Name : Sarvagya Singh **SAPID : 60009200030**

Experiment 2

Aim: Analysis of Yahoo Finance Indexes, Currencies / FX, Cryptocurrencies, Funds & ETFs, Treasury Yields, Stock Fundamentals, Meta Data and Performance Metrics, Balance Sheet, P&L, Cashflows

Objective:
- Get historical data for stocks
- Plot the stock market data and analyse the performance
- Get the fundamental, futures and options data

Theory:

Historical Stock Price Data
Stocks are very volatile instruments and therefore very important to thoroughly analyse the price behaviour before making any trading decisions. Hence fetching and analysing the prices is crucial. The stock data can be downloaded from different packages such as yahoo finance, quandl and alpha vantage. This data can be fetched the daily and minute level from yahoo finance. It is essential to understand data structures, data analysis, dealing with financial data, and for generating trading signals. This experiment will build learning on how to fetch various data like pricing data of stocks, fundamental.

Yahoo Finance Market Data
Yahoo Finance: One of the first sources to get historical daily price-volume stock market data is Yahoo finance. Use pandas_datareader or yfinance module to get the data and then can download or store in a csv file by using pandas.to_csv method. The yfinance module has the download method which can be used to download the stock market data. To get stock market data for different geographies, search the ticker symbol on Yahoo finance and use that as the ticker.

Stock Market Data Visualization and Analysis
After having the stock market data, the next step is to create trading strategies and analyse the performance. The ease of analysing the performance is the key advantage of the Python.
Analyse the cumulative returns, drawdown plot, different ratios such as
- Sharpe ratio,
- Sortino ratio, and
- Calmar ratio.

Conclusion

A.Y.: 2022-23 **Class/ Sem: T.Y.B.Tech/ Sem-VI** **Sub:** Computational Finance

To be able to use the Python codes to fetch the stock market data of your favourites stocks, build the strategies using this stock market data and analyse this data.

Lab Experiment to be done by students:

1. Analysis and Import Financial Indexes
2. Import Currencies / FX
3. Import Cryptocurrencies trading
4. Import Funds & ETFs
5. Import Treasury Yields
6. Generate Ticker Object
7. Implement Stock Fundamentals, Meta Data and Performance Metrics
8. Analyse Financials (Balance Sheet, P&L, Cashflows)
9. Generate Put / Call Options (Stocks)
10. Stream real-time Data Build a model by Importing Stocks data for market analysis.
11. Perform Dividend Indexes Analysis on specified data source.

