```
!pip install yfinance
import pandas as pd
```

```
Requirement already satisfied: yfinance in /usr/local/lib/python3.10/dist-packages (0.2.31)
Requirement already satisfied: pandas>=1.3.0 in /usr/local/lib/python3.10/dist-packages (from yfinance) (1.5.3)
Requirement already satisfied: numpy>=1.16.5 in /usr/local/lib/python3.10/dist-packages (from yfinance) (1.23.5)
Requirement already satisfied: requests>=2.31 in /usr/local/lib/python3.10/dist-packages (from yfinance) (2.31.0)
Requirement already satisfied: multitasking>=0.0.7 in /usr/local/lib/python3.10/dist-packages (from yfinance) (0.0.11)
Requirement already satisfied: lxml>=4.9.1 in /usr/local/lib/python3.10/dist-packages (from yfinance) (4.9.3)
Requirement already satisfied: appdirs>=1.4.4 in /usr/local/lib/python3.10/dist-packages (from yfinance) (1.4.4)
Requirement already satisfied: pytz>=2022.5 in /usr/local/lib/python3.10/dist-packages (from yfinance) (2023.3.post1)
Requirement already satisfied: frozendict>=2.3.4 in /usr/local/lib/python3.10/dist-packages (from yfinance) (2.3.8)
Requirement already satisfied: peewee>=3.16.2 in /usr/local/lib/python3.10/dist-packages (from yfinance) (3.17.0)
Requirement already satisfied: beautifulsoup4>=4.11.1 in /usr/local/lib/python3.10/dist-packages (from yfinance) (4.11.2)
Requirement already satisfied: html5lib>=1.1 in /usr/local/lib/python3.10/dist-packages (from yfinance) (1.1)
Requirement already satisfied: soupsieve>1.2 in /usr/local/lib/python3.10/dist-packages (from beautifulsoup4>=4.11.1->yfinance) (2.5)
Requirement already satisfied: six>=1.9 in /usr/local/lib/python3.10/dist-packages (from html5lib>=1.1->yfinance) (1.16.0)
Requirement already satisfied: webencodings in /usr/local/lib/python3.10/dist-packages (from html5lib>=1.1->yfinance) (0.5.1)
Requirement already satisfied: python-dateutil>=2.8.1 in /usr/local/lib/python3.10/dist-packages (from pandas>=1.3.0->yfinance) (2.8.2)
Requirement already satisfied: charset-normalizer<4,>=2 in /usr/local/lib/python3.10/dist-packages (from requests>=2.31->yfinance) (3.3.
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.10/dist-packages (from requests>=2.31->yfinance) (3.4)
Requirement already satisfied: urllib3<3,>=1.21.1 in /usr/local/lib/python3.10/dist-packages (from requests>=2.31->yfinance) (2.0.7)
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.10/dist-packages (from requests>=2.31->yfinance) (2023.7.22)
```

```
import pandas as pd
import yfinance as yf
from datetime import datetime


start_date = datetime.now() - pd.DateOffset(months=3)
end_date = datetime.now()


tickers = ['AAPL', 'MSFT', 'NFLX', 'GOOG']


