Week 4 Assignment: Python

This assignment has separate parts so that you can practice the concepts from this week. You will also practice functional programming. Below, you will see that this assignment has three parts. However, rather than writing three separate .py programs, you will place all parts into the **same program**. So, you will submit ONE .py file. This one program (.py file) will use functions to control which Part is being called and run….

To do this, you will first create a function that allows the user to choose which Part to run. The function will describe (print out) the options to the User and will ask the User to make a selection. Based on the User selection, your code will run one of the three functions. The first function will contain all of Part 1. The second function will contain all of Part 2. The third function will contain all of Part 3 (Part 3 is optional but strongly recommended). Your program should do this in a loop so that the User can choose any part, as many times as they wish, and can decide when to quit. So, you will need to think about how and where to place a loop so that the User can control the flow.

As always – do this in parts. First, just get Part 1 to work on its own. Then, get Part 2 to work on its own. Then Part 3 (optional but recommended). Once all the Parts work on their own, place them into functions so that you can call each part. Next, create the function that controls the other functions and the User input. Finally, add the looping so that the User can choose and choose until they want to quit. Part 3 is optional but highly recommended.

In total, your final .py Python Program will have 3 or 4 functions.

Function 0: User input to choose which function to call next. This loops and allows User to choose and choose until they want to quit.
Function 1: This function will be the code for Part 1 and will complete all requirements of Part 1.
Function 2: This function will be the code for Part 2 and will complete all requirements of Part 2.
Function 3: This function will be the code for Part 3 and will complete all requirements of Part 3. (Part 3 is Optional but recommended - so you may or may not have this function)

Part 1: Web Scraping

1) Write Python code that asks a User to enter a stock ticker (symbol). For example, the User might enter AAPL for Apple Stock, or TWTR for Twitter Stock etc.

2) The program will **directly** scrape https://finance.yahoo.com/ and will return the current price of the stock entered by the User.

3) To do this, you will NOT use an API. You will scrape the site, use regular expressions and data cleaning as needed, and will return the current stock price. This will force you to read in very dirty data and get the value you want out of the data.

4) If you cannot access https://finance.yahoo.com/ you can use any other stock price site.

5) Your Python program will loop and will allow the User to continue to enter stock tickers and to get the current stock price. The User will choose whether to end or continue.

6) Your program will print out the ticker that the User entered and the current price for each ticker that the User chooses until the User quits.

Part 2: Using an API.

Register and get required keys for the AirNow API.

DO NOT USE MY KEYS ☺

RE: https://docs.airnowapi.org/webservices

- You will need to register FIRST – or you will not be able to access or do things.

- You will need to log in.

- You will need a KEY.

1) Write Python code and uses either urllib or requests (I prefer requests). I also recommend using JSON. Use the AirNow API as well as the correct URL and GET/POST.

The AirNow site will generate URL examples that you can use in your code. For example – this is a URL generated by AirNow:

http://www.airnowapi.org/aq/forecast/zipCode/?format=application/json&zipCode=80302&date=2017-06-03&distance=25&API_KEY=D9AA91E7-070D-4221-867C-XXXXXXXXXXXXXX

2) Access the AirNow data for ANY zip code that the **User chooses**. So, your Python code will ask the User to enter a zip code and will then print AND write the results**. The results will be written to Part2_OUTFILE.txt.** For each zipcode the User chooses, you will again print the results (as a clean dataframe) AND will append the results to the Part2_OUTFILE.txt. You can test your code on the following zip codes: 20002, 90210, and 80302.

3) Clean up the data so that your program writes (and prints) the results to a file (OUTFILE.txt) and shows in the results the zipcode, the date, the state, the city, and the AQI results. It is recommended that you create a dataframe of the results. Results should be easy to read and to understand.

4) **The User can enter as many zip codes as they wish and will choose to quit when they wish.**

Remember – this Part if one function in a larger program. So, when the User quits entering zip code, your program does not end. Rather, it returns to Function 0 and asks the User which Part they want to run next or if they want to quit all together. You will need to really think about the "control" in this overall program.

Hint: I have given you a lot of code and tutorial information about this AirNow topic.

RE: http://drgates.georgetown.domains/ANLY500/AirNowPythonCode.txt

DO NOT USE MY API KEY!! I have shared it so you can see what the code really looks like. However, you MUST get and use your own key.

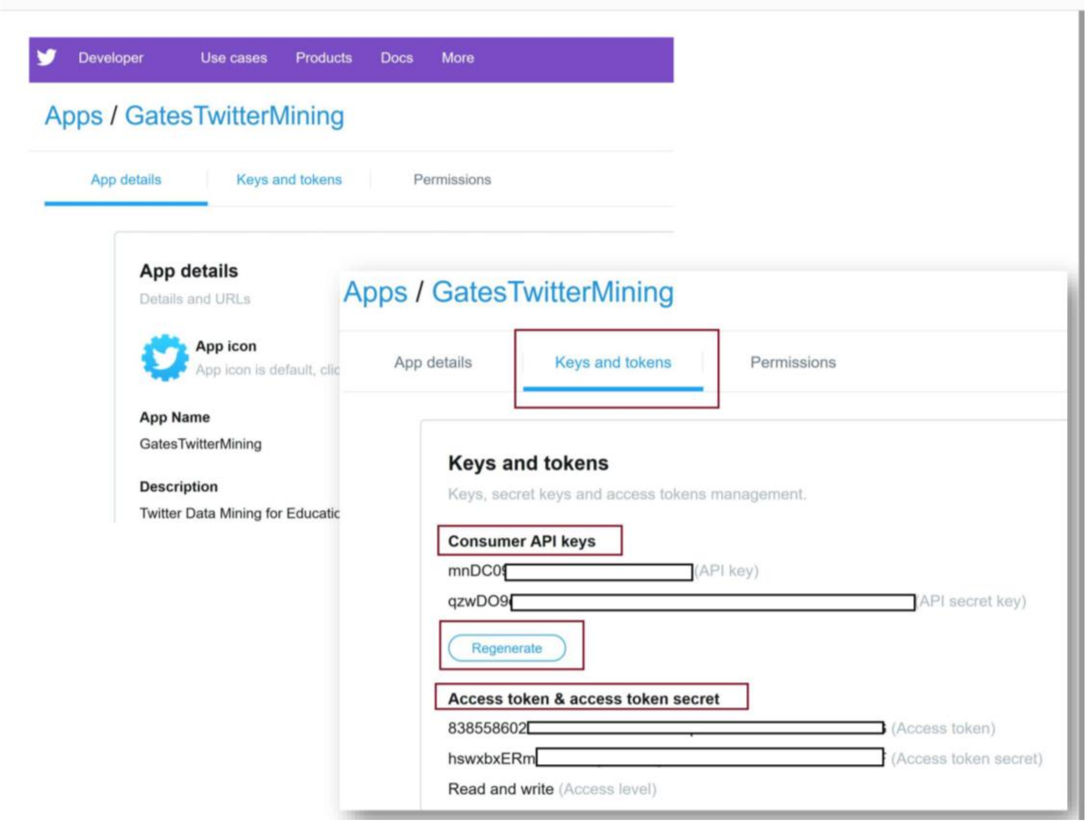
Part 3: Mining Twitter (Optional – this part is optional but highly recommended)

NOTE: This portion is strongly recommended. However, it is optional because some students, depending on location, may have difficulty getting a Twitter Dev Account. I strongly recommend that you try and complete this part.

DO NOT USE MY API KEYS!! I have shared it so you can see what the code really looks like. However, you MUST get and use your own key.

To use Twitter, you must get a Twitter Dev Account. From there, you will find the keys, etc.

**** Use your Georgetown Email and Georgetown info to get a Twitter Dev account.**



Apps / GatesTwitterMining

App details
Details and URLs

App icon
App icon is default, clic

App Name
GatesTwitterMining

Description
Twitter Data Mining for Educatic

Apps / GatesTwitterMining

Keys and tokens
Keys, secret keys and access tokens management.

Consumer API keys
mnDC0[redacted] (API key)
qzwDO9[redacted] API secret key)

Access token & access token secret
838558602[redacted] (Access token)
hswxbxERm[redacted] (Access token secret)
Read and write (Access level)

1) Review the following code:

http://drgates.georgetown.domains/ANLY500/twitter/

2) Run the code for any #YOURCHOICE.

3) Write code to generate a Word Cloud for your #whatever choice.

4) The goal here is to review, understand, and use the code.

Deliverables:

1) This assignment will ask you to write 3 or 4 functions (depending on whether you do Part 3). The first function controls all the other functions and the User choices for which function to run and when to quit. The other two or three functions are each connected to the Parts of this assignment.

2) Create ONE (1) zip folder. Into that folder, place the Python code, your WordCloud pasted into a Word Doc if you completed Part 3, and any and all other output files that are required, etc.

5) Submit the ONE zip folder via email to the TA. Email title: **Assignment 4 Your Name Summer Class**