Metadata

```
Course:  DS 5100
Module:  07 Python Classes
Topic:   HW 07 Stock Class
Author:  R.C. Alvarado
Date:    7 July 2023
```

Student Info

- Name: Hilde Younce
- Net UD: ksg8xy
- URL of this file in GitHub: https://github.com/hyounce/DS5100-ksg8xy/blob/main/lessons/M07/hw07.ipynb

Instructions

In your **private course repo on Rivanna**, use this Jupyter notebook and the data file described to write code that performs the tasks below.

Save your notebook in the `M07` directory.

Remember to add and commit these files to your repo.

Then push your commits to your repo on GitHib.

Be sure to fill out the **Student Info** block above.

To submit your homework, save the notebook as a PDF and upload it to GradeScope, following the instructions.

TOTAL POINTS: 12

Overview

In this assignment you will define a class and use it to perform the requested tasks.

Before answering the questions, read the market data from `apple_data.csv` into a Pandas dataframe. The file is in the HW for this module in the course repo.

Setting Up

In [ ]:
```
import pandas as pd
import numpy as np
```

Prepare the Data

Read in the dataset from the attached file `apple_data.csv` using `pd.read_csv()`.

In [ ]:
```
apple_data = pd.read_csv("apple_data-2.csv")
```

Task 1

(5 PTS)

Define a class with these features:

Class Name: `Stock`

Attributes:

- `ticker` : a string to hold the stock symbol
- `sector` : a string to hold the sector name
- `prices` : a dataframe to hold the prices for the stock

Methods:

- `print_sector` to just print out the sector string.
- `get_row_count` to count the number of rows in the price dataframe. Set an attribute "price_records" equal to this count.

- `__init__` to build objects. Initialize with the three attribute values passed to the constructor.

In [ ]:
```
class Stock:

    def __init__(self, ticker, sector, prices):
        self.ticker = ticker
        self.sector = sector
        self.prices = prices

    def print_sector(self):
        print(self.sector)
    def get_row_count(self):
        self.price_records = len(self.prices)
```

Task 2

(1 PT)

Create an instance of your class with the these initial values:

- `ticker` : 'AAPL'
- `sector` : 'technology'
- `prices` : *the imported price dataframe*

Then Use the dot operator to print the stock's ticker.

```
In [ ]: stock1 = Stock('AAPL', 'technology', apple_data)
        stock1.ticker
```

```
Out[ ]: 'AAPL'
```

Task 3

(1 PT)

Use the `print_sector()` method to print the sector.

```
In [ ]: stock1.print_sector()
```

```
technology
```

Task 4

(2 PTS)

Use the `get_row_count()` method to compute the number of price records and set price_records.

Use the dot operator to access the stock's price_records, printing the result.

```
In [ ]: stock1.get_row_count()
        stock1.price_records
```

```
Out[ ]: 135
```

Task 5

(1 PT)

Add a new column called `'month'` to the `prices` attribute and put the month number there.

Hint: You can use `.apply()` with a lambda function to split the month string and keep the second element.

In [ ]:
```
stock1.prices['month'] = stock1.prices['date'].apply(lambda date: date.split
stock1.prices
```

Out[ ]:

	date	adj_close	month
0	2020-01-02	298.829956	01
1	2020-01-03	295.924713	01
2	2020-01-06	298.282715	01
3	2020-01-07	296.879883	01
4	2020-01-08	301.655548	01
...	...	...	...
130	2020-07-09	383.010010	07
131	2020-07-10	383.679993	07
132	2020-07-13	381.910004	07
133	2020-07-14	388.230011	07
134	2020-07-15	390.899994	07

135 rows × 3 columns

Task 6

(1 PT)

Use `.groupby()` to compute the mean `adj_close` by month. Save your result is a dataframe, not a series.

In [ ]:
```
mean_df = stock1.prices.groupby('month').agg({'adj_close': 'mean'})
mean_df
```

Out[ ]:

month	adj_close
01	310.337596
02	310.271843
03	261.735581
04	271.650839
05	309.785164
06	345.806360
07	378.385999

In [ ]:
```
type(mean_df)
```

Out[ ]:
```
pandas.core.frame.DataFrame
```

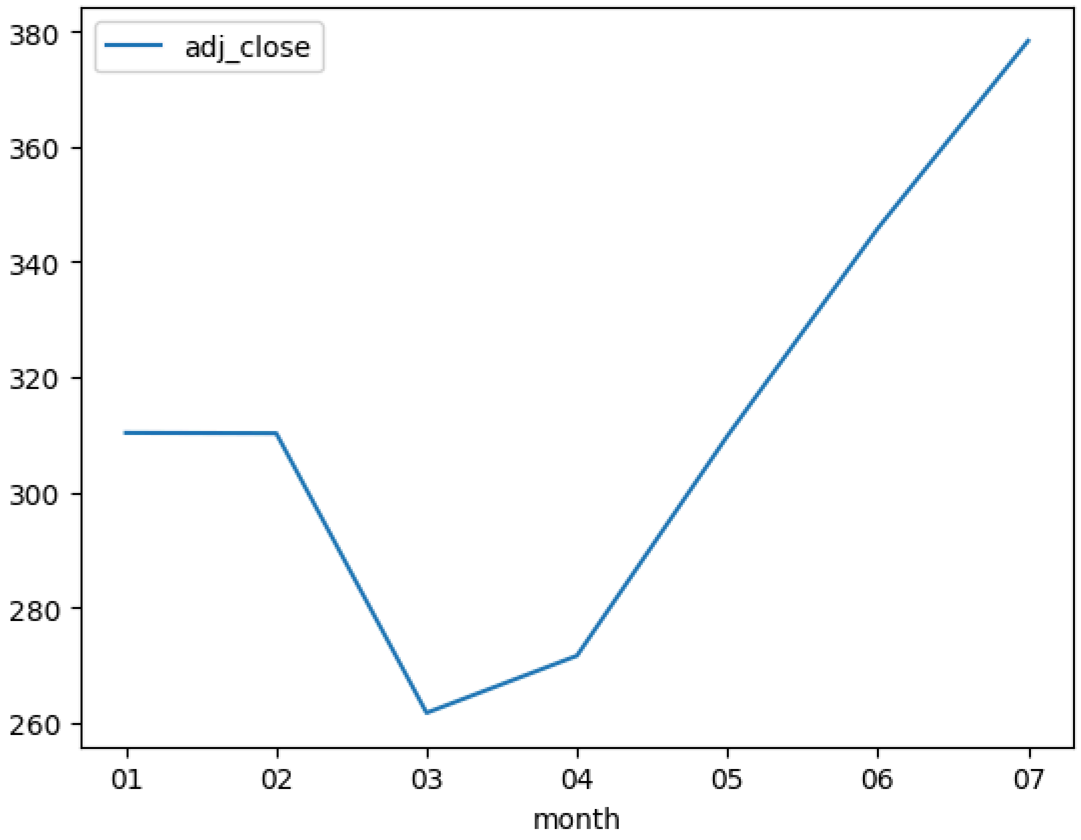
Task 7

(1 PT)

Plot the mean `adj_close` by month using a simple line plot.

In [ ]:
```
mean_df.plot.line()
```

Out[ ]:
```
<Axes: xlabel='month'>
```


adj_close
380
360
340
320
300
280
260
01
02
03
04
05
06
07
month