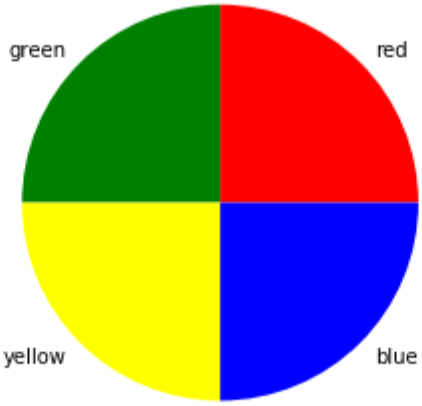
[3]
```
#!pip install bokeh
#!pip install matplotlib
#!pip install seaborn
```

```
Requirement already satisfied: seaborn in
c:\users\maniv\anaconda3\lib\site-packages (0.9.0)
Requirement already satisfied: scipy>=0.14.0 in
c:\users\maniv\anaconda3\lib\site-packages (from seaborn) (1.2.1)
Requirement already satisfied: pandas>=0.15.2 in
c:\users\maniv\anaconda3\lib\site-packages (from seaborn) (0.24.2)
Requirement already satisfied: numpy>=1.9.3 in
c:\users\maniv\anaconda3\lib\site-packages (from seaborn) (1.16.2)
Requirement already satisfied: matplotlib>=1.4.3 in
c:\users\maniv\anaconda3\lib\site-packages (from seaborn) (3.0.3)
Requirement already satisfied: pytz>=2011k in
c:\users\maniv\anaconda3\lib\site-packages (from pandas>=0.15.2-
>seaborn) (2018.9)
Requirement already satisfied: python-dateutil>=2.5.0 in
c:\users\maniv\anaconda3\lib\site-packages (from pandas>=0.15.2-
>seaborn) (2.8.0)
Requirement already satisfied: cycler>=0.10 in
c:\users\maniv\anaconda3\lib\site-packages (from matplotlib>=1.4.3-
>seaborn) (0.10.0)
Requirement already satisfied: kiwisolver>=1.0.1 in
c:\users\maniv\anaconda3\lib\site-packages (from matplotlib>=1.4.3-
>seaborn) (1.0.1)
Requirement already satisfied: pyparsing!=2.0.4,!=2.1.2,!=2.1.6,>=2.0.1
in c:\users\maniv\anaconda3\lib\site-packages (from matplotlib>=1.4.3-
>seaborn) (2.3.1)
Requirement already satisfied: six>=1.5 in
c:\users\maniv\anaconda3\lib\site-packages (from python-dateutil>=2.5.0-
>pandas>=0.15.2->seaborn) (1.12.0)
Requirement already satisfied: setuptools in
c:\users\maniv\anaconda3\lib\site-packages (from kiwisolver>=1.0.1-
>matplotlib>=1.4.3->seaborn) (40.8.0)
```

[6]
```
import matplotlib.pyplot as plt
import numpy as np
%matplotlib inline
```

[8]
```
labels = 'red', 'green', 'yellow', 'blue'
sizes = [25,25,25,25]
colors = ['red', 'green', 'yellow', 'blue']
plt.pie(sizes, labels = labels, colors = colors)
plt.axis('equal')
plt.show()
```