df_list = []


for ticker in tickers:
    data = yf.download(ticker, start=start_date, end=end_date)
    df_list.append(data)
```

```
[*********************100%%**********************]  1 of 1 completed
[*********************100%%**********************]  1 of 1 completed
[*********************100%%**********************]  1 of 1 completed
[*********************100%%**********************]  1 of 1 completed
```

```
df=pd.read_csv('stocks.csv')

print(df.head())
```

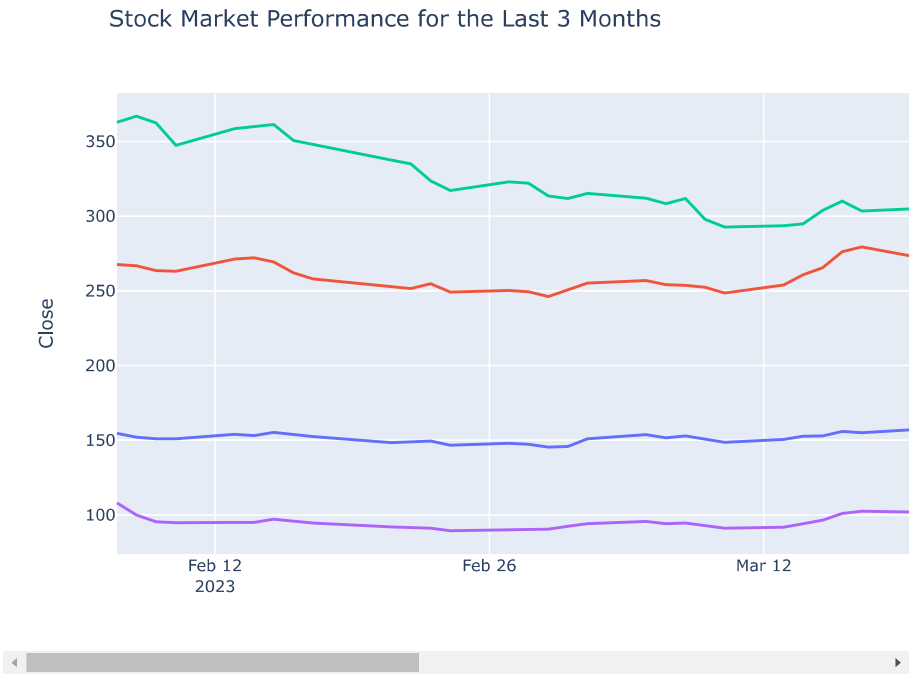
```
  Ticker        Date        Open        High         Low       Close  \
0   AAPL  2023-02-07  150.639999  155.229996  150.639999  154.649994
1   AAPL  2023-02-08  153.880005  154.580002  151.169998  151.919998
2   AAPL  2023-02-09  153.779999  154.330002  150.419998  150.869995
3   AAPL  2023-02-10  149.460007  151.339996  149.220001  151.009995
4   AAPL  2023-02-13  150.949997  154.259995  150.919998  153.850006

    Adj Close    Volume
0  154.414230  83322600
1  151.688400  64120100
2  150.639999  56007100
3  151.009995  57450700
4  153.850006  62199000
```

```
df = df.reset_index()
print(df.head())
```

```
   index Ticker        Date        Open        High         Low       Close  \
0      0   AAPL  2023-02-07  150.639999  155.229996  150.639999  154.649994
1      1   AAPL  2023-02-08  153.880005  154.580002  151.169998  151.919998
2      2   AAPL  2023-02-09  153.779999  154.330002  150.419998  150.869995
3      3   AAPL  2023-02-10  149.460007  151.339996  149.220001  151.009995
4      4   AAPL  2023-02-13  150.949997  154.259995  150.919998  153.850006
```

```
    Adj Close    Volume
0  154.414230  83322600
1  151.688400  64120100
2  150.639999  56007100
3  151.009995  57450700
4  153.850006  62199000
```

```
import plotly.express as px
fig = px.line(df, x='Date',
              y='Close',
              color='Ticker',
              title="Stock Market Performance for the Last 3 Months")
fig.show()
```

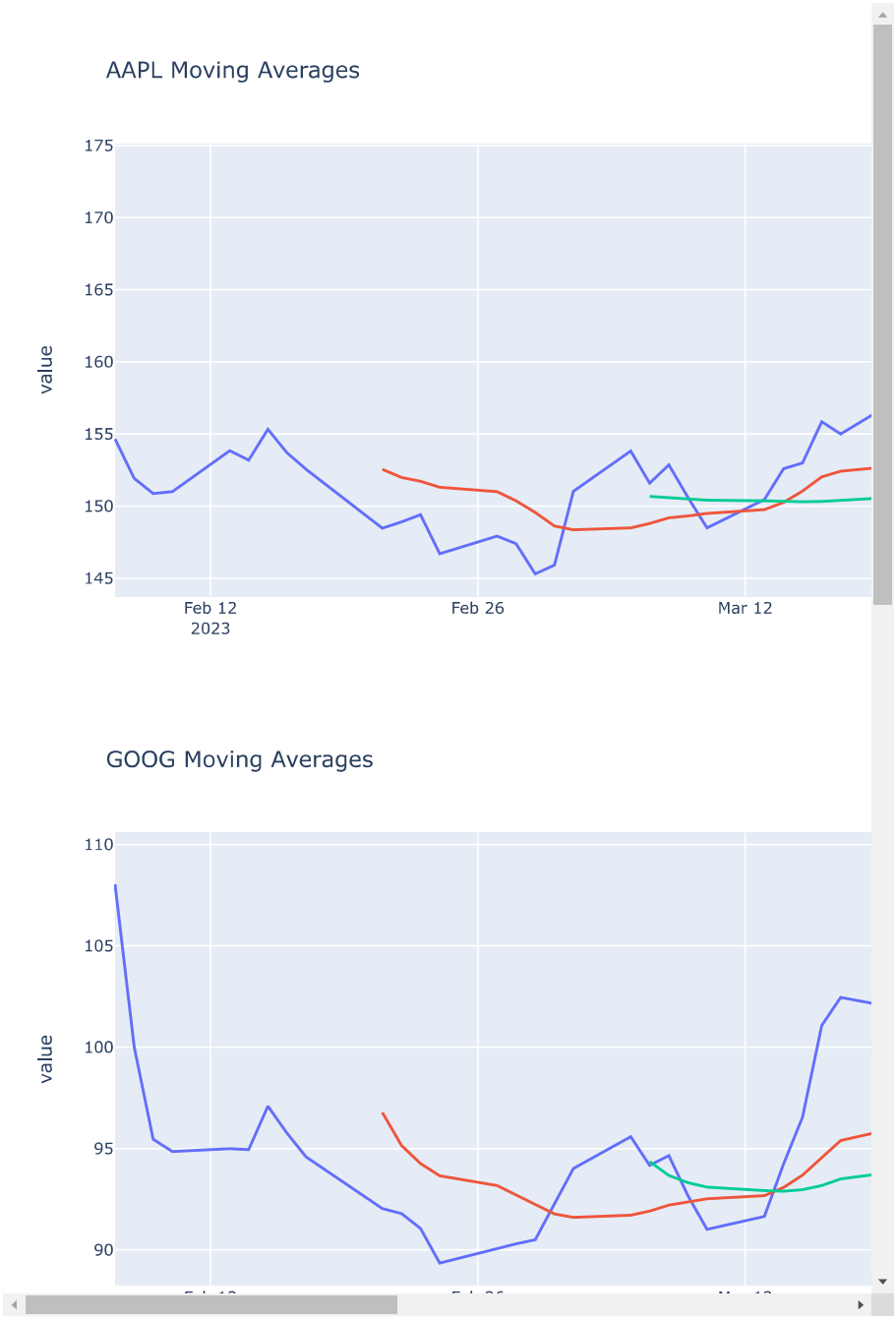



```
fig = px.area(df, x='Date', y='Close', color='Ticker',
              facet_col='Ticker',
              labels={'Date':'Date', 'Close':'Closing Price', 'Ticker':'Company'},
              title='Stock Prices for Apple, Microsoft, Netflix, and Google')
fig.show()
```

```
df['MA10'] = df.groupby('Ticker')['Close'].rolling(window=10).mean().reset_index(0, drop=True)
df['MA20'] = df.groupby('Ticker')['Close'].rolling(window=20).mean().reset_index(0, drop=True)

for ticker, group in df.groupby('Ticker'):
    print(f'Moving Averages for {ticker}')
    print(group[['MA10', 'MA20']])
```

```
