CS 620–Introduction to Data Science and Analytics, HW3

You have just been hired as an analyst for an investment firm. Your first assignment is to analyze data for stocks in the S&P 500. The S&P 500 is a stock index that contains the 500 largest publicly traded companies.

You have been given two sources of data to work with. The first is an XML file that contains the Symbol (ticker), company name, sector, and industry for every stock in the S&P 500, as of summer 2016.
The second is a CSV file that contains pricing information for stocks in the S&P 500 between August 2009 and August 2010. There is one row in the CSV file for every stock, on every date that the market was open. Each row contains the date as a string, the stock's ticker, the day's opening price, the day's high price, the day's low price, the day's closing price, and the volume traded that day.

Use the provided files SP500_ind.csv and SP500_symbols.xml and the starter code.
Implement using python and associated libraries (NumPy, Pandas etc.,).

- Complete the following functions in the starter code.

```
def ticker_find(xml_root, ticker):
        """This function takes in the root of the xml dataset and a
        Symbol (ticker).  Return the name of the ticker. If the name is
        not available in the given SP500_symbols.xml then this function will assign
        "No data in SP500" for the ticker name
        Ex: for ticker "A", the function returns Agilent Technologies Inc
        """
def calc_avg_open(csv_data_sp500, ticker):
        """This function takes in the csv_data_sp500 and a ticker.
         Return the average opening price for the stock as a float value.
        """
def calc_vwap(csv_data_sp500, ticker):
        """This function takes in the csv_data_sp500 and a ticker.  Return the volume weighted
        average price (VWAP) of the stock as a float.  In order to do this, first find the average
        price of the stock on each day.  Then, multiply that price with the volume on that day.
        Take the sum of these values.  Finally, divide that value by the sum of all the volumes.
        (hint: average price for each day = (high + low + close)/3)
        """
def calc_sma(company_data, days):
        """
                Compute Simple Moving Average for the given days for a given company
        """
def calc_rsi(company_data):
        ""
                Compute Relative Strength Index for a given company.
        ""
```

1. To perform these calculations, you should call above functions in a logical order, with the appropriate parameters. Use the given driver code. Warning: **DO NOT CHANGE** the driver code. You are free to experiment with the driver code, but don't forget to revert it back to the original form. It's advised to run the program during development for a single company (example ticker: "JAVA") until you get the correct results.

Note that stocks move in and out of the S&P 500. Some stocks may be represented in the CSV file, but not in the XML file (and vice-versa). Display "No data in SP500" for the names of these tickers. For example, for ticker JAVA,

Ticker: JAVA Name: No data in SP500 Avg: 9.01 VWAP: 9.00 RSI: 51.17

2. Calculate the Simple Moving Average (SMA) for a company and the number of days (e.g., SMA-10 for 10 days, SMA-14 for 14 days).
   Simple Moving Average is one of the core technical indicators used by traders and investors for the technical analysis of a stock. Simple moving average is calculated by adding the "Close" price of last N number of days (window size N) and then dividing it by the number of days.

   SMA = (Sum of ticker Close value over the past N days) / (N)

   Note that Pandas dataframe.rolling() function provides the feature of rolling window calculations given window=N parameter. You don't need to display the SMA values of this function.

3. Calculate the Relative Strength Index for all the companies.
   Relative Strength Index (RSI) is a momentum oscillator that measures the speed and change of price movements of a stock. RSI oscillates between zero and 100. RSI is generally created for a certain duration for a ticker (e.g., 14 days), but in this problem, we are going to calculate RSI for all the given days for a certain ticker.

   First calculate the rolling difference using the "Close" value for a ticker.

   rolling_difference = (Close of current day – Close of previous day)

   (Hint: pandas rolling() function can provide rolling window of 2 days and use apply and a lambda function to calculate the difference)

   Create two DataFrame columns ['Gain'] and ['Loss'] where Gain is when the rolling_difference is greater than zero(>0) and Loss is when the rolling_difference is less than or equal (<=) to zero. Don't forget to add zero in each case (if there's Gain, then add zero to Loss and vice versa). Hint: You can use list comprehension to generate gain and loss using the rolling_difference column.

   Now calculate the "Average Gain" and "Average Loss"

   Relative Strength (RS) = Average Gain / Average Loss

   RSI = 100 – (100/(1+RS))

4. You **CANNOT USE** any regular python loops inside calc_avg_open, calc_vwap, calc_sma and calc_rsi functions. Instead, use DataFrame techniques learned in pandas.

What to turn in: Submit your .py file to Blackboard. Due: Sunday, October 31, 2021, 11.59pm

Lastname-hw3.py should contain the following information at the top:
CS620
HW3
@author: <Your Name and UIN>