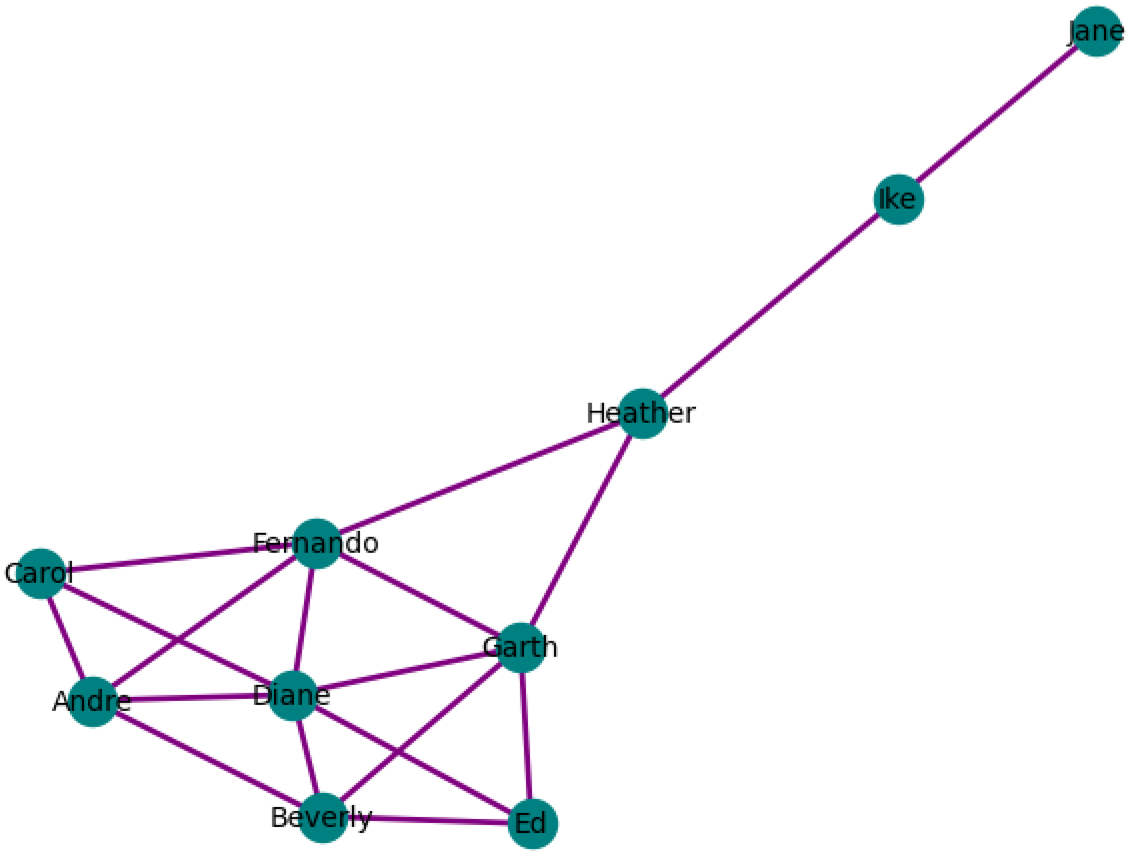
Jane = 0, Ike = 1, Heather = 2, Fernando = 3, Garth = 4, Carol = 5, Diane = 6, Ed = 7, Andre = 8, Beverly = 9

This may not seem logical to some, it helps me to work backwards and look at the connections one by one to see if I can get to the desired output.

The connections made were the following:

In [6]:
```
# Add the edges based on the connections mentioned above
G.add_edges_from([('Jane', 'Ike'), ('Ike', 'Heather'), ('Heather', 'Fernando'), ('Heathe
                  ('Fernando', 'Garth'), ('Fernando', 'Carol'), ('Fernando', 'Diane'), (
                  ('Garth', 'Ed'), ('Carol', 'Diane'), ('Carol', 'Andre'), ('Diane', 'An
                  ('Diane', 'Beverly'), ('Diane', 'Ed'), ('Ed', 'Beverly'), ('Andre', 'F
                  ('Andre', 'Beverly'), ('Beverly', 'Garth')])
```

In [7]:
```
# Draw the graph
nx.draw(G, with_labels=True, font_size=10, font_color='black', node_color='teal' ,
        edge_color='purple', width = 2)
```



Based on the drawing above I was not totally off but I still need to work on how I create the connections to look like the original drawing.

