CPS 470/570: Computer Networks and Security

Programming Assignment #2, 100 pts, 3 weeks

This is an individual assignment! NO team work.

No late submission will be accepted

Receive 5 bonus points if turn in the complete work **without errors** at least one day before deadline

Receive an *F* for this course if any academic dishonesty occurs

---

1. Purpose

This project crawls twitter social networks and conducts basic data analytics.

2. Description

2.1. Twitter basics

Twitter users post tweets (i.e., "status updates") – text messages of up to 140 characters, which can also contain images, video media, or links to other online resources – and interact with others by following or responding to their messages. The data structure of a tweet contains (in addition to the text) metadata such as a username and a user screen name, numerical identifiers, a timestamp, the language of the text, the location of the user, and/or the ways in which the status update references other messages or users on the platform.

The types of information that we can extract from twitter:

- Information about a user
- User's Followers or Friends
- Tweets published by a user
- Search results on Twitter
- Places & Geo

It offers three Twitter APIs:

- **REST APIs**: Provide programmatic access to read and write Twitter data
- **Streaming APIs**: Once a request for information is made, the Streaming APIs provide a continuous stream of updates with no further input from the user
- **Search API**: The Twitter Search API searches against a sampling of recent Tweets published in the past 7 days

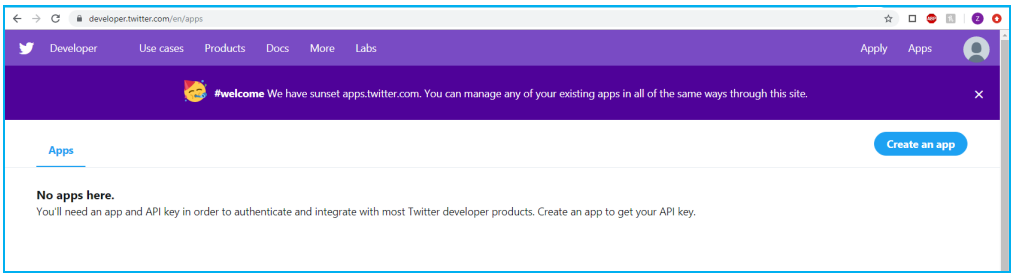
Tweets are delivered in the format JSON (JavaScript Object Notation between web servers and clients) that can be processed using a number of programming languages and software packages. There are many existing libraries in common programming languages to facilitate interaction with the Twitter APIs, e.g., tweepy (for python) and twitter4j (for java). While you are welcome to use twitter4j, this handout shows instructions for using tweenpy. The main steps include:

- **Apply for a developer account at twitter**
- **Install tweepy**
- **Create a python project and write code for crawling!**

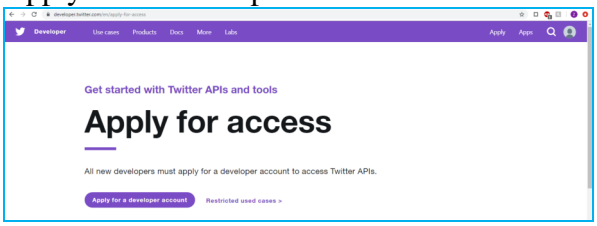
Each will be explained in details in the following.

2.2. Apply for a developer account

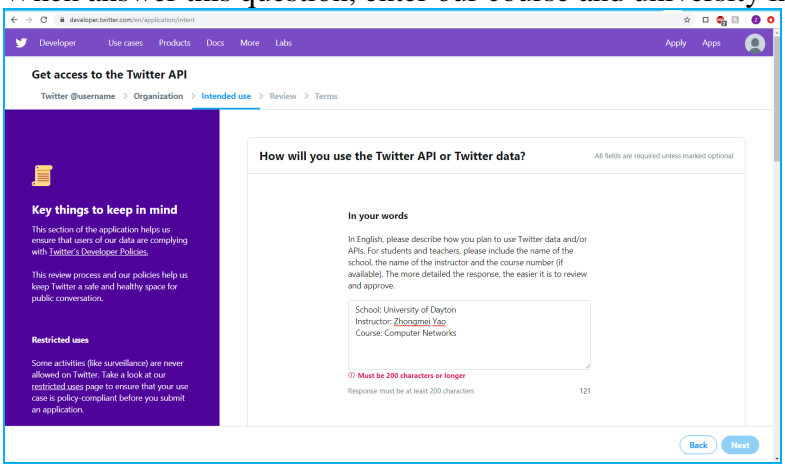
Go to https://developer.twitter.com/en/apps. **Create a twitter account** and then click **Create an app**.



