Assignment #2 (10 points):Due March 21

1. Select a ‘starting point,’ i.e. a user on Twitter, which could be yourself or somebody else.

2. Retrieve his/her friends, which should be a list of id’s, and followers, which is another list of id’s, perhaps using the **get_friends_followers_id()** function from the Cookbook, or your own program if you prefer. **Note:** When you use get_friends_followers_id() or its equivalent, you are allowed to set the maximum number of friends and followers to be 5000 (but no less) in order to save API calls, and hence save your waiting time due to time limits.

3. Use those 2 lists from Step 2 to find **reciprocal friends**, which is yet another list of id’s. (The definition of ‘reciprocal friends’ can be found in my slides.) These are the **distance-1 friends**.

Assignment #2 (10 points):Due March 21

4. From that list of reciprocal friends, select **5** most popular friends, as determined by their **followers_count** in their user profiles. (I suggest you use the **get_user_profile()** function from the Cookbook to retrieve the user profiles of the reciprocal friends.)
5. Repeat this process (Steps 2, 3 & 4) for each of the distance-1 friends, then distance-2 friends, and so on and so forth, using a **crawler**, until you have gathered at least **100** users/nodes for your social network. **Note:** I suggest you modify the crawler (**crawl_followers()**) function from the Cookbook or my simplified crawler to do this. However, please note that either one of these 2 crawlers retrieves only followers. You need to modify it to get both followers and friends, in order to compute the reciprocal friends .

Assignment #2 (10 points):Due March 21

6. Create a social network based on the results (nodes and edges) from Step 5, using the **Networkx** package, adding all the nodes and edges. (**Note:** you don't have to wait till crawling is all done to create your social network. You may crawl and build your social network at the same time. It's up to you to decide how you want to write your program.)

7. Calculate the **diameter** and **average distance** of your network, using certain built-in functions provided by Networkx (in 3.23 Distance Measures & 3.51 Shortest Paths. (**Note:** it's your job to figure our which functions you should use to achieve your goals.) Or, you could write your own functions if you prefer.

Assignment #2 (10 points):Due March 21

Deliverables

a) **Program output**: Your program should output your network size (in terms of numbers of nodes & edges), average distance & diameter. Save program output to a file and submit the file. if possible, please also generate an image file for your social network using matplotlib, as I demonstrated in class.

b) **Your program source code with comments describing each class, function and program segment**. Make sure it runs. Also indicate which part is your own code. (**Note:** Reusing code from the textbook/cookbook, my slides, and any python libraries is allowed, but you must cite your source.)

c) Put your program output file(s), source code (with comments), and any data file in **one** folder, zip it and submit the zipped folder via Blackboard.

Grading Rubrics

- Program not running: -3
- Program crashed: -2 or -1 depending on when
- Fewer than 100 nodes collected: less than 90: -0.5; additional -0.5 per 10 nodes less than 90
- Diameter not (correctly) calculated: -1
- Average distance not (correctly) calculated: -1
- Network not created: -3
- Network not created correctly: -2
- Reciprocal friends not done correctly: -2
- Top 5 reciprocal friends not done correctly: -2
- No crawler: -3
- Crawler not done correctly: -2
- Low quality code, or no comments: -1
- Other unforeseen issues: depending on the severity