RACHEL MCCREARY
CS 432 ASSIGNMENT 6 REPORT

PART 1

I decided to use my friend Tyler Shaw's Twitter account(@AtHeartEngineer) and re-ran my program from Assignment 4, omitting the follower counts:

```
rachel@Rachel:~$ cd Desktop/CS_432/A6
rachel@Rachel:~/Desktop/CS_432/A6$ python part1_w_tokens.py
 > followers.txt
rachel@Rachel:~/Desktop/CS_432/A6$
```

followers.txt (~/Desktop/CS_432/A6) - gedit

```
Followers: 503
Manoj_Chandra11
DevAhmedhosni
dineshpaladhi
noumenal_woman
activehistory
rdayMedia
ErichdotPy
DanKerchner
risse691
IuliaCristina12
fromADMwithlove
AndreeaCorinaM2
PSleeman
Shaaban_Migo
KevinClemmons
```

part1_w_tokens.py

```python
import tweepy
from tweepy import OAuthHandler
import time

access_token = "XXXXXX"
access_token_secret = "XXXXXX"
consumer_key = "XXXXXX"
consumer_secret = "XXXXXX"

if __name__ == '__main__':

    auth = tweepy.OAuthHandler(consumer_key, consumer_secret)
    auth.set_access_token(access_token, access_token_secret)
    api = tweepy.API(auth)

    data = api.get_user('phonedude_mln')
    print 'Followers: ' + str(data.followers_count)

    for user in tweepy.Cursor(api.followers, screen_name="phonedude_mln", count = 200).items():
        print user.screen_name
```

Then I used "show_friendship" to find who followed each other within that list, putting any pairs which contained "True" into relationships.txt:

```
rachel@Rachel:~/Desktop/CS_432/A6$ python part1_w_tokens.py
False jimmyatmovement
False sdxcentral


False Margueritaig53
False sdxcentral


False strgcmarketing
False sdxcentral


False ChristeRachel
False sdxcentral


False lDX75MukIi28wmo
False sdxcentral


False Bubbleahsaturn
False sdxcentral
```

relationships.txt (~/Desktop/CS_432/A6) - gedit

```
True OctantSoftware
False sdxcentral

False toomuchsawdust
True jimmyatmovement

False carneyval
True jimmyatmovement

False wrestang
True jimmyatmovement

False TheDaycations
True jimmyatmovement

False kissanddrink
True jimmyatmovement

False DaveEEEEEE
True jimmyatmovement
```

~/Desktop/CS_432/A6/part1_w_tokens.py - Sublime Text (UNREGISTERED)

part1_w_tokens.py | index.html | relationships.json

```python
if __name__ == '__main__':

    auth = tweepy.OAuthHandler(consumer_key, consumer_secret)
    auth.set_access_token(access_token, access_token_secret)
    api = tweepy.API(auth, wait_on_rate_limit=True, wait_on_rate_limit_notify=True)

    #for user in tweepy.Cursor(api.followers, screen_name="AtHeartEngineer", count = 200).items():
        #print user.screen_name

    people = []
    followers = []
    with open('tylers_followers.txt') as f:
        for line in f:
            people.append(line)
            followers.append(line)

    for person in people:
        for follower in followers:
            if person == follower:
                pass
            else:
                relationship = api.show_friendship(source_screen_name=person, target_screen_name=follower
                source, target = relationship
                print str(source.followed_by) + " " + target.screen_name
                print str(target.followed_by) + " " + source.screen_name
                print "\n"
```

Line 36, Column 27 | Tab Size: 4

Then I put this information into a JSON file to be used for the D3 graph:

rachel@Rachel: ~/Desktop/CS_432/A6

```
rachel@Rachel:~$ cd Desktop/CS_432/A6
rachel@Rachel:~/Desktop/CS_432/A6$ python json-builder.py > relationships.json
rachel@Rachel:~/Desktop/CS_432/A6$
```