Sarvagya Singh

60009200030 -- K1

FMC -- lab2

```
pip install yfinance
```

```
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/public/simple/
Collecting yfinance
  Downloading yfinance-0.2.12-py2.py3-none-any.whl (59 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 59.2/59.2 KB 1.3 MB/s eta 0:00:00
Requirement already satisfied: pandas>=1.3.0 in /usr/local/lib/python3.9/dist-packages (from yfinance) (1.3.5)
Collecting appdirs>=1.4.4
  Downloading appdirs-1.4.4-py2.py3-none-any.whl (9.6 kB)
Collecting frozendict>=2.3.4
  Downloading frozendict-2.3.5-cp39-cp39-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (112 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 112.8/112.8 KB 3.4 MB/s eta 0:00:00
Collecting requests>=2.26
  Downloading requests-2.28.2-py3-none-any.whl (62 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 62.8/62.8 KB 2.3 MB/s eta 0:00:00
Requirement already satisfied: numpy>=1.16.5 in /usr/local/lib/python3.9/dist-packages (from yfinance) (1.22.4)
Collecting beautifulsoup4>=4.11.1
  Downloading beautifulsoup4-4.11.2-py3-none-any.whl (129 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 129.4/129.4 KB 1.6 MB/s eta 0:00:00
Requirement already satisfied: multitasking>=0.0.7 in /usr/local/lib/python3.9/dist-packages (from yfinance) (0.0.11)
Collecting cryptography>=3.3.2
  Downloading cryptography-39.0.2-cp36-abi3-manylinux_2_28_x86_64.whl (4.2 MB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 4.2/4.2 MB 42.0 MB/s eta 0:00:00
Requirement already satisfied: pytz>=2022.5 in /usr/local/lib/python3.9/dist-packages (from yfinance) (2022.7.1)
Requirement already satisfied: lxml>=4.9.1 in /usr/local/lib/python3.9/dist-packages (from yfinance) (4.9.2)
Collecting html5lib>=1.1
  Downloading html5lib-1.1-py2.py3-none-any.whl (112 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 112.2/112.2 KB 9.0 MB/s eta 0:00:00
Collecting soupsieve>1.2
  Downloading soupsieve-2.4-py3-none-any.whl (37 kB)
Requirement already satisfied: cffi>=1.12 in /usr/local/lib/python3.9/dist-packages (from cryptography>=3.3.2->yfinance) (1.15.1)
Requirement already satisfied: six>=1.9 in /usr/local/lib/python3.9/dist-packages (from html5lib>=1.1->yfinance) (1.15.0)
Requirement already satisfied: webencodings in /usr/local/lib/python3.9/dist-packages (from html5lib>=1.1->yfinance) (0.5.1)
Requirement already satisfied: python-dateutil>=2.7.3 in /usr/local/lib/python3.9/dist-packages (from pandas>=1.3.0->yfinance) (2.8.2)
Collecting charset-normalizer<4,>=2
  Downloading charset_normalizer-3.1.0-cp39-cp39-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (199 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 199.2/199.2 KB 17.7 MB/s eta 0:00:00
Requirement already satisfied: urllib3<1.27,>=1.21.1 in /usr/local/lib/python3.9/dist-packages (from requests>=2.26->yfinance) (1.26.14)
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.9/dist-packages (from requests>=2.26->yfinance) (2.10)
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.9/dist-packages (from requests>=2.26->yfinance) (2022.12.7)
Requirement already satisfied: pycparser in /usr/local/lib/python3.9/dist-packages (from cffi>=1.12->cryptography>=3.3.2->yfinance) (2.21)
Installing collected packages: appdirs, soupsieve, html5lib, frozendict, charset-normalizer, requests, cryptography, beautifulsoup4, yfinance
  Attempting uninstall: html5lib
    Found existing installation: html5lib 1.0.1
    Uninstalling html5lib-1.0.1:
      Successfully uninstalled html5lib-1.0.1
  Attempting uninstall: requests
    Found existing installation: requests 2.25.1
    Uninstalling requests-2.25.1:
      Successfully uninstalled requests-2.25.1
  Attempting uninstall: beautifulsoup4
    Found existing installation: beautifulsoup4 4.6.3
    Uninstalling beautifulsoup4-4.6.3:
      Successfully uninstalled beautifulsoup4-4.6.3
Successfully installed appdirs-1.4.4 beautifulsoup4-4.11.2 charset-normalizer-3.1.0 cryptography-39.0.2 frozendict-2.3.5 html5lib-1.1 requests-2.28.2 soupsieve-2.4 yfinance-0.2.12
```

```
import yfinance as yf
import pandas as pd
import numpy as np
```

```
forex_data = yf.download("EURUSD=X", start = '2019-01-02', end = '2021-12-31')
forex_data
```

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

Date	Open	High	Low	Close	Adj Close	Volume
2019-01-02	1.146132	1.149700	1.134572	1.146171	1.146171	0
2019-01-03	1.131734	1.140914	1.131734	1.131811	1.131811	0
2019-01-04	1.139095	1.141774	1.134816	1.139108	1.139108	0
2019-01-07	1.141292	1.147447	1.140524	1.141044	1.141044	0
2019-01-08	1.147934	1.148633	1.142465	1.147974	1.147974	0
...	...	...	...	...	...	...
2021-12-24	1.132464	1.134430	1.130838	1.132734	1.132734	0
2021-12-27	1.132387	1.133500	1.130416	1.132426	1.132426	0
2021-12-28	1.132978	1.133600	1.129038	1.133003	1.133003	0
2021-12-29	1.131337	1.137001	1.127536	1.131478	1.131478	0
2021-12-30	1.135976	1.135976	1.130071	1.136015	1.136015	0

781 rows × 6 columns

```
cryto = yf.download("BTC-USD", start = '2019-01-02', end = '2021-12-31')
cryto
```