I decided to approach this differently, now starting with Andre and working my way through each person vertically to make the connections:

Andre = 0, Beverly = 1, Carol = 2, Diane = 3, Ed = 4, Fernando = 5, Garth = 6, Heather = 7, Ike = 8, Jane = 9

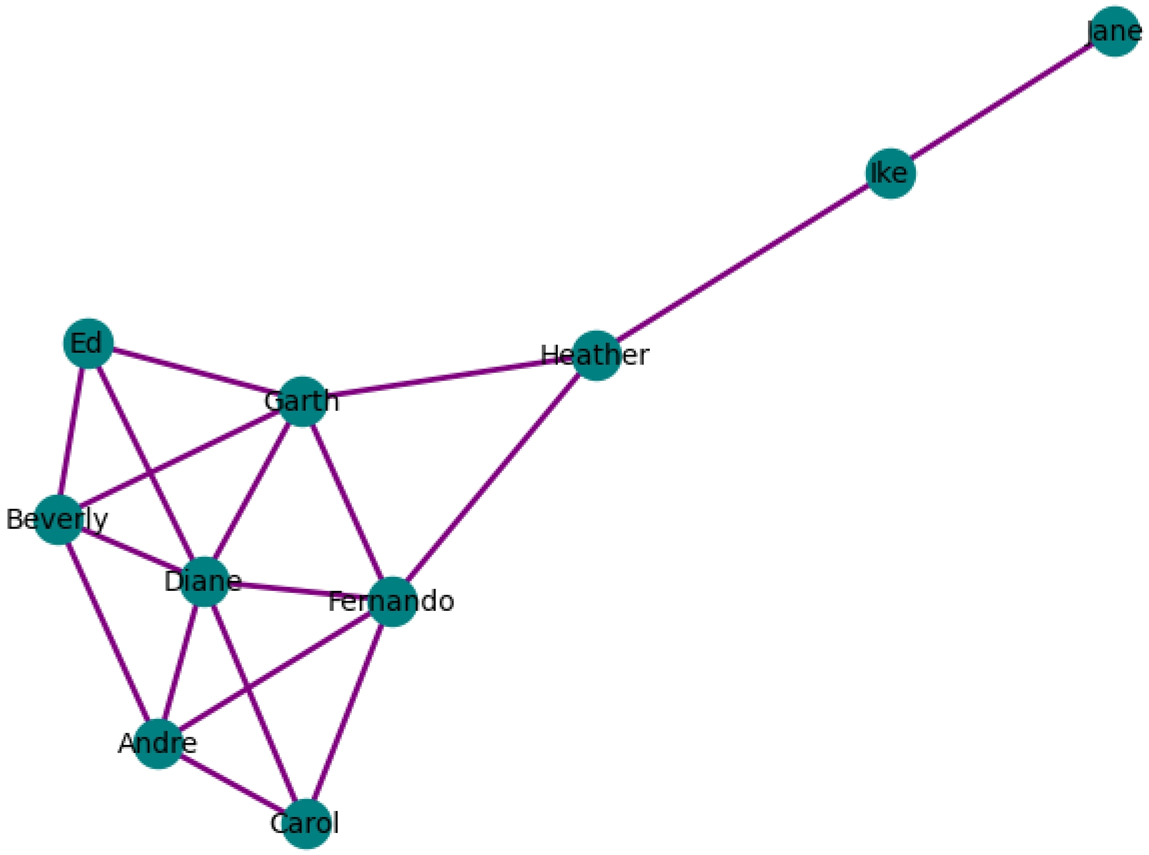
In [8]:
```
# Create a second empty graph
```

```
G_2 = nx.Graph()

# Add the edges to form the connections
G_2.add_edges_from([('Andre', 'Beverly'), ('Andre', 'Carol'), ('Andre', 'Fernando'), ('A
                    ('Beverly', 'Ed'), ('Beverly', 'Diane'), ('Beverly', 'Garth'), ('Carol
                    ('Carol', 'Diane'), ('Diane', 'Ed'), ('Diane', 'Garth'), ('Diane', 'Fe
                    ('Ed', 'Garth'), ('Fernando', 'Garth'), ('Fernando', 'Heather'), ('Gar
                    ('Heather', 'Ike'), ('Ike', 'Jane')])
```

In [9]:

```
# Draw the graph
nx.draw(G_2, with_labels=True, font_size=10, font_color='black', node_color='teal' ,
        edge_color='purple', width = 2)
```



Now although this graph does look similar to the original, there are a couple of names who are inverted.