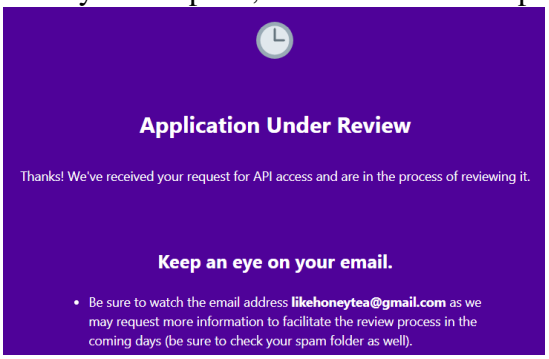
Apply for a Developer account:



When answer this question, enter our course and university information:



Once you complete, wait for the review process:



You might receive an email from twitter asking more information. The following is how I replied to the email (shown in blue color):

- The core use case, intent, or business purpose for your use of the Twitter APIs.

The core purpose is to gain learning experiences in the course CPS 470 Computer Networks at University of Dayton.

- If you intend to **analyze** Tweets, Twitter users, or their content, share details about the analyses you plan to conduct, and the methods or techniques.

In this course project, I may do simple analysis such as how many twitter accounts, the average number of followers per account. Apply machine learning methods to see if an account is a bot account or not.

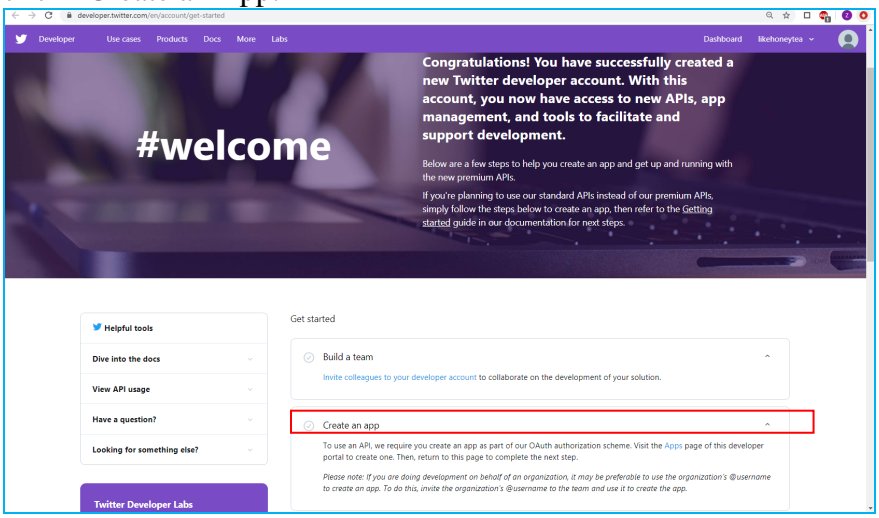
- If your use involves **Tweeting, Retweeting, or liking content, share how you'll interact with Twitter accounts, or their content.**

I will not use tweeting, retweeting, or liking contents.

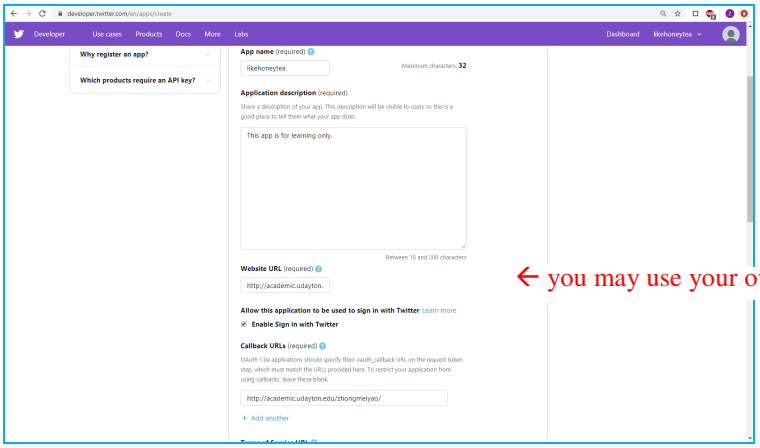
- If you'll display Twitter content off of Twitter, explain how, and where, Tweets and Twitter content will be displayed with your product or service, including whether Tweets and Twitter content will be **displayed at row level, or aggregated.**

The main purpose is academic use. If results are good enough, I may submit the results to a conference such as IEEE INFOCOM. Tweets and twitter content will NOT be displayed. NO user ID will be displayed. NO privacy information will be displayed.

Twitter then approved my Developer account quickly. Click the approval link in the email. Now click "Create an App:"

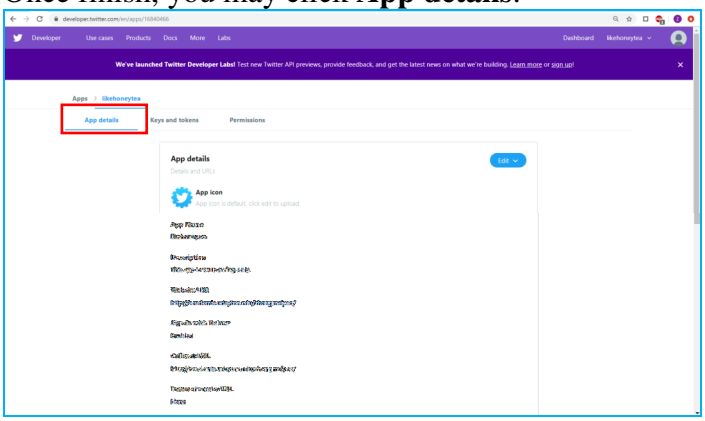


When enter required information, you may provide similar answers as follows:

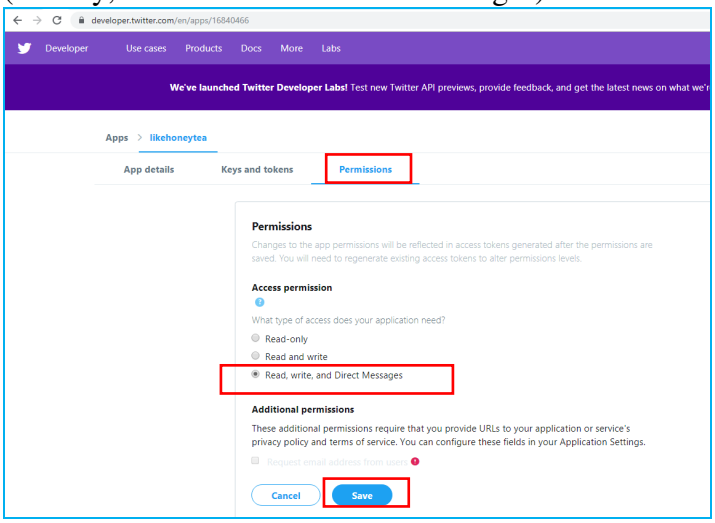


← you may use your own url if you have one

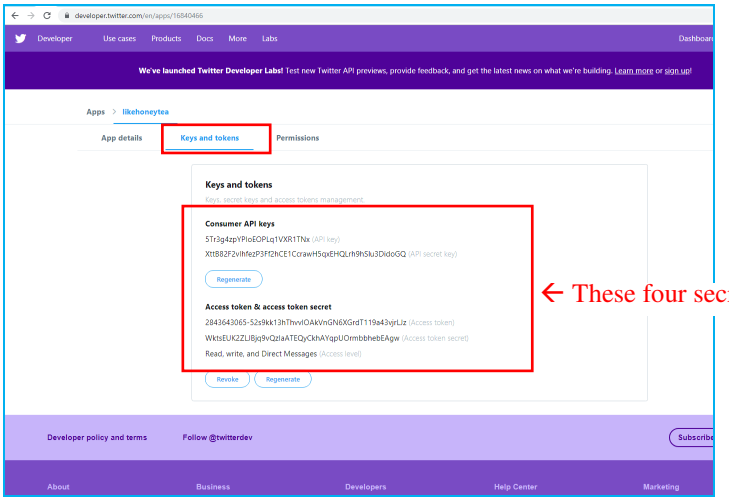
Once finish, you may click **App details**:



Click the **Permissions tab** and configure your application with the permission level you need (namely, read-write-with-direct messages) and then save:



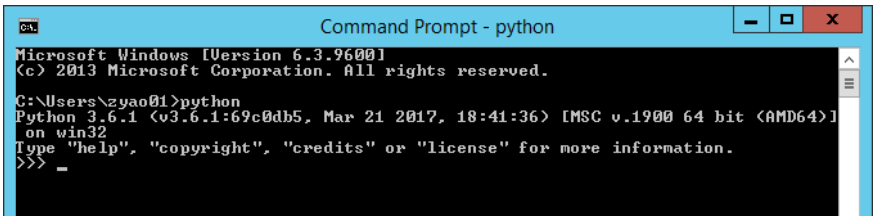
Click **Keys and tokens**. Obtain the indicated access token and access token secret from the screen and use them in your application.



← These four secrets will be used in your code

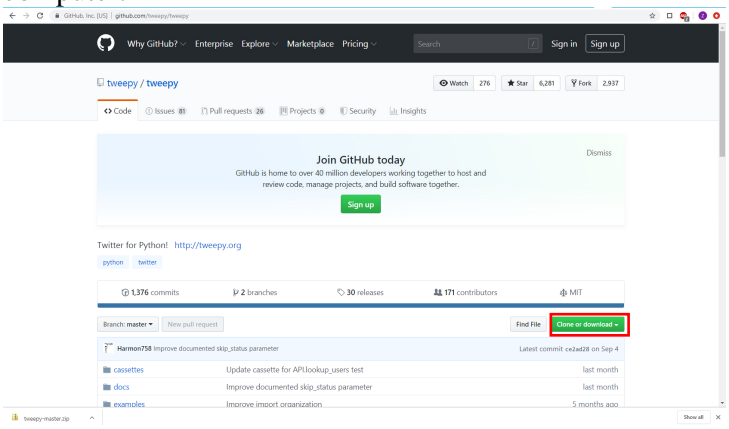
2.3. Install tweepy

First, make sure that you have installed python on your computer. In command line, after enter python, you should see Python version:

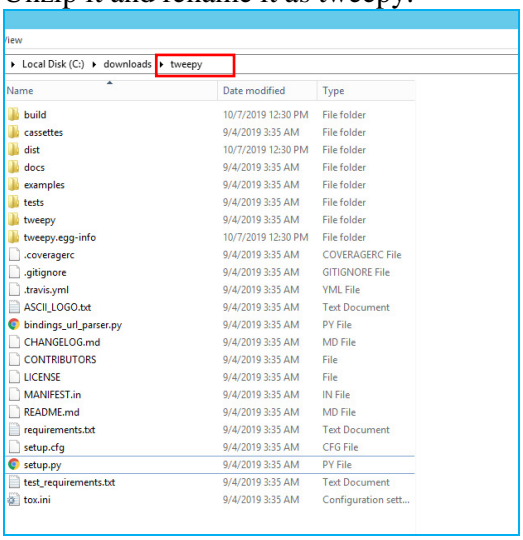


The above shows I installed python 3.6 on my computer. Enter `exit()` to exit. If you have not installed python, go to isidore/resources to see instructions on how to do python and pycharm.

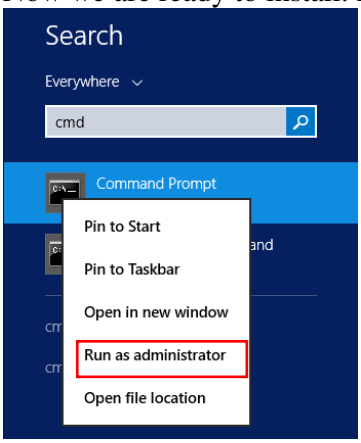
For windows users, download tweepy from https://github.com/tweepy/tweepy and save it on your computer.



Unzip it and rename it as tweepy:



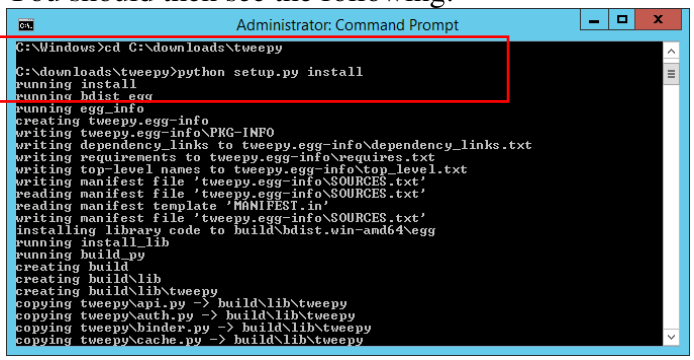
Now we are ready to install. Right click on **Command Prompt** and choose "Run as administrator:"