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

Date	Open	High	Low	Close	Adj Close	Volume
2019-01-02	3849.216309	3947.981201	3817.409424	3943.409424	3943.409424	5244856836
2019-01-03	3931.048584	3935.685059	3826.222900	3836.741211	3836.741211	4530215219
2019-01-04	3832.040039	3865.934570	3783.853760	3857.717529	3857.717529	4847965467
2019-01-05	3851.973877	3904.903076	3836.900146	3845.194580	3845.194580	5137609824
2019-01-06	3836.519043	4093.297363	3826.513184	4076.632568	4076.632568	5597027440
...	...	...	...	...	...	...
2021-12-26	50428.691406	51196.378906	49623.105469	50809.515625	50809.515625	20964372926
2021-12-27	50802.609375	51956.328125	50499.468750	50640.417969	50640.417969	24324345758
2021-12-28	50679.859375	50679.859375	47414.210938	47588.855469	47588.855469	33430376883
2021-12-29	47623.871094	48119.742188	46201.496094	46444.710938	46444.710938	30049226299
2021-12-30	46490.605469	47879.964844	46060.312500	47178.125000	47178.125000	26686491018

1094 rows × 6 columns

```
ticker = ['^DJI', '^GSPC']
```

```
yf.download(ticker, period = '5y')
```

```
[*********************100%***********************]  2 of 2 completed
```

	Adj Close		Close		High		Low		Open		Volume	
	^DJI	^GSPC	^DJI	^GSPC	^DJI	^GSPC	^DJI	^GSPC	^DJI	^GSPC	^DJI	^GSPC
Date												
2018-03-05	24874.759766	2720.939941	24874.759766	2720.939941	24961.000000	2728.090088	24387.150391	2675.750000	24471.310547	2681.060059	383910000	3733640000
2018-03-06	24884.119141	2728.120117	24884.119141	2728.120117	24995.240234	2732.080078	24708.410156	2711.260010	24965.890625	2730.179932	331770000	3389030000
2018-03-07	24801.359375	2726.800049	24801.359375	2726.800049	24849.679688	2730.600098	24535.119141	2701.739990	24758.150391	2710.179932	394070000	3406690000
2018-03-08	24895.210938	2738.969971	24895.210938	2738.969971	24950.490234	2740.449951	24703.050781	2722.649902	24853.410156	2732.750000	327300000	3206040000
2018-03-09	25335.740234	2786.570068	25335.740234	2786.570068	25336.330078	2786.570068	25004.890625	2751.540039	25004.890625	2752.909912	371570000	3376870000
...	...	...	...	...	...	...	...	...	...	...	...	...
2023-02-27	32889.089844	3982.239990	32889.089844	3982.239990	33189.281250	4018.050049	32814.179688	3973.550049	32906.160156	3992.360107	243660000	3836950000
2023-02-28	32656.699219	3970.149902	32656.699219	3970.149902	32873.468750	3997.500000	32636.429688	3968.979980	32873.468750	3977.189941	303760000	5043400000
2023-03-01	32661.839844	3951.389893	32661.839844	3951.389893	32746.150391	3971.729980	32500.710938	3939.050049	32656.369141	3963.340088	292810000	4249480000
2023-03-02	33003.570312	3981.350098	33003.570312	3981.350098	33083.449219	3990.840088	32665.849609	3928.159912	32780.968750	3938.679932	317450000	4244900000
2023-03-03	33390.968750	4045.639893	33390.968750	4045.639893	33405.820312	4048.290039	33008.410156	3995.169922	33076.328125	3998.020020	300050000	4084730000

1259 rows × 12 columns

```
indexes = yf.download(ticker, period = '5y').Close
indexes
```

```
[*********************100%***********************]  2 of 2 completed
```

Date	^DJI	^GSPC
2018-03-05	24874.759766	2720.939941
2018-03-06	24884.119141	2728.120117
2018-03-07	24801.359375	2726.800049
2018-03-08	24895.210938	2738.969971
2018-03-09	25335.740234	2786.570068
...	...	...
2023-02-27	32889.089844	3982.239990
2023-02-28	32656.699219	3970.149902
2023-03-01	32661.839844	3951.389893
2023-03-02	33003.570312	3981.350098
2023-03-03	33390.968750	4045.639893

