Extracting Stock Data Using a Python Library

A company's stock share is a piece of the company more precisely:

A stock (also known as equity) is a security that represents the ownership of a fraction of a corporation. This entitles the owner of the stock to a proportion of the corporation's assets and profits equal to how much stock they own. Units of stock are called "shares." [1]

An investor can buy a stock and sell it later. If the stock price increases, the investor profits, If it decreases,the investor with incur a loss. Determining the stock price is complex; it depends on the number of outstanding shares, the size of the company's future profits, and much more. People trade stocks throughout the day the stock ticker is a report of the price of a certain stock, updated continuously throughout the trading session by the various stock market exchanges.

You are a data scientist working for a hedge fund; it's your job to determine any suspicious stock activity. In this lab you will extract stock data using a Python library. We will use the yfinance library, it allows us to extract data for stocks returning data in a pandas dataframe. You will use the lab to extract.

Table of Contents

- Using yfinance to Extract Stock Info
- Using yfinance to Extract Historical Share Price Data
- Using yfinance to Extract Historical Dividends Data
- Exercise

Estimated Time Needed: **30 min**

In [1]:
```
!pip install yfinance==0.2.4
#!pip install pandas==1.3.3
```

```
Collecting yfinance==0.2.4
  Downloading yfinance-0.2.4-py2.py3-none-any.whl (51 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 51.4/51.4 kB 8.1 MB/s eta 0:00:
00
Requirement already satisfied: pandas>=1.3.0 in /home/jupyterlab/conda/envs/p
ython/lib/python3.7/site-packages (from yfinance==0.2.4) (1.3.5)
Requirement already satisfied: numpy>=1.16.5 in /home/jupyterlab/conda/envs/p
ython/lib/python3.7/site-packages (from yfinance==0.2.4) (1.21.6)
Requirement already satisfied: requests>=2.26 in /home/jupyterlab/conda/envs/
python/lib/python3.7/site-packages (from yfinance==0.2.4) (2.29.0)
Collecting multitasking>=0.0.7 (from yfinance==0.2.4)
  Downloading multitasking-0.0.11-py3-none-any.whl (8.5 kB)
Collecting lxml>=4.9.1 (from yfinance==0.2.4)
  Using cached lxml-4.9.2-cp37-cp37m-manylinux_2_17_x86_64.manylinux2014_x86_
64.manylinux_2_24_x86_64.whl (6.6 MB)
Collecting appdirs>=1.4.4 (from yfinance==0.2.4)
  Downloading appdirs-1.4.4-py2.py3-none-any.whl (9.6 kB)
Requirement already satisfied: pytz>=2022.5 in /home/jupyterlab/conda/envs/py
thon/lib/python3.7/site-packages (from yfinance==0.2.4) (2023.3)
Collecting frozendict>=2.3.4 (from yfinance==0.2.4)
  Downloading frozendict-2.3.8-cp37-cp37m-manylinux_2_17_x86_64.manylinux2014
_x86_64.whl (101 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 101.7/101.7 kB 17.4 MB/s eta 0:0
0:00
Requirement already satisfied: cryptography>=3.3.2 in /home/jupyterlab/conda/
envs/python/lib/python3.7/site-packages (from yfinance==0.2.4) (38.0.2)
Collecting beautifulsoup4>=4.11.1 (from yfinance==0.2.4)
  Using cached beautifulsoup4-4.12.2-py3-none-any.whl (142 kB)
Requirement already satisfied: html5lib>=1.1 in /home/jupyterlab/conda/envs/p
ython/lib/python3.7/site-packages (from yfinance==0.2.4) (1.1)
Requirement already satisfied: soupsieve>1.2 in /home/jupyterlab/conda/envs/p
ython/lib/python3.7/site-packages (from beautifulsoup4>=4.11.1->yfinance==0.
2.4) (2.3.2.post1)
Requirement already satisfied: cffi>=1.12 in /home/jupyterlab/conda/envs/pyth
on/lib/python3.7/site-packages (from cryptography>=3.3.2->yfinance==0.2.4)
(1.15.1)
Requirement already satisfied: six>=1.9 in /home/jupyterlab/conda/envs/pytho
n/lib/python3.7/site-packages (from html5lib>=1.1->yfinance==0.2.4) (1.16.0)
Requirement already satisfied: webencodings in /home/jupyterlab/conda/envs/py
thon/lib/python3.7/site-packages (from html5lib>=1.1->yfinance==0.2.4) (0.5.
1)
Requirement already satisfied: python-dateutil>=2.7.3 in /home/jupyterlab/con
da/envs/python/lib/python3.7/site-packages (from pandas>=1.3.0->yfinance==0.
2.4) (2.8.2)
Requirement already satisfied: charset-normalizer<4,>=2 in /home/jupyterlab/c
onda/envs/python/lib/python3.7/site-packages (from requests>=2.26->yfinance==
0.2.4) (3.1.0)
Requirement already satisfied: idna<4,>=2.5 in /home/jupyterlab/conda/envs/py
thon/lib/python3.7/site-packages (from requests>=2.26->yfinance==0.2.4) (3.4)
Requirement already satisfied: urllib3<1.27,>=1.21.1 in /home/jupyterlab/cond
a/envs/python/lib/python3.7/site-packages (from requests>=2.26->yfinance==0.
2.4) (1.26.15)
Requirement already satisfied: certifi>=2017.4.17 in /home/jupyterlab/conda/e
nvs/python/lib/python3.7/site-packages (from requests>=2.26->yfinance==0.2.4)
(2023.5.7)
Requirement already satisfied: pycparser in /home/jupyterlab/conda/envs/pytho
```

```
n/lib/python3.7/site-packages (from cffi>=1.12->cryptography>=3.3.2->yfinance
==0.2.4) (2.21)
Installing collected packages: multitasking, appdirs, lxml, frozendict, beaut
ifulsoup4, yfinance
  Attempting uninstall: lxml
    Found existing installation: lxml 4.6.4
    Uninstalling lxml-4.6.4:
      Successfully uninstalled lxml-4.6.4
  Attempting uninstall: beautifulsoup4
    Found existing installation: beautifulsoup4 4.10.0
    Uninstalling beautifulsoup4-4.10.0:
      Successfully uninstalled beautifulsoup4-4.10.0
Successfully installed appdirs-1.4.4 beautifulsoup4-4.12.2 frozendict-2.3.8 l
xml-4.9.2 multitasking-0.0.11 yfinance-0.2.4
```