Then enter the following commands to install the package:

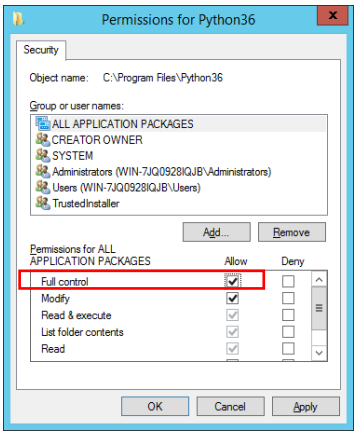
```
cd C:\downloads\tweepy
python setup.py install
```

You should then see the following:



Just in case, if you were unable to install tweepy, do the following:

```
Find the python root directory (in C disk):
 --> right click on python directory --> properties --> security --> edit
 --> give users Full Control--> Apply, OK
```



Wait the process finished. Then install the tweepy package in the command window.

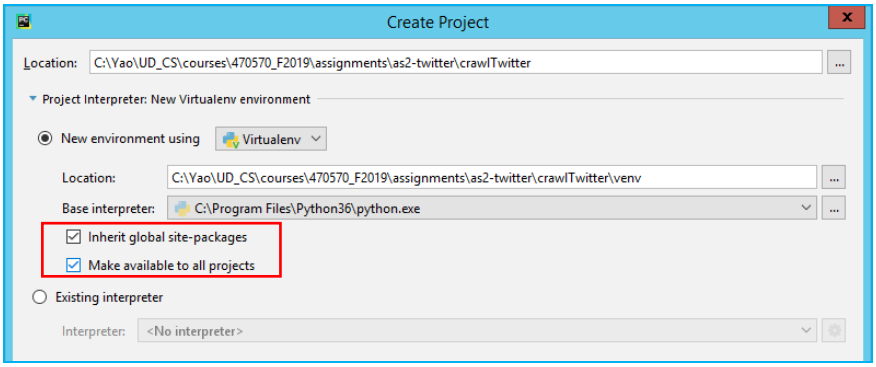
For macOS/Linux users, you may enter the following to download and then install tweepy:

```
$ git clone https://github.com/tweepy/tweepy.git
$ pip install tweepy
```

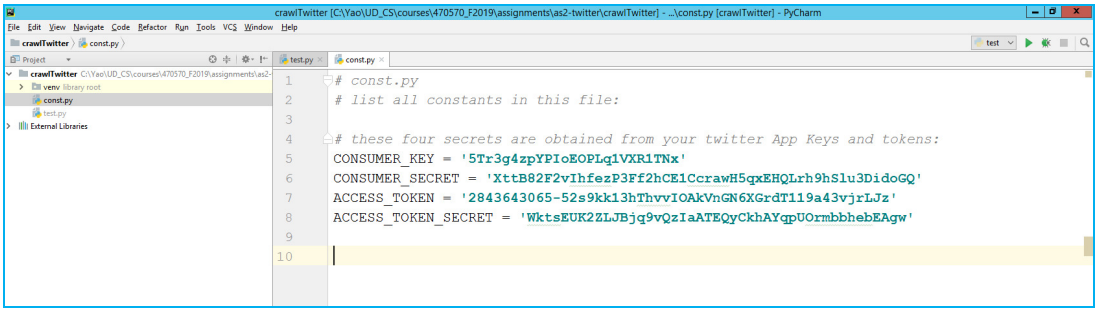
Pip is the package installer for python. Note that pip is already installed if you are using Python 2 >=2.7.9 or Python 3 >=3.4 downloaded from python.org.

2.4. Create a python project for crawling twitter

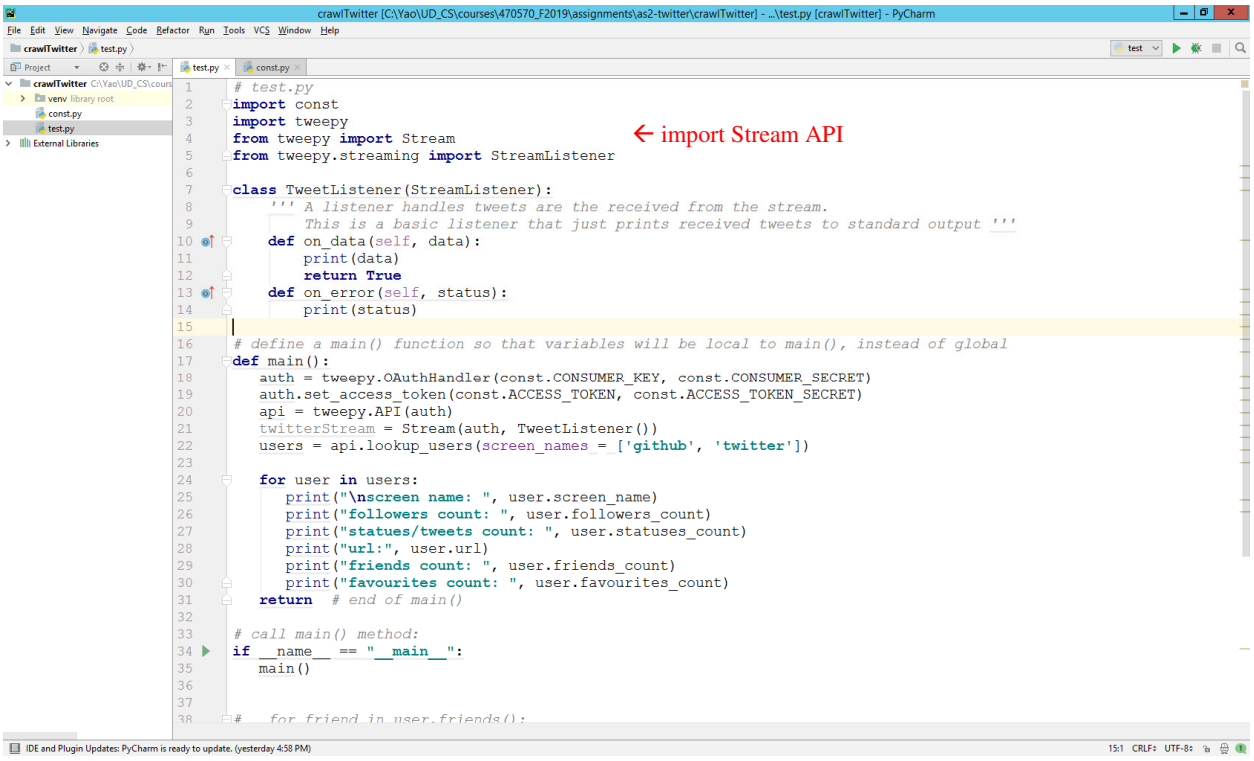
Now launch PyCharm. When creating a new project, make sure that you click “Inherit global site-packages” and “Make available to all projects” (otherwise, the tweepy package is not accessible in this project):



Right click on Project name (e.g., “crawlTwitter”) and add a new python file (e.g., “const.py”). The **const.py** file contains constants only:



Right click on Project name and add another new python file, e.g., test.py. Enter python code and enjoy!




```python
# test.py
import const
import tweepy
from tweepy import Stream
from tweepy.streaming import StreamListener

class TweetListener(StreamListener):
    ''' A listener handles tweets are the received from the stream.
        This is a basic listener that just prints received tweets to standard output '''
    def on_data(self, data):
        print(data)
        return True
    def on_error(self, status):
        print(status)

# define a main() function so that variables will be local to main(), instead of global
def main():
    auth = tweepy.OAuthHandler(const.CONSUMER_KEY, const.CONSUMER_SECRET)
    auth.set_access_token(const.ACCESS_TOKEN, const.ACCESS_TOKEN_SECRET)
    api = tweepy.API(auth)
    twitterStream = Stream(auth, TweetListener())
    users = api.lookup_users(screen_names = ['github', 'twitter'])

    for user in users:
        print("\nscreen name: ", user.screen_name)
        print("followers count: ", user.followers_count)
        print("statues/tweets count: ", user.statuses_count)
        print("url:", user.url)
        print("friends count: ", user.friends_count)
        print("favourites count: ", user.favourites_count)
    return  # end of main()

# call main() method:
if __name__ == "__main__":
    main()
```

Right click on "test.py" and choose **run 'test'**. You should then see information of given users *github* and *twitter*, respectively:

```
C:\Yao\UD_CS\courses\470570_F2019\assignments\as2-twitter\crawlTwitter\venv\Scripts\python.exe C:/Yao/UD_CS/courses/470570_F2019/assig

screen name:  github
followers count:  1857689
statues/tweets count:  4840
url: None
friends count:  306
favourites count:  4387

screen name:  Twitter
followers count:  56694575
statues/tweets count:  12057
url: https://t.co/TAXQpsHa5X
friends count:  103
favourites count:  6345

Process finished with exit code 0
```

Twitter APIs have rate limiting. See details at https://developer.twitter.com/en/docs/basics/rate-limiting.