1259 rows × 2 columns

```
norm = indexes.div(indexes.iloc[0]).mul(100) #other divided by 1st row and multiplied by
norm
```

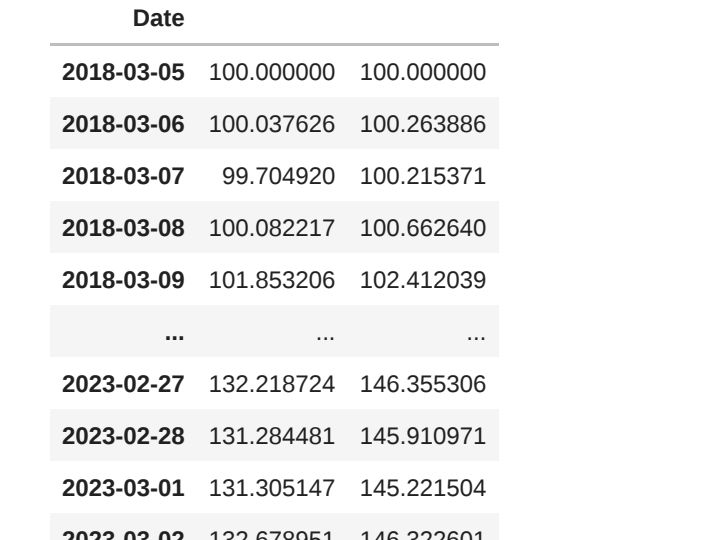
Date	^DJI	^GSPC
2018-03-05	100.000000	100.000000
2018-03-06	100.037626	100.263886
2018-03-07	99.704920	100.215371
2018-03-08	100.082217	100.662640
2018-03-09	101.853206	102.412039
...	...	...
2023-02-27	132.218724	146.355306
2023-02-28	131.284481	145.910971
2023-03-01	131.305147	145.221504
2023-03-02	132.678951	146.322601
2023-03-03	134.236347	148.685380

1259 rows × 2 columns

```
24884.119141/24874.759766 *100
```

```
100.03762599151929
```

```
import matplotlib.pyplot as plt
norm.plot()
plt.show()
```



Importing Currencies

```
ticker = 'EURUSD=X'
ticker2 = 'USDEUR=X'
```

```
yf.download(ticker, period = '5y')
```

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

Date	Open	High	Low	Close	Adj Close	Volume
2018-03-05	1.231846	1.234827	1.227551	1.231542	1.231542	0
2018-03-06	1.233776	1.241989	1.233122	1.233654	1.233654	0
2018-03-07	1.241928	1.244060	1.238605	1.241665	1.241665	0
2018-03-08	1.241619	1.244478	1.230633	1.241465	1.241465	0
2018-03-09	1.230784	1.233380	1.227551	1.230663	1.230663	0
...	...	...	...	...	...	...
2023-02-27	1.055476	1.061864	1.053352	1.055476	1.055476	0
2023-02-28	1.061121	1.064500	1.058235	1.061121	1.061121	0
2023-03-01	1.057731	1.069107	1.055669	1.057731	1.057731	0
2023-03-02	1.066894	1.067270	1.058347	1.066894	1.066894	0
2023-03-03	1.059973	1.062925	1.059131	1.059973	1.059973	0

1304 rows × 6 columns

```
yf.download(ticker2, period = '5y')
```

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

Date	Open	High	Low	Close	Adj Close	Volume
2018-03-05	0.81179	0.81463	0.809830	0.81199	0.81199	0
2018-03-06	0.81052	0.81095	0.805160	0.81060	0.81060	0
2018-03-07	0.80520	0.80736	0.803820	0.80537	0.80537	0
2018-03-08	0.80540	0.81259	0.803550	0.80550	0.80550	0
2018-03-09	0.81249	0.81463	0.810780	0.81257	0.81257	0
...	...	...	...	...	...	...
2023-02-27	0.94744	0.94935	0.941740	0.94744	0.94744	0
2023-02-28	0.94240	0.94497	0.939408	0.94240	0.94240	0
2023-03-01	0.94542	0.94646	0.935360	0.94542	0.94542	0
2023-03-02	0.93730	0.94487	0.936970	0.93730	0.93730	0
2023-03-03	0.94342	0.94417	0.940800	0.94342	0.94342	0

1304 rows × 6 columns

```
ticker3 = 'USDGBP=X' #british pounds
yf.download(ticker3, period = '5y')
```

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

Date	Open	High	Low	Close	Adj Close	Volume
2018-03-05	0.72470	0.72626	0.72060	0.72490	0.72490	0
2018-03-06	0.72202	0.72356	0.71796	0.72190	0.72190	0
2018-03-07	0.71941	0.72209	0.71890	0.71942	0.71942	0
2018-03-08	0.71910	0.72387	0.71880	0.71910	0.71910	0
2018-03-09	0.72447	0.72520	0.72012	0.72434	0.72434	0
...	...	...	...	...	...	...
2023-02-27	0.83674	0.83870	0.82940	0.83665	0.83665	0
2023-02-28	0.82898	0.83142	0.82355	0.82908	0.82908	0
2023-03-01	0.83116	0.83564	0.82725	0.83101	0.83101	0
2023-03-02	0.83123	0.83839	0.83082	0.83126	0.83126	0
2023-03-03	0.83679	0.83687	0.83203	0.83671	0.83671	0

1304 rows × 6 columns

Importing crypocurrency

```
ticker = ['BTC-EUR', 'ETH-USD']
yf.download(ticker, period = '5y')
```

```
[*********************100%***********************]  2 of 2 completed
```

	Adj Close		Close		High		Low		Open		Volume	
	BTC-EUR	ETH-USD	BTC-EUR	ETH-USD	BTC-EUR	ETH-USD	BTC-EUR	ETH-USD	BTC-EUR	ETH-USD	BTC-EUR	ETH-USD
Date												
2018-03-04	9346.918945	866.677979	9346.918945	866.677979	9346.918945	867.950012	9041.243164	840.280029	9334.578125	856.185974	4939636080	1697939968
2018-03-05	9381.728516	853.684021	9381.728516	853.684021	9495.456055	869.919983	9285.865234	853.520020	9362.994141	866.846008	5243628327	1898489984
2018-03-06	8682.964844	816.950989	8682.964844	816.950989	9323.145508	853.817017	8623.740234	809.931030	9322.389648	853.817017	5503157458	1943069952
2018-03-07	8029.626953	752.830994	8029.626953	752.830994	8805.591797	825.606995	7808.668945	726.546997	8702.295898	816.270996	7088800271	2175419904
2018-03-08	7635.115723	704.596008	7635.115723	704.596008	8195.284180	773.767029	7533.180176	696.169006	8018.242188	752.570007	5839975607	1891640064
...	...	...	...	...	...	...	...	...	...	...	...	...
2023-02-28	21885.335938	1605.895142	21885.335938	1605.895142	22221.937500	1644.435791	21816.548828	1602.428101	22170.509766	1634.502441	19415754884	6323676957
2023-03-01	22164.501953	1663.433716	22164.501953	1663.433716	22427.162109	1663.433716	21846.828125	1601.548584	21888.718750	1606.040405	23117097627	7701847224
2023-03-02	22150.041016	1647.319336	22150.041016	1647.319336	22244.996094	1672.051147	21932.421875	1622.599976	22164.943359	1663.549072	19235382456	7080950926
2023-03-03	21014.589844	1569.167603	21014.589844	1569.167603	22156.498047	1648.475342	20890.906250	1552.451538	22151.142578	1647.557739	24491284673	9922240199
2023-03-04	21008.167969	1571.052856	21008.167969	1571.052856	21044.912109	1575.361938	21000.185547	1569.191162	21014.791016	1569.558594	18166724608	7019942400

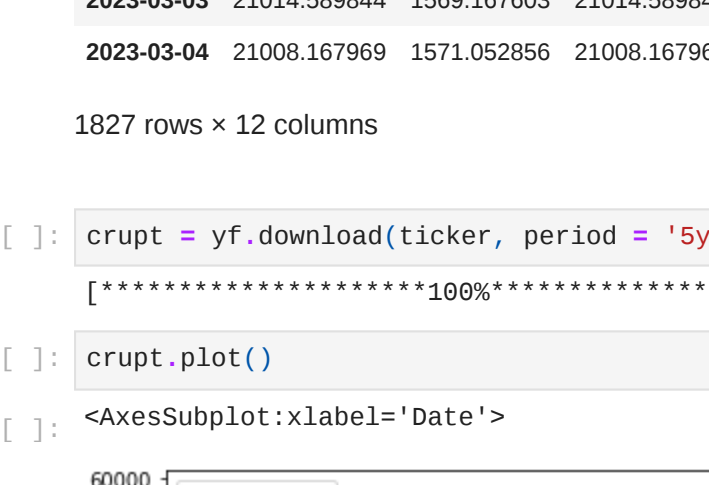
1827 rows × 12 columns

```
crupt = yf.download(ticker, period = '5y').Close
```

```
[*********************100%***********************]  2 of 2 completed
```

```
crupt.plot()
```

```
<AxesSubplot:xlabel='Date'>
```



Importing Funds

```
ticker = 'TLT'
ticker2 = 'OMOIX'
```

```
yf.download(ticker, period = '5y')
```

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

Date	Open	High	Low	Close	Adj Close	Volume
2018-03-05	118.750000	118.800003	117.629997	118.029999	106.336670	5658200
2018-03-06	118.360001	118.680000	118.120003	118.139999	106.435806	5452000
2018-03-07	118.570000	118.680000	117.959999	118.010002	106.318657	4461800
2018-03-08	118.309998	118.989998	118.260002	118.690002	106.931297	7086700
2018-03-09	117.989998	118.269997	117.589996	117.910004	106.228577	6570400
...	...	...	...	...	...	...
2023-02-27	101.290001	101.699997	100.910004	101.250000	101.001129	14209200
2023-02-28	100.570000	101.720001	100.279999	101.709999	101.459999	14955700
2023-03-01	100.820000	101.029999	99.959999	100.370003	100.370003	19000000
2023-03-02	99.059998	99.650002	98.879997	99.480003	99.480003	23769200
2023-03-03	100.930000	101.900002	100.570000	101.889999	101.889999	30854200

1259 rows × 6 columns

```
yf.download(ticker2, period = '5y')
```

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

Date	Open	High	Low	Close	Adj Close	Volume
2018-03-05	26.250000	26.250000	26.250000	26.250000	20.343348	0
2018-03-06	26.260000	26.260000	26.260000	26.260000	20.351097	0
2018-03-07	26.219999	26.219999	26.219999	26.219999	20.320095	0
2018-03-08	26.219999	26.219999	26.219999	26.219999	20.320095	0
2018-03-09	26.200001	26.200001	26.200001	26.200001	20.304600	0
...	...	...	...	...	...	...
2023-02-27	23.879999	23.879999	23.879999	23.879999	23.879999	0
2023-02-28	23.870001	23.870001	23.870001	23.870001	23.870001	0
2023-03-01	23.830000	23.830000	23.830000	23.830000	23.830000	0
2023-03-02	23.860001	23.860001	23.860001	23.860001	23.860001	0
2023-03-03	23.930000	23.930000	23.930000	23.930000	23.930000	0

1259 rows × 6 columns

Importing Treasury

```
ticker = "^TNX"
ticker1 = "^FVX"
```

```
ticker2 = ["^TNX", "^FVX"]
```

```
yf.download(ticker2, period = '5y')
```

```
[*********************100%***********************]  2 of 2 completed
```

	Adj Close		Close		High		Low		Open		Volume	
	^FVX	^TNX	^FVX	^TNX	^FVX	^TNX	^FVX	^TNX	^FVX	^TNX	^FVX	^TNX
Date												
2018-03-05	2.644	2.881	2.644	2.881	2.657	2.895	2.590	2.830	2.603	2.842	0	0
2018-03-06	2.645	2.877	2.645	2.877	2.660	2.897	2.625	2.857	2.657	2.894	0	0
2018-03-07	2.649	2.883	2.649	2.883	2.650	2.883	2.618	2.850	2.630	2.864	0	0
2018-03-08	2.633	2.866	2.633	2.866	2.654	2.888	2.618	2.848	2.650	2.885	0	0
2018-03-09	2.654	2.894	2.654	2.894	2.677	2.914	2.633	2.870	2.645	2.875	0	0
...	...	...	...	...	...	...	...	...	...	...	...	...
2023-02-27	4.173	3.922	4.173	3.922	4.224	3.967	4.154	3.897	4.224	3.967	0	0
2023-02-28	4.168	3.916	4.168	3.916	4.229	3.983	4.159	3.904	4.191	3.947	0	0
2023-03-01	4.263	3.994	4.263	3.994	4.279	4.006	4.198	3.934	4.203	3.940	0	0
2023-03-02	4.325	4.073	4.325	4.073	4.350	4.091	4.304	4.044	4.314	4.050	0	0
2023-03-03	4.253	3.964	4.253	3.964	4.317	4.026	4.240	3.962	4.279	4.013	0	0

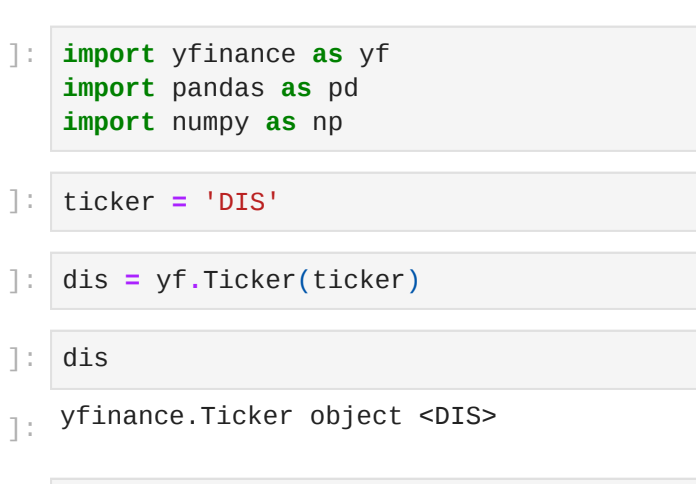
1259 rows × 12 columns

```
yields = yf.download(ticker2, period = '5y').Close
```

```
[*********************100%***********************]  2 of 2 completed
```

```
yields.plot()
```

```
<AxesSubplot:xlabel='Date'>
```



Ticker Object

```
import yfinance as yf
import pandas as pd
import numpy as np
```

```
ticker = 'DIS'
```

```
dis = yf.Ticker(ticker)
```

```
dis
```

```
yfinance.Ticker object <DIS>
```

```
dis.ticker
```

```
'DIS'
```

```
dis.history()
```

Date	Open	High	Low	Close	Volume	Dividends	Stock Splits
2023-02-13 00:00:00-05:00	108.099998	108.290001	106.550003	107.660004	12341500	0.0	0.0
2023-02-14 00:00:00-05:00	106.820000	108.440002	105.820000	107.660004	10768100	0.0	0.0
2023-02-15 00:00:00-05:00	107.089996	109.269997	106.699997	109.239998	8156000	0.0	0.0
2023-02-16 00:00:00-05:00	107.550003	108.019997	105.699997	105.830002	9479900	0.0	0.0
2023-02-17 00:00:00-05:00	105.029999	105.800003	103.879997	105.220001	9492200	0.0	0.0
2023-02-21 00:00:00-05:00	103.550003	103.730003	101.830002	102.089996	10648200	0.0	0.0
2023-02-22 00:00:00-05:00	101.330002	102.559998	101.089996	101.680000	9885100	0.0	0.0
2023-02-23 00:00:00-05:00	101.449997	102.050003	99.610001	101.730003	10974800	0.0	0.0
2023-02-24 00:00:00-05:00	99.529999	100.430000	98.959999	100.300003	8523000	0.0	0.0
2023-02-27 00:00:00-05:00	100.730003	101.470001	100.059998	100.449997	6641200	0.0	0.0
2023-02-28 00:00:00-05:00	100.449997	100.750000	99.570000	99.610001	8233700	0.0	0.0
2023-03-01 00:00:00-05:00	99.190002	99.629997	98.099998	98.540001	7898300	0.0	0.0
2023-03-02 00:00:00-05:00	97.739998	99.290001	97.419998	98.919998	7776600	0.0	0.0
2023-03-03 00:00:00-05:00	99.669998	101.180000	98.879997	101.139999	7428900	0.0	0.0
2023-03-06 00:00:00-05:00	101.610001	102.699997	100.489998	100.660004	6773300	0.0	0.0
2023-03-07 00:00:00-05:00	100.400002	100.940002	98.889999	99.059998	6951300	0.0	0.0
2023-03-08 00:00:00-05:00	99.000000	99.620003	98.300003	99.300003	5668700	0.0	0.0
2023-03-09 00:00:00-05:00	99.449997	100.050003	95.919998	96.139999	8275600	0.0	0.0
2023-03-10 00:00:00-05:00	95.599998	95.900002	92.830002	93.570000	13964800	0.0	0.0