Trying one more time, this time focusing on creating a dictionary list for the connections.

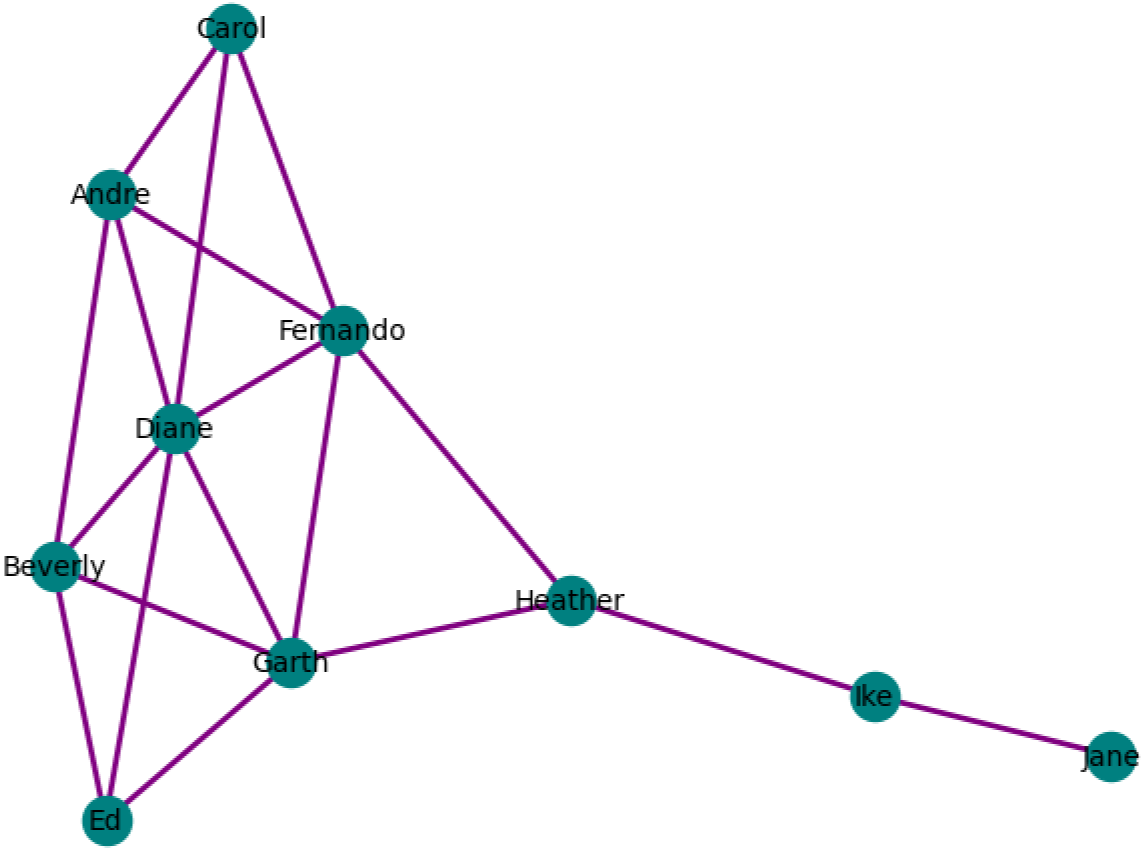
In [17]:

```
# Create a third empty graph
G_3 = nx.Graph()


# Create the dictionary list for the connections
G_3 = nx.from_dict_of_lists(
        {
            'Andre': ['Beverly', 'Carol', 'Diane', 'Fernando'],
            'Beverly': ['Andre', 'Diane', 'Ed', 'Garth'],
            'Carol': ['Andre', 'Diane', 'Fernando'],
            'Diane': ['Andre', 'Beverly', 'Carol', 'Ed', 'Fernando', 'Garth'],
            'Ed': ['Beverly', 'Diane', 'Garth'],
            'Fernando': ['Andre', 'Carol', 'Diane', 'Garth', 'Heather'],
            'Garth': ['Beverly', 'Diane', 'Ed', 'Fernando', 'Heather'],
            'Heather': ['Fernando', 'Garth', 'Ike'],
            'Ike': ['Heather', 'Jane'],
            'Jane': ['Ike'],
        }
    )
```

In [18]:

```
# Draw the graph
nx.draw(G_3, with_labels=True, font_size=10, font_color='black', node_color='teal' ,
        edge_color='purple', width = 2)
```



Overall, the graphs created were close to the Krackhardt kite provided.

GitHub Repo:

References:

- Labels for the graph: