Exercise — Programming for Data Analysis II
Building a Securities' Dataset — 14th April 2020

1) Scenario of Interest

Assume that, in order to build a dataset of securities' data, we want to retrieve the data related to the quotations of a given list of securities (from publicly available web resources):

- AAPL (Apple)
- GE (General Electric)
- GM (General Motors)
- ...

- A sequential program (like the following one) may accomplish this task.
  - Note. It is possible in Python to retrieve the contents from web ULRs using the requests module.
  - Note. The following program is able to access resources publicly available at URLs like:
    - https://financialmodelingprep.com/api/v3/historical-price-full/AAPL?serietype=line
    - In this particular case the URL contains the AAPL symbol, which represents Apple.
      - By changing the symbol in the URL, you can retrieve data for a different security.
    - The format of the response is JSON (JSON (JavaScript Object Notation), that is a lightweight data-interchange format).
      - What is JSON?
        - https://developers.squarespace.com/what-is-json
        - https://en.wikipedia.org/wiki/JSON
      - A "snippet" of the JSON content retrieved is as follows:

```
{
    "symbol": "AAPL",
    "historical": [
        {
            "date": "2014-06-13",
            "close": 83.6603
        },
        {
            "date": "2014-06-16",
            "close": 84.5035
        },
        {
            "date": "2014-06-17",
            "close": 84.3935
        },
        {
            "date": "2014-06-18",
            "close": 84.4852
        },
```


- A list of symbols may be found here (first column of the table):
  - http://markets.cboe.com/us/equities/market_statistics/listed_symbols/

Symbol	Volume	Matched	Routed	Bid Size	Bid Price	Ask Size	Ask Price	Last Price
VXX	1,402,773	1,381,041	21,732	5,694	$24.45	1,300	$24.46	$24.46
EZU	424,447	421,817	2,630	6,000	$39.04	24,350	$39.05	$39.04
USMV	304,054	302,877	1,177	1,429	$63.50	19,400	$63.51	$63.51
ITB	245,286	243,987	1,299	300	$41.56	3,900	$41.57	$41.56
IEFA	243,892	243,592	300	18,500	$60.73	6,300	$60.74	$60.74
MTUM	127,288	123,885	3,403	500	$119.11	100	$119.13	$119.12
CBOE	123,181	122,220	961	200	$115.15	100	$115.26	$115.21
INDA	110,639	110,638	1	3,900	$32.08	2,000	$32.09	$32.09
JPST	93,569	93,569	0	14,372	$50.45	227,235	$50.46	$50.45

- An implementation (sequential program) to solve our data retrieval problem is as follows:

```
################################## CODE ##################################
import requests
import time

symbols = ['AAPL', 'GE', 'GM']

urls = []
for symbol in symbols:
  urls.append( 'https://financialmodelingprep.com/api/v3/historical-price-full/'
            + symbol +
            '?serietype=line')
start = time.time()
for url in urls:
  try:
     response = requests.get(url)
     # If the response was successful, no Exception will be raised
     response.raise_for_status()
     print(response.text)
  except Exception as exception:
     print('An exception occurred: ' + str(exception))
  else:
     print('Success!')

end = time.time()
print('Took %.3f seconds' % (end - start))
################################## CODE ##################################
```

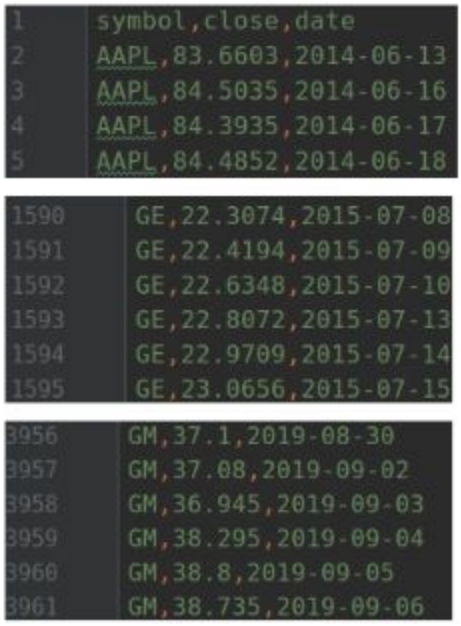
2) Goals

a) Modify the program to store the results in various .json files (a file for each response).

b) Modify the program to achieve concurrency using Threads.
   i) Do you get a faster (or slower) solution? Evaluate the execution time difference (sequential vs threading).

c) Run the program with the input given by a list of 10 securities (what securities? student's choice - for instance, securities belonging to the same macro sector, the same index, …, or without any apparent relationship).

d) Write a python program able to load the contents of all the downloaded files and to merge all the contents in a unique .csv file, after setting the proper header (symbol, close, date). The final result should be as follows:



e) Load the data in a Python Dataframe and extract some information (of your choice).

f) Write a 2 page report document explaining your choices and your code.