Stock Fundamentals, metadata and performance

```
dis
```

```
yfinance.Ticker object <DIS>
```

```
dis.info
```

```
pd.Series(dis.info)
```

```
<ipython-input-29-faf0221366c5>:1: DeprecationWarning: The default dtype for empty Series will be 'object' instead of 'float64' in a future version. Specify a dtype explicitly to silence this warning.
  pd.Series(dis.info)
Series([], dtype: float64)
```

```
df = pd.Series(dis.info, name = 'DIS').to_frame().T
```

```
<ipython-input-23-ff25e3ba9cdf>:1: DeprecationWarning: The default dtype for empty Series will be 'object' instead of 'float64' in a future version. Specify a dtype explicitly to silence this warning.
  df = pd.Series(dis.info, name = 'DIS').to_frame().T
```

```
dis
```

```
yfinance.Ticker object <DIS>
```

```
df = dis.history()
```

```
df
```

Date	Open	High	Low	Close	Volume	Dividends	Stock Splits
2023-02-13 00:00:00-05:00	108.099998	108.290001	106.550003	107.660004	12341500	0.0	0.0
2023-02-14 00:00:00-05:00	106.820000	108.440002	105.820000	107.660004	10768100	0.0	0.0
2023-02-15 00:00:00-05:00	107.089996	109.269997	106.699997	109.239998	8156000	0.0	0.0
2023-02-16 00:00:00-05:00	107.550003	108.019997	105.699997	105.830002	9479900	0.0	0.0
2023-02-17 00:00:00-05:00	105.029999	105.800003	103.879997	105.220001	9492200	0.0	0.0
2023-02-21 00:00:00-05:00	103.550003	103.730003	101.830002	102.089996	10648200	0.0	0.0
2023-02-22 00:00:00-05:00	101.330002	102.559998	101.089996	101.680000	9885100	0.0	0.0
2023-02-23 00:00:00-05:00	101.449997	102.050003	99.610001	101.730003	10974800	0.0	0.0
2023-02-24 00:00:00-05:00	99.529999	100.430000	98.959999	100.300003	8523000	0.0	0.0
2023-02-27 00:00:00-05:00	100.730003	101.470001	100.059998	100.449997	6641200	0.0	0.0
2023-02-28 00:00:00-05:00	100.449997	100.750000	99.570000	99.610001	8233700	0.0	0.0
2023-03-01 00:00:00-05:00	99.190002	99.629997	98.099998	98.540001	7898300	0.0	0.0
2023-03-02 00:00:00-05:00	97.739998	99.290001	97.419998	98.919998	7776600	0.0	0.0
2023-03-03 00:00:00-05:00	99.669998	101.180000	98.879997	101.139999	7428900	0.0	0.0
2023-03-06 00:00:00-05:00	101.610001	102.699997	100.489998	100.660004	6773300	0.0	0.0
2023-03-07 00:00:00-05:00	100.400002	100.940002	98.889999	99.059998	6951300	0.0	0.0
2023-03-08 00:00:00-05:00	99.000000	99.620003	98.300003	99.300003	5668700	0.0	0.0
2023-03-09 00:00:00-05:00	99.449997	100.050003	95.919998	96.139999	8275600	0.0	0.0
2023-03-10 00:00:00-05:00	95.599998	95.900002	92.830002	93.570000	13964800	0.0	0.0

```
df.shape
```

```
(19, 7)
```