*You are currently looking at **version 1.2** of this notebook. To download notebooks and datafiles, as well as get help on Jupyter notebooks in the Coursera platform, visit the Jupyter Notebook FAQ (https://www.coursera.org/learn/python-social-network-analysis/resources/yPcBs) course resource.*

Assignment 3

In this assignment you will explore measures of centrality on two networks, a friendship network in Part 1, and a blog network in Part 2.

Part 1

Answer questions 1-4 using the network `G1`, a network of friendships at a university department. Each node corresponds to a person, and an edge indicates friendship.

The network has been loaded as networkx graph object `G1`.

In [1]:
```
import networkx as nx

G1 = nx.read_gml('friendships.gml')
```

Question 1

Find the degree centrality, closeness centrality, and normalized betweeness centrality (excluding endpoints) of node 100.

This function should return a tuple of floats `(degree_centrality, closeness_centrality, betweenness_centrality)`.

In [2]:
```
def answer_one():

    # Your Code Here
    degree = nx.degree_centrality(G1)[100]
    closeness = nx.closeness_centrality(G1)[100]
    betweenness = nx.betweenness_centrality(G1)[100]

    return (degree, closeness, betweenness)

answer_one()
```

Out[2]: (0.0026501766784452294, 0.2654784240150094, 7.142902633244772e-05)

For Questions 2, 3, and 4, assume that you do not know anything about the structure of the network, except for the all the centrality values of the nodes. That is, use one of the covered centrality measures to rank the nodes and find the most appropriate candidate.

Question 2

Suppose you are employed by an online shopping website and are tasked with selecting one user in network G1 to send an online shopping voucher to. We expect that the user who receives the voucher will send it to their friends in the network. You want the voucher to reach as many nodes as possible. The voucher can be forwarded to multiple users at the same time, but the travel distance of the voucher is limited to one step, which means if the voucher travels more than one step in this network, it is no longer valid. Apply your knowledge in network centrality to select the best candidate for the voucher.

This function should return an integer, the name of the node.

In [3]:
```
def answer_two():

    # Your Code Here
    degree = nx.degree_centrality(G1)
    node = max(degree.keys(), key=(lambda key: degree[key]))

    return node

answer_two()
```

Out[3]: 105

Question 3

Now the limit of the voucher's travel distance has been removed. Because the network is connected, regardless of who you pick, every node in the network will eventually receive the voucher. However, we now want to ensure that the voucher reaches the nodes in the lowest average number of hops.

How would you change your selection strategy? Write a function to tell us who is the best candidate in the network under this condition.

This function should return an integer, the name of the node.

In [4]:
```
def answer_three():

    # Your Code Here

    closeness = nx.closeness_centrality(G1)
    node = max(closeness.keys(), key=(lambda key: closeness[key]))

    return node

answer_three()
```

Out[4]: 23

Question 4

Assume the restriction on the voucher's travel distance is still removed, but now a competitor has developed a strategy to remove a person from the network in order to disrupt the distribution of your company's voucher. Your competitor is specifically targeting people who are often bridges of information flow between other pairs of people. Identify the single riskiest person to be removed under your competitor's strategy?

This function should return an integer, the name of the node.

In [5]:
```
def answer_four():

    # Your Code Here
    betweenness = nx.betweenness_centrality(G1)
    node = max(betweenness.keys(), key=(lambda key: betweenness[key]))

    return node

answer_four()
```

Out[5]: 333

Part 2

`G2` is a directed network of political blogs, where nodes correspond to a blog and edges correspond to links between blogs. Use your knowledge of PageRank and HITS to answer Questions 5-9.

In [6]:
```
G2 = nx.read_gml('blogs.gml')
```

Question 5

Apply the Scaled Page Rank Algorithm to this network. Find the Page Rank of node 'realclearpolitics.com' with damping value 0.85.

This function should return a float.

In [7]:
```
def answer_five():

    # Your Code Here
    rank = nx.pagerank(G2, alpha=.85)['realclearpolitics.com']

    return rank

answer_five()
```

Out[7]:
```
0.004636694781649093
```

Question 6

Apply the Scaled Page Rank Algorithm to this network with damping value 0.85. Find the 5 nodes with highest Page Rank.

This function should return a list of the top 5 blogs in desending order of Page Rank.

In [8]:
```
def answer_six():

    # Your Code Here
    rank = nx.pagerank(G2, alpha=.85)
    nodes = sorted(rank.keys(), key=lambda key:rank[key], reverse=True)[:5]

    return nodes

answer_six()
```

Out[8]:
```
['dailykos.com',
 'atrios.blogspot.com',
 'instapundit.com',
 'blogsforbush.com',
 'talkingpointsmemo.com']
```

Question 7

Apply the HITS Algorithm to the network to find the hub and authority scores of node 'realclearpolitics.com'.

Your result should return a tuple of floats `(hub_score, authority_score)`.

In [9]:
```
def answer_seven():

    # Your Code Here
    hits = nx.hits(G2)
    node = 'realclearpolitics.com'
    hub = hits[0][node]
    authority = hits[1][node]

    return hub, authority

answer_seven()
```

Out[9]:
```
(0.000324355614091667, 0.003918957645699856)
```

Question 8

Apply the HITS Algorithm to this network to find the 5 nodes with highest hub scores.

This function should return a list of the top 5 blogs in desending order of hub scores.

In [10]:
```
def answer_eight():

    # Your Code Here
    hits = nx.hits(G2)
    hubs = hits[0]
    blogs = sorted(hubs.keys(), key=lambda key:hubs[key], reverse=True)[:5]

    return blogs

answer_eight()
```

Out[10]:
```
['politicalstrategy.org',
 'madkane.com/notable.html',
 'liberaloasis.com',
 'stagefour.typepad.com/commonprejudice',
 'bodyandsoul.typepad.com']
```

Question 9

Apply the HITS Algorithm to this network to find the 5 nodes with highest authority scores.

This function should return a list of the top 5 blogs in desending order of authority scores.

In [11]:
```
def answer_nine():

    # Your Code Here
    hits = nx.hits(G2)
    authority = hits[1]
    blogs = sorted(authority.keys(), key=lambda key:authority[key], reverse=True)[:5]

    return blogs

answer_nine()
```

Out[11]:
```
['dailykos.com',
 'talkingpointsmemo.com',
 'atrios.blogspot.com',
 'washingtonmonthly.com',
 'talkleft.com']
```

In [ ]:
```

```