 Moving Averages for AAPL
           MA10      MA20
 0          NaN       NaN
 1          NaN       NaN
 2          NaN       NaN
 3          NaN       NaN
 4          NaN       NaN
 ..         ...       ...
 57  166.631000  165.2730
 58  166.837999  165.3915
 59  166.819998  165.4825
 60  166.733998  165.5840
 61  167.588998  166.0295

 [62 rows x 2 columns]
 Moving Averages for GOOG
            MA10        MA20
 186         NaN         NaN
 187         NaN         NaN
 188         NaN         NaN
 189         NaN         NaN
 190         NaN         NaN
 ..          ...         ...
 243  106.209000  106.416500
 244  106.295000  106.470000
 245  106.405001  106.520000
 246  106.336001  106.533001
 247  106.366500  106.398750

 [62 rows x 2 columns]
 Moving Averages for MSFT
            MA10        MA20
 62          NaN         NaN
 63          NaN         NaN
 64          NaN         NaN
 65          NaN         NaN
 66          NaN         NaN
 ..          ...         ...
 119  291.889999  289.487000
 120  293.594000  290.395999
 121  295.188998  291.256999
 122  297.119000  292.310500
 123  299.607999  293.262999

 [62 rows x 2 columns]
 Moving Averages for NFLX
            MA10        MA20
 124         NaN         NaN
 125         NaN         NaN
 126         NaN         NaN
 127         NaN         NaN
 128         NaN         NaN
 ..          ...         ...
 181  326.276999  333.262498
 182  324.661996  331.725998
 183  324.279996  330.353497
 184  323.822995  329.274997
 185  323.300995  328.446498
```

```
for ticker, group in df.groupby('Ticker'):
    fig = px.line(group, x='Date', y=['Close', 'MA10', 'MA20'],
                  title=f"{ticker} Moving Averages")
    fig.show()
```


AAPL Moving Averages
175
170
165
160
155
150
145
value
Feb 12
2023
Feb 26
Mar 12
GOOG Moving Averages
110
105
100
95
90
value

```
df['Volatility'] = df.groupby('Ticker')['Close'].pct_change().rolling(window=10).std().reset_index(0, drop=True)
fig = px.line(df, x='Date', y='Volatility',
              color='Ticker',
              title='Volatility of All Companies')
fig.show()
```


```
# create a DataFrame with the stock prices of Apple and Microsoft
apple = df.loc[df['Ticker'] == 'AAPL', ['Date', 'Close']].rename(columns={'Close': 'AAPL'})
microsoft = df.loc[df['Ticker'] == 'MSFT', ['Date', 'Close']].rename(columns={'Close': 'MSFT'})
df_corr = pd.merge(apple, microsoft, on='Date')

# create a scatter plot to visualize the correlation
fig = px.scatter(df_corr, x='AAPL', y='MSFT',
                 trendline='ols',
                 title='Correlation between Apple and Microsoft')
fig.show()
```