In [2]:
```
import yfinance as yf
import pandas as pd
```

Using the yfinance Library to Extract Stock Data

Using the `Ticker` module we can create an object that will allow us to access functions to extract data. To do this we need to provide the ticker symbol for the stock, here the company is Apple and the ticker symbol is `AAPL`.

In [3]:
```
apple = yf.Ticker("AAPL")
```

Now we can access functions and variables to extract the type of data we need. You can view them and what they represent here https://aroussi.com/post/python-yahoo-finance.

In [4]:
```
!wget https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM
```

```
--2023-05-26 06:26:44--  https://cf-courses-data.s3.us.cloud-object-storage.a
ppdomain.cloud/IBMDeveloperSkillsNetwork-PY0220EN-SkillsNetwork/data/apple.js
on
Resolving cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud (cf-cour
ses-data.s3.us.cloud-object-storage.appdomain.cloud)... 169.63.118.104
Connecting to cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud (cf-
courses-data.s3.us.cloud-object-storage.appdomain.cloud)|169.63.118.104|:44
3... connected.
HTTP request sent, awaiting response... 200 OK
Length: 5699 (5.6K) [application/json]
Saving to: ‘apple.json’

apple.json          100%[===================>]   5.57K  --.-KB/s    in 0s

2023-05-26 06:26:44 (38.6 MB/s) - ‘apple.json’ saved [5699/5699]
```

Stock Info

Using the attribute `info` we can extract information about the stock as a Python dictionary.

In [5]:

```
import json
with open('apple.json') as json_file:
    apple_info = json.load(json_file)
    # Print the type of data variable
    #print("Type:", type(apple_info))
apple_info
```

Out[5]:

```
{'zip': '95014',
 'sector': 'Technology',
 'fullTimeEmployees': 100000,
 'longBusinessSummary': 'Apple Inc. designs, manufactures, and markets smar
tphones, personal computers, tablets, wearables, and accessories worldwide.
It also sells various related services. In addition, the company offers iPh
one, a line of smartphones; Mac, a line of personal computers; iPad, a line
of multi-purpose tablets; AirPods Max, an over-ear wireless headphone; and
wearables, home, and accessories comprising AirPods, Apple TV, Apple Watch,
Beats products, HomePod, and iPod touch. Further, it provides AppleCare sup
port services; cloud services store services; and operates various platform
s, including the App Store that allow customers to discover and download ap
plications and digital content, such as books, music, video, games, and pod
casts. Additionally, the company offers various services, such as Apple Arc
ade, a game subscription service; Apple Music, which offers users a curated
listening experience with on-demand radio stations; Apple News+, a subscrip
tion news and magazine service; Apple TV+, which offers exclusive original
content; Apple Card, a co-branded credit card; and Apple Pay, a cashless pa
yment service, as well as licenses its intellectual property. The company s
erves consumers, and small and mid-sized businesses; and the education, ent
erprise, and government markets. It distributes third-party applications fo
r its products through the App Store. The company also sells its products t
hrough its retail and online stores, and direct sales force; and third-part
y cellular network carriers, wholesalers, retailers, and resellers. Apple I
nc. was incorporated in 1977 and is headquartered in Cupertino, Californi
a.',
 'city': 'Cupertino',
 'phone': '408 996 1010',
 'state': 'CA',
 'country': 'United States',
 'companyOfficers': [],
 'website': 'https://www.apple.com',
 'maxAge': 1,
 'address1': 'One Apple Park Way',
 'industry': 'Consumer Electronics',
 'ebitdaMargins': 0.33890998,
 'profitMargins': 0.26579002,
 'grossMargins': 0.43019,
 'operatingCashflow': 112241000448,
 'revenueGrowth': 0.112,
 'operatingMargins': 0.309,
 'ebitda': 128217997312,
 'targetLowPrice': 160,
 'recommendationKey': 'buy',
 'grossProfits': 152836000000,
 'freeCashflow': 80153247744,
 'targetMedianPrice': 199.5,
 'currentPrice': 177.77,
 'earningsGrowth': 0.25,
 'currentRatio': 1.038,
 'returnOnAssets': 0.19875,
 'numberOfAnalystOpinions': 44,
 'targetMeanPrice': 193.53,
 'debtToEquity': 170.714,
 'returnOnEquity': 1.45567,
 'targetHighPrice': 215,
```

```
'totalCash': 63913000960,
'totalDebt': 122797998080,
'totalRevenue': 378323009536,
'totalCashPerShare': 3.916,
'financialCurrency': 'USD',
'revenuePerShare': 22.838,
'quickRatio': 0.875,
'recommendationMean': 1.8,
'exchange': 'NMS',
'shortName': 'Apple Inc.',
'longName': 'Apple Inc.',
'exchangeTimezoneName': 'America/New_York',
'exchangeTimezoneShortName': 'EDT',
'isEsgPopulated': False,
'gmtOffSetMilliseconds': '-14400000',
'quoteType': 'EQUITY',
'symbol': 'AAPL',
'messageBoardId': 'finmb_24937',
'market': 'us_market',
'annualHoldingsTurnover': None,
'enterpriseToRevenue': 7.824,
'beta3Year': None,
'enterpriseToEbitda': 23.086,
'52WeekChange': 0.4549594,
'morningStarRiskRating': None,
'forwardEps': 6.56,
'revenueQuarterlyGrowth': None,
'sharesOutstanding': 16319399936,
'fundInceptionDate': None,
'annualReportExpenseRatio': None,
'totalAssets': None,
'bookValue': 4.402,
'sharesShort': 111286790,
'sharesPercentSharesOut': 0.0068,
'fundFamily': None,
'lastFiscalYearEnd': 1632528000,
'heldPercentInstitutions': 0.59397,
'netIncomeToCommon': 100554997760,
'trailingEps': 6.015,
'lastDividendValue': 0.22,
'SandP52WeekChange': 0.15217662,
'priceToBook': 40.38392,
'heldPercentInsiders': 0.0007,
'nextFiscalYearEnd': 1695600000,
'yield': None,
'mostRecentQuarter': 1640390400,
'shortRatio': 1.21,
'sharesShortPreviousMonthDate': 1644883200,
'floatShares': 16302795170,
'beta': 1.185531,
'enterpriseValue': 2959991898112,
'priceHint': 2,
'threeYearAverageReturn': None,
'lastSplitDate': 1598832000,
'lastSplitFactor': '4:1',
'legalType': None,
```

```
'lastDividendDate': 1643932800,
'morningStarOverallRating': None,
'earningsQuarterlyGrowth': 0.204,
'priceToSalesTrailing12Months': 7.668314,
'dateShortInterest': 1647302400,
'pegRatio': 1.94,
'ytdReturn': None,
'forwardPE': 27.099087,
'lastCapGain': None,
'shortPercentOfFloat': 0.0068,
'sharesShortPriorMonth': 108944701,
'impliedSharesOutstanding': 0,
'category': None,
'fiveYearAverageReturn': None,
'previousClose': 178.96,
'regularMarketOpen': 178.55,
'twoHundredDayAverage': 156.03505,
'trailingAnnualDividendYield': 0.004833482,
'payoutRatio': 0.1434,
'volume24Hr': None,
'regularMarketDayHigh': 179.61,
'navPrice': None,
'averageDailyVolume10Day': 93823630,
'regularMarketPreviousClose': 178.96,
'fiftyDayAverage': 166.498,
'trailingAnnualDividendRate': 0.865,
'open': 178.55,
'toCurrency': None,
'averageVolume10days': 93823630,
'expireDate': None,
'algorithm': None,
'dividendRate': 0.88,
'exDividendDate': 1643932800,
'circulatingSupply': None,
'startDate': None,
'regularMarketDayLow': 176.7,
'currency': 'USD',
'trailingPE': 29.55445,
'regularMarketVolume': 92633154,
'lastMarket': None,
'maxSupply': None,
'openInterest': None,
'marketCap': 2901099675648,
'volumeAllCurrencies': None,
'strikePrice': None,
'averageVolume': 95342043,
'dayLow': 176.7,
'ask': 178.53,
'askSize': 800,
'volume': 92633154,
'fiftyTwoWeekHigh': 182.94,
'fromCurrency': None,
'fiveYearAvgDividendYield': 1.13,
'fiftyTwoWeekLow': 122.25,
'bid': 178.4,
'tradeable': False,
```

```
 'dividendYield': 0.005,
 'bidSize': 3200,
 'dayHigh': 179.61,
 'regularMarketPrice': 177.77,
 'preMarketPrice': 178.38,
 'logo_url': 'https://logo.clearbit.com/apple.com'}
```

We can get the `'country'` using the key country

In [6]:
```
apple_info['country']
```

Out[6]:
```
'United States'
```

Extracting Share Price

A share is the single smallest part of a company's stock that you can buy, the prices of these shares fluctuate over time. Using the `history()` method we can get the share price of the stock over a certain period of time. Using the `period` parameter we can set how far back from the present to get data. The options for `period` are 1 day (1d), 5d, 1 month (1mo) , 3mo, 6mo, 1 year (1y), 2y, 5y, 10y, ytd, and max.

In [7]:
```
apple_share_price_data = apple.history(period="max")
```

The format that the data is returned in is a Pandas DataFrame. With the `Date` as the index the share `Open`, `High`, `Low`, `Close`, `Volume`, and `Stock Splits` are given for each day.

In [8]:
```
apple_share_price_data.head()
```

Out[8]:

Date	Open	High	Low	Close	Volume	Dividends	Stock Splits
1980-12-12 00:00:00-05:00	0.099584	0.100017	0.099584	0.099584	469033600	0.0	0.0
1980-12-15 00:00:00-05:00	0.094821	0.094821	0.094388	0.094388	175884800	0.0	0.0
1980-12-16 00:00:00-05:00	0.087893	0.087893	0.087461	0.087461	105728000	0.0	0.0
1980-12-17 00:00:00-05:00	0.089625	0.090058	0.089625	0.089625	86441600	0.0	0.0
1980-12-18 00:00:00-05:00	0.092224	0.092657	0.092224	0.092224	73449600	0.0	0.0

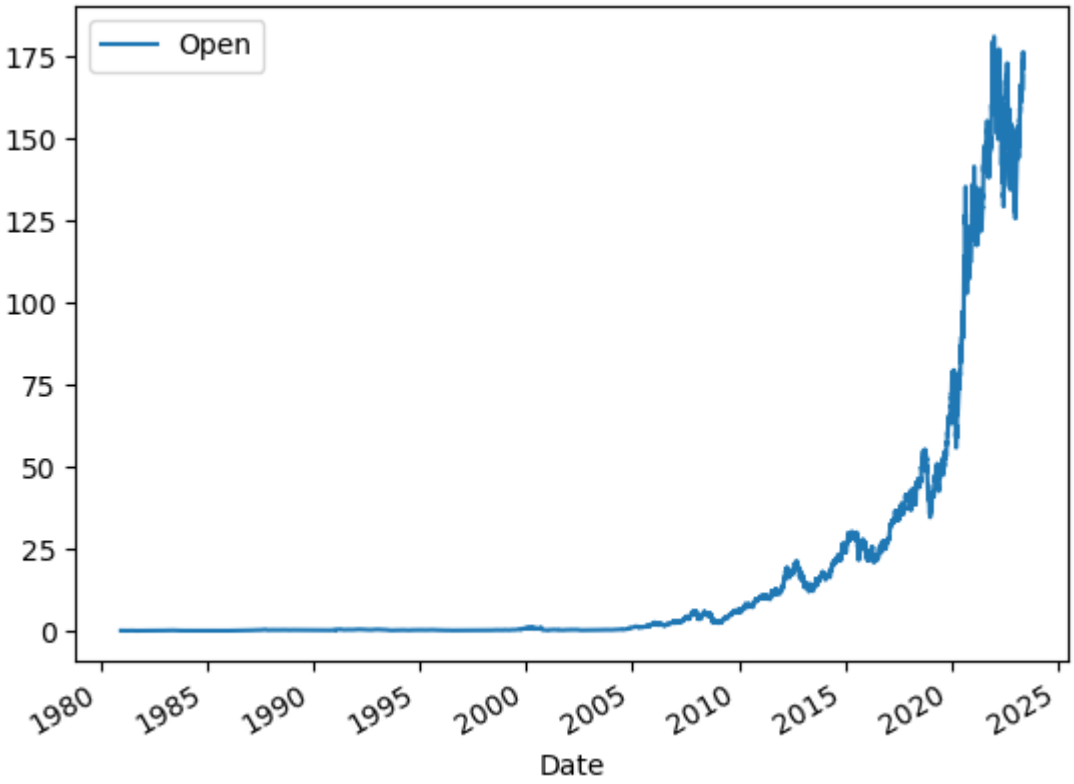
We can reset the index of the DataFrame with the `reset_index` function. We also set the `inplace` paramter to `True` so the change takes place to the DataFrame itself.

In [9]:
```
apple_share_price_data.reset_index(inplace=True)
```

We can plot the `Open` price against the `Date`:

In [10]:
```
apple_share_price_data.plot(x="Date", y="Open")
```

Out[10]:
```
<AxesSubplot:xlabel='Date'>
```



Extracting Dividends

Dividends are the distribution of a companys profits to shareholders. In this case they are defined as an amount of money returned per share an investor owns. Using the variable `dividends` we can get a dataframe of the data. The period of the data is given by the period defined in the 'history` function.

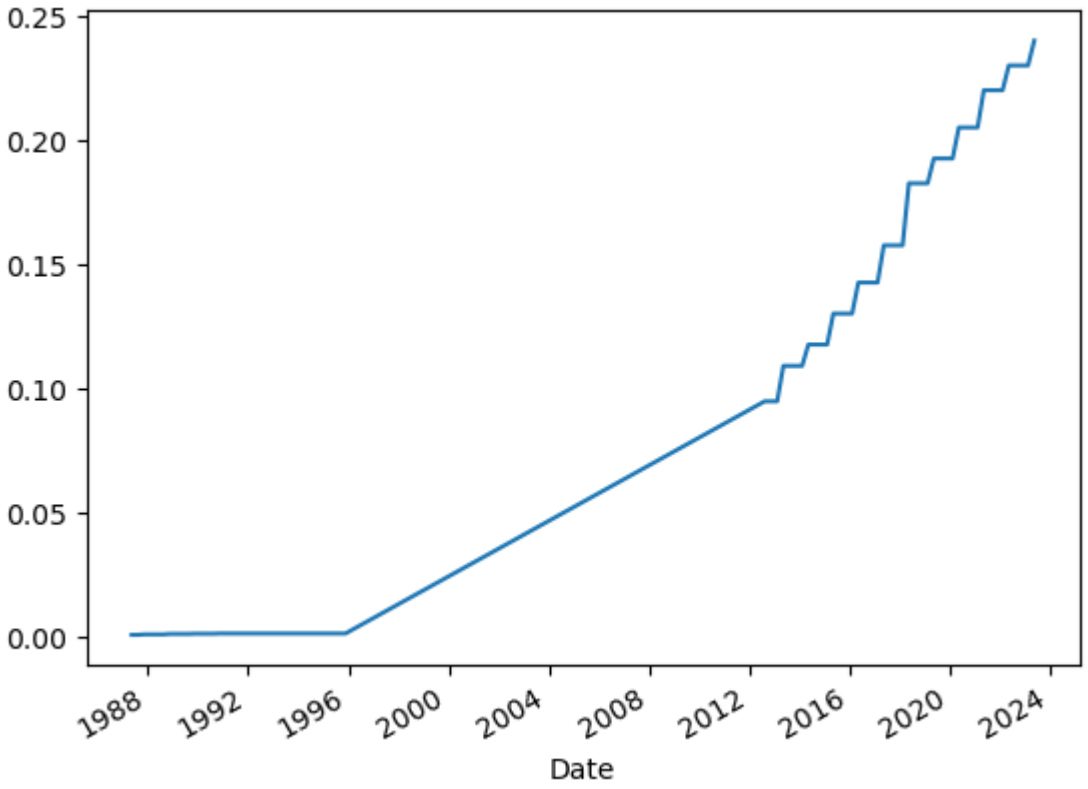
In [12]:
```
apple.dividends
```

Out[12]:
```
Date
1987-05-11 00:00:00-04:00    0.000536
1987-08-10 00:00:00-04:00    0.000536
1987-11-17 00:00:00-05:00    0.000714
1988-02-12 00:00:00-05:00    0.000714
1988-05-16 00:00:00-04:00    0.000714
                               ...
2022-05-06 00:00:00-04:00    0.230000
2022-08-05 00:00:00-04:00    0.230000
2022-11-04 00:00:00-04:00    0.230000
2023-02-10 00:00:00-05:00    0.230000
2023-05-12 00:00:00-04:00    0.240000
Name: Dividends, Length: 79, dtype: float64
```

We can plot the dividends overtime:

In [13]:
```
apple.dividends.plot()
```

Out[13]:
```
<AxesSubplot:xlabel='Date'>
```



Exercise

Now using the `Ticker` module create an object for AMD (Advanced Micro Devices) with the ticker symbol is `AMD` called; name the object `amd`.