relationships.json (~/Desktop/CS_432/A6) - gedit

full_output.txt | temp_output.txt | relationships.txt | relationships.json

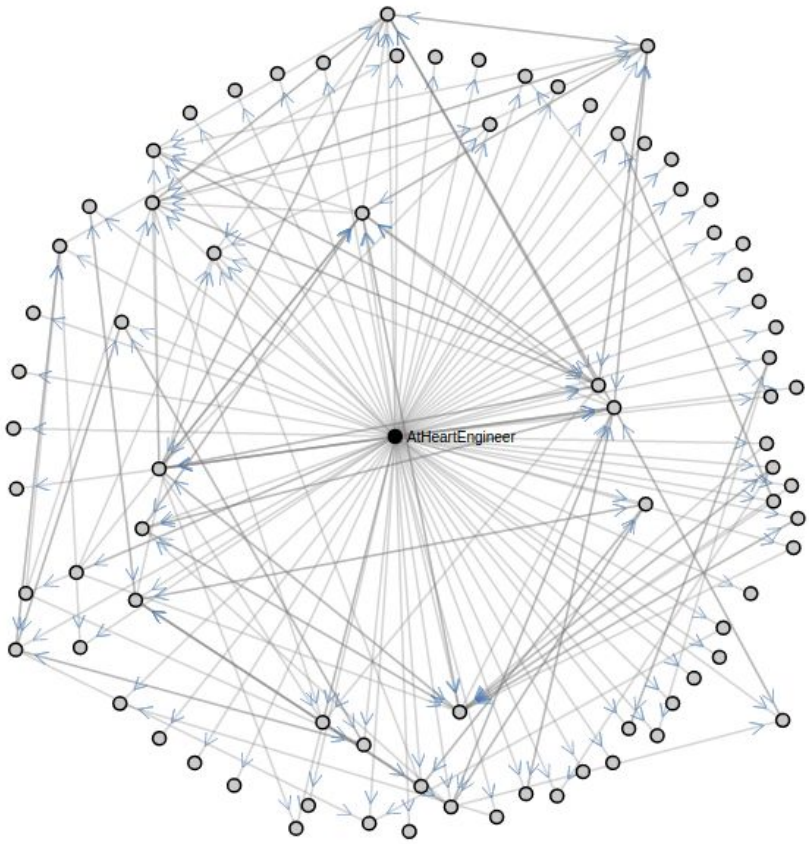
```
        {
            "name": "bashpr0mpt"
        },
        {
            "name": "stephrusso7"
        },
        {
            "name": "joshombroso"
        },
        {
            "name": "BethanyFerrell"
        },
        {
            "name": "FreeStateNH"
        }
    ],
    "links": [
        {
            "source": 0,
            "target": 1
        },
        {
            "source": 0,
            "target": 2
        },
        {
            "source": 0,
            "target": 3
        },
        {
            "source": 0,
            "target": 4
        },
        {
            "source": 0,
            "target": 5
        },
        {
```

JSON | Tab Width: 8 | Ln 1018, Col 2 | INS

index.html | json-builder.py

```python
import collections
import json

followers = collections.OrderedDict()
followers['AtHeartEngineer'] = []
with open('tylers_followers.txt') as f:
    for i, line in enumerate(f):
        line = line.strip()
        followers[line] = []
        followers['AtHeartEngineer'].append(line)
        # node_ids[line] = i + 1
node_ids = {}
for i, follower in enumerate(followers):
    node_ids[follower] = i
#print (followers)
source_target_pairs = []
current_pair = []
i = 0
with open('relationships.txt') as f:
    for line in f:
        if not line.startswith("#") and not line.strip() == '':
            i += 1
            current_pair.append(line)
            if (i % 2 == 0):
                source_target_pairs.append(current_pair)
                current_pair = []

for source_target in source_target_pairs:
    source_split = source_target[0].split()
    target_split = source_target[1].split()
    source_user = source_split[1].strip()
    target_user = target_split[1].strip()
    if source_split[0] == 'True':
        if source_user not in followers[target_user]:
            followers[target_user].append(source_user)
    if target_split[0] == 'True':
        if target_user not in followers[source_user]:
            followers[source_user].append(target_user)

d3_json = {"nodes": [], "links": []}
for follower in followers:
    #print ('follower', follower)
    d3_json['nodes'].append({'name':follower})
    source = node_ids[follower]
    for t in followers[follower]:
        target = node_ids[t]
        d3_json['links'].append({'source': source, 'target': target})

#print (json.dumps(followers, indent=4, sort_keys=True))
#print ('\n\n')
print (json.dumps(d3_json, indent=4, sort_keys=False))
```

While looking through D3 examples, I stumbled across [this version]() of the force-directed graph, which I liked and decided to use. I stumbled for a couple of days figuring out how to run the website's example, but eventually got it to work. I used [this]() website for help adding arrows to indicate who's following who. My final D3 graph can be found here (mouse-over for labels): http://www.cs.odu.edu/~rmccrear/index.html