- The **REST API** rate limits can be found at https://developer.twitter.com/en/docs/basics/rate-limits. For instance, the GET users/ calls are limited to 15 requests every 15 minutes.
- One can connect to the **Streaming API** to form a HTTP request. The Streaming API free-ups rate limits for more use, but it still has rate limiting. Check this link https://developer.twitter.com/en/docs/tweets/filter-realtime/guides/connecting.

You should not exceed the rate limits when doing this project; otherwise, your developer account might be blocked by twitter.

Tweepy has well written documenation at https://tweepy.readthedocs.io/en/latest/index.html. The sample code above is from the website. Your task is to read documentation and address the questions for this assignment.

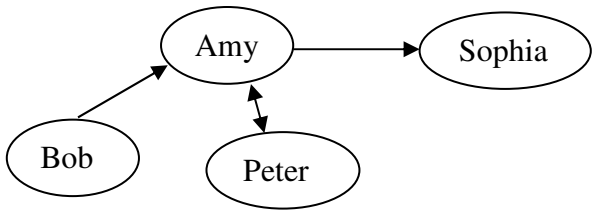
2.5. Report

The report should address the following questions:

Task 1) (15pts) A use profile provides a rich source of information to study twitter users. Given a list of user's screen names, **write a crawler to display** this users' profile information. You should get the following information for any existing twitter user:

```
User name:
Screen name:
User ID:
Location:
User description:
The number of followers:
The number of friends:
The number of tweets (i.e., statuses):
User URL:
```

Task 2) (15pts) There are two types of connection between users: follower and friend. Friendship is bidirectional while following is one direction. In the following figure, Amy and Peter are friends (meaning that they follow each other), Bob is **following** Amy (so Bob is Amy's **follower**), and Amy follows Sophia.



Given a list of user's screen names (any existing names), **write a crawler to collect** the users' social network information (i.e., display friends and the first 20 followers). Note that friends are bidirectional (e.g., Amy and Peter are friends as they follow each other).

Task 3) (30pts) Twitter provides APIs to collect tweets that contain the specified keywords or originate from a given geographic region. The returned objects of the search are in JavaScript Object Notation (JSON). You will exact some fields in JSON. You will look at both Search API and streaming API.

a) (15pts) Write a crawler to collect the first 50 tweets that contain these two keywords: [**`Ohio, weather`**].

b) (15pts) Write a crawler to collect the first 50 tweets that originate from **`Dayton region`**, specified by **`point_radius`**: [**Longitude of the center**, **Latitude of the center,**

radius], where the radius can be up to 25 miles. Any tweet containing a geo point that falls within this region will be matched. Dayton OH geographic information is: Latitude 39.758949, Longitude -84.191605. Note that Google Map takes a slightly different format, i.e., [Latitude, Longitude].

Task 4) (40pts) Use the twitter API for your own idea. The following are some ideas for your reference. You can go to **scholar.google.com** and search papers/books on "twitter" to do brainstorming.

a) Write code to deliver tweets of your interest to your email at 8:00AM every day;

b) Write code to deliver tweets on stock price changes to your email and make suggestions, e.g., buy/sell this stock;

c) Write code to detect fake news;

d) Write code to detect bot accounts (i.e., accounts controlled by attackers);

e) Write code to detect users who need help (e.g., people with mental health issues);

f) **Or** your own idea.

3. Turn In

- Your report that provides answers to the tasks in Section 2.5.
- Source code: You can either submit a single script file that finishes all tasks altogether or individual .py files for each task.
- Submit a README file: The Python version you used to test your code; how to compile/run your source code.

4. References

- Tweepy documenation at https://tweepy.readthedocs.io/en/latest/index.html.
- Developer Website: https://developer.twitter.com/
- API resource documentation: https://developer.twitter.com/en/docs
- Twitter libraries: https://developer.twitter.com/en/docs/developer-utilities/twitter-libraries