[9]
```
import matplotlib.pyplot as plt
import networkx as nx
```

[10]
```
def plot_weighted_graph():
    "Plot a weighted graph"

    #2. Add nodes
    G = nx.Graph() #Create a graph object called G
    node_list = ['Karpov','Kasparov','Kramnik','Anand']
    for node in node_list:
        G.add_node(node)

    #Note: You can also try a spring_layout
    pos=nx.circular_layout(G)

 nx.draw_networkx_nodes(G,pos,node_color='green',node_size=7500)

    #3. If you want, add labels to the nodes
    labels = {}
    for node_name in node_list:
        labels[str(node_name)] =str(node_name)
    nx.draw_networkx_labels(G,pos,labels,font_size=16)


    #4. Add the edges (4C2 = 6 combinations)
    #NOTE: You usually read this data in from some source
    #To keep the example self contained, I typed this out
    G.add_edge(node_list[0],node_list[1],weight=170) #Karpov vs
Kasparov
    G.add_edge(node_list[0],node_list[2],weight=15) #Karpov vs
Kramnik
    G.add_edge(node_list[0],node_list[3],weight=45) #Karpov vs
Anand
    G.add_edge(node_list[1],node_list[2],weight=49) #Kasparov vs
Kramnik
```

```
    G.add_edge(node_list[1],node_list[3],weight=51) #Kasparov vs
Anand
    G.add_edge(node_list[2],node_list[3],weight=91) #Kramnik vs
Anand

    all_weights = []
    #4 a. Iterate through the graph nodes to gather all the
weights
    for (node1,node2,data) in G.edges(data=True):
        all_weights.append(data['weight']) #we'll use this when
determining edge thickness

    #4 b. Get unique weights
    unique_weights = list(set(all_weights))

    #4 c. Plot the edges - one by one!
    for weight in unique_weights:
        #4 d. Form a filtered list with just the weight you want
to draw
        weighted_edges = [(node1,node2) for
(node1,node2,edge_attr) in G.edges(data=True) if
edge_attr['weight']==weight]
        #4 e. I think multiplying by [num_nodes/sum(all_weights)]
makes the graphs edges look cleaner
        width = weight*len(node_list)*3.0/sum(all_weights)

 nx.draw_networkx_edges(G,pos,edgelist=weighted_edges,width=width
)

    #Plot the graph
    plt.axis('off')
    plt.title('How often have they played each other?')
    plt.savefig("chess_legends.png")
    plt.show()
```

[11]
```
plot_weighted_graph()
```

```
C:\Users\maniv\Anaconda3\lib\site-
packages\networkx\drawing\nx_pylab.py:611: MatplotlibDeprecationWarning:
isinstance(..., numbers.Number)
  if cb.is_numlike(alpha):
```

[12]
```
print("Plot a weighted graph")
```

```
Plot a weighted graph
```

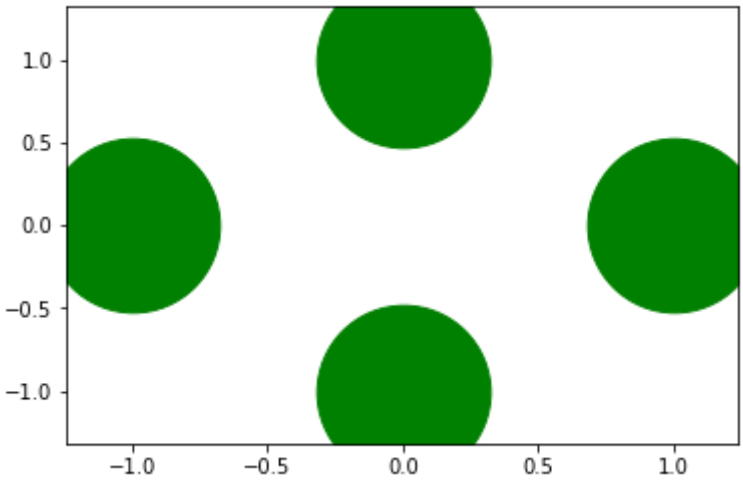
[13]
```
G = nx.Graph()
```

[14]
```
node_list = ['Karpov','Kasparov','Kramnik','Anand']
```

[15]
```
for node in node_list:
    G.add_node(node)
```

[17]
```
pos = nx.circular_layout(G)
nx.draw_networkx_nodes(G, pos, node_color = 'green',
node_size=7500)
```

```
<matplotlib.collections.PathCollection at 0x1c28c500710>
```



```
[ ]
```