PART 2

I used tweepy to get the followers' real names from Twitter:

```python

if __name__ == '__main__':

    auth = tweepy.OAuthHandler(consumer_key, consumer_secret)
    auth.set_access_token(access_token, access_token_secret)
    api = tweepy.API(auth, wait_on_rate_limit=True, wait_on_rate_limit_notify=True)

    with open('tylers_followers.txt') as f:
        for line in f:
            user = line.strip()
            real_name = api.get_user(screen_name=user)
            print (user, real_name.name)
            # print (user, real_name['name'])
```

```
('vonKurajan', u'KoOoOoOo')
('TheDaycations', u'The Daycations')
('starlyfe2014', u'STARLYFE')
('usarms', u'US Armorment')
('kissanddrink', u'Carlie Steiner')
('4PartyRVA', u'PartyRVA')
('TeaTimeDC', u'Tea Time DC')
('ogsystems', u'OGSystems (OGS)')
('The_Bagel_Guy', u'Clarkie')
('mjliuzzo', u'Michael Liuzzo')
('Raihan_Islam', u'Raihan Islam')
('DaveEEEEEE', u'Dave Edwards')
('CompyKelly', u'CompyKelly')
```

I went through and removed organizations from the list of real names, then removed extra characters and extracted the first names only:

real_names.txt | genderize.py

```python

with open('real_names.txt') as f:
    for line in f:
        line = line.replace("(", "")
        line = line.replace(")", "")
        name = line.split(", ")[1]
        # Removing the "u" and " ' " and extra whitespace at end
        name = name[2:-2]
        # Getting only the first names
        name = name.split()[0]
        print (name)

```

```
Jimmy
Rachel
stephen
Robbie
Terry
Will
Chuck
Brooke
Rob
Nate
Andrew
Lisa
Matt
NANA
Melissa
mohayed
Nicholas
David
ALEXIS
Noreen
Brendon
Ymani
Tanner
Christina
Patrick
Afriyie
Patrick
Sean
Lauren
```

Then, using genderize.io, I mapped the gender of the users to their usernames. I had to keep the real names mapped to the usernames in the process:

real_names.txt | genderize.py | json-builder.py

```python
import urllib.request
import json

genders = {}

with open('real_names.txt') as f:
    for line in f:
        line = line.replace("(", "")
        line = line.replace(")", "")
        split = line.split(", ")
        username = split[0]
        username = username[1:-1]
        name = split[1]
        # Removing the "u" and " ' " and extra whitespace at end
        name = name[2:-2]
        # Getting only the first names
        name = name.split()[0]
        #print (username, name)
        string = "https://api.genderize.io/?name=" + name
        response = urllib.request.urlopen(string).read()
        #print (response)
        response = response.decode("utf-8")
        json_object = json.loads(response)
        gender = json_object['gender']
        genders[username] = gender

print (json.dumps(genders, indent=4, sort_keys=False))
```


```
{
    "cfullerton133": "male",
    "75_christina": "female",
    "Ben_Shear_": "male",
    "DaveEEEEEE": "male",
    "CollinsYmani": null,
    "wj_yeager": "male",
    "stephrusso7": "female",
    "Its_Drewdog": "male",
    "fivecoins23": "female",
    "Raihan_Islam": "female",
    "tannerwood85": "male",
    "SeanExposure": "male",
```

Then, I used this information to prove that there was no gender homophily in Tyler's twitter followers:

genderize.py | genders.json | relationships.json | json-builder.py | bookmarks

```python
q = 0

for person in genders:
    if genders[person] == "male":
        p += 1
    elif genders[person] == "female":
        q += 1

p /= len(genders)
q /= len(genders)

print ("p = " , p)
print ("q = " , q)

threshold = 2*p*q
print ("2pq = " , threshold)

edges = d3_json['links']
nodes = d3_json['nodes']

edge_count = 0
heterogeneous_count = 0
for edge in edges:
    source_number = edge['source']
    target_number = edge['target']
    source_username = nodes[source_number]['name']
    target_username = nodes[target_number]['name']
    if source_username in genders and target_username in genders:
        source_gender = genders[source_username]
        target_gender = genders[target_username]
        edge_count += 1
        if source_gender != target_gender and source_gender is not N
            heterogeneous_count += 1
```

```
p =  0.6041666666666666
q =  0.3541666666666667
2pq =  0.4279513888888889
ratio =  0.5208333333333334
[Finished in 0.1s]
```