Quest 3

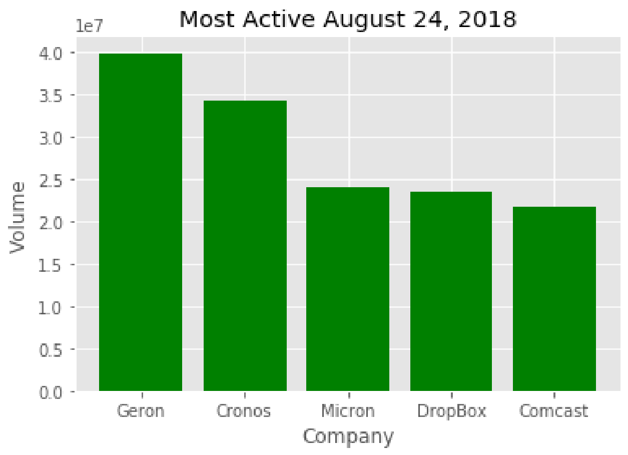
Due: Sunday Sep 9

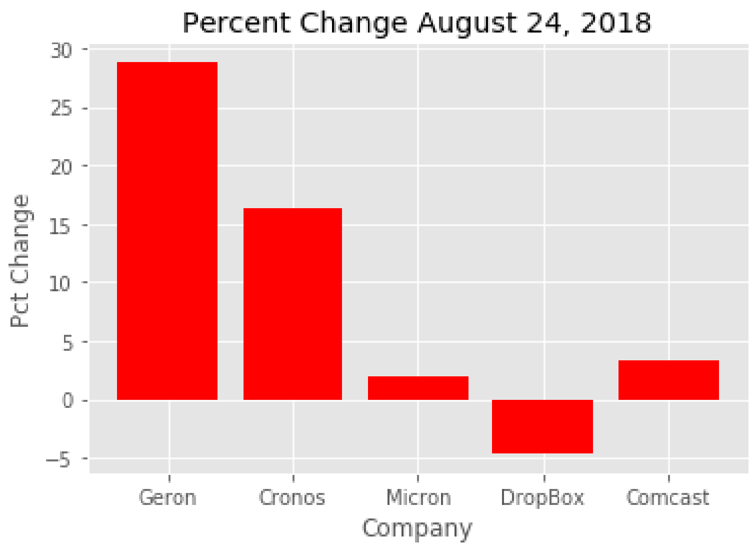
Part A. Beautiful Soup

Use Beautiful Soup to scrape the web page at: https://www.nasdaq.com/quotes/stock-quotes.aspx

This page shows the most active stocks for the day. Wait until the end of a day (any day) and scrape the page for the date, the most active stocks, their volume and percent change in value.

Based on the data you retrieve, use Matplotlib to draw two graphs of the form:





Add to Canvas Pdf: your code and the two graphs

Part B. Timing Comparison

In Quest1 you were asked to write the following code without using the Counter class.

```
def wordsInStringToDictWordCount(s):
    """ return a dict of words in string and count
    >>> wordsInStringToDictWordCount('foo bar   bar') ->
             {'foo':1, 'bar':2}
    >>> wordsInStringToDictWordCount('') -> {}
    constraint: MAY NOT USE: Counter
    """
    return {"foo": 2, "bar": 1}
```

- Your task is to compare the performance of your code vs. an implementation that uses the Counter class and use pyPlot to display the results.
- Use timeit.timeit to compute the time for the two functions.
- Create a string that contains 100,000 words. You may extract from a web page or create the string using a random number generator.

Plot your timing results using a PyPlot bar graph.

HOWTO: calling timeit from within Python code – timeit runs as external program so you must import any functions you wish to time, as in:

```
import timeit
def foo(k):
    sum = 0
    for i in range(k):
        sum = sum + i
    return sum

mynumber = 10
mytime = timeit.timeit('foo(10000)',
                        'from __main__ import foo', mynumber=10 )
print (mynumber / 10)
```

Add to Canvas PDF:

- python code (remove above docstring)
- graph of comparison

Part C. PyTest for Regex

Write a function using regexes that tests whether a string is a valid email address. Here we define a valid email address as having four parts:

1. local part: any characters in the set A-Z,a-z,0-9 or dot '.' where the dot cannot be the first or the last character
2. the '@' symbol
3. domain1: any character in A-Z, a-z, or underscore '_'   provided the underscore is not the first or last character
4. domain2: one of .com, .org, .edu

DO:

- Write PyTest code that tests your function against a collection of valid and invalid email addresses.
- Add to Canvas PDF: your code, your testcode and the results.

Part D. CSV Files

- Download the file worldcup.csv from Canvas
- Use the Python csv module to extract the data and create json from the csv with one json object for each row of data.
- Write code to display the json for the world cups with the 5 greatest total goals scored
- Extract the year total goals scored and use PyPlot to plot a bar chart of the goals (y-axis) vs year (x-axis)

Add to Canvas PDF: your code, 5 json objects, bar chart

Submit to Canvas:

- A pdf organized by section