In [21]:
```
amd = yf.Ticker("AMD")
```

In [22]:
```
!wget https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM
```

```
--2023-05-26 06:35:35--  https://cf-courses-data.s3.us.cloud-object-storage.a
ppdomain.cloud/IBMDeveloperSkillsNetwork-PY0220EN-SkillsNetwork/data/amd.json
Resolving cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud (cf-cour
ses-data.s3.us.cloud-object-storage.appdomain.cloud)... 169.63.118.104
Connecting to cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud (cf-
courses-data.s3.us.cloud-object-storage.appdomain.cloud)|169.63.118.104|:44
3... connected.
HTTP request sent, awaiting response... 200 OK
Length: 5838 (5.7K) [application/json]
Saving to: ‘amd.json.1’

amd.json.1          100%[===================>]   5.70K  --.-KB/s    in 0s

2023-05-26 06:35:35 (64.7 MB/s) - ‘amd.json.1’ saved [5838/5838]
```

In [23]:
```
import json
with open('amd.json') as json_file:
    amd_info = json.load(json_file)
    # Print the type of data variable
```

```
    #print("Type:", type(apple_info))
amd_info
```

Out[23]:

```
{'zip': '95054',
 'sector': 'Technology',
 'fullTimeEmployees': 15500,
 'longBusinessSummary': 'Advanced Micro Devices, Inc. operates as a semicon
ductor company worldwide. The company operates in two segments, Computing a
nd Graphics; and Enterprise, Embedded and Semi-Custom. Its products include
x86 microprocessors as an accelerated processing unit, chipsets, discrete a
nd integrated graphics processing units (GPUs), data center and professiona
l GPUs, and development services; and server and embedded processors, and s
emi-custom System-on-Chip (SoC) products, development services, and technol
ogy for game consoles. The company provides processors for desktop and note
book personal computers under the AMD Ryzen, AMD Ryzen PRO, Ryzen Threadrip
per, Ryzen Threadripper PRO, AMD Athlon, AMD Athlon PRO, AMD FX, AMD A-Seri
es, and AMD PRO A-Series processors brands; discrete GPUs for desktop and n
otebook PCs under the AMD Radeon graphics, AMD Embedded Radeon graphics bra
nds; and professional graphics products under the AMD Radeon Pro and AMD Fi
rePro graphics brands. It also offers Radeon Instinct, Radeon PRO V-series,
and AMD Instinct accelerators for servers; chipsets under the AMD trademar
k; microprocessors for servers under the AMD EPYC; embedded processor solut
ions under the AMD Athlon, AMD Geode, AMD Ryzen, AMD EPYC, AMD R-Series, an
d G-Series processors brands; and customer-specific solutions based on AMD
CPU, GPU, and multi-media technologies, as well as semi-custom SoC product
s. It serves original equipment manufacturers, public cloud service provide
rs, original design manufacturers, system integrators, independent distribu
tors, online retailers, and add-in-board manufacturers through its direct s
ales force, independent distributors, and sales representatives. The compan
y was incorporated in 1969 and is headquartered in Santa Clara, Californi
a.',
 'city': 'Santa Clara',
 'phone': '408 749 4000',
 'state': 'CA',
 'country': 'United States',
 'companyOfficers': [],
 'website': 'https://www.amd.com',
 'maxAge': 1,
 'address1': '2485 Augustine Drive',
 'industry': 'Semiconductors',
 'ebitdaMargins': 0.24674,
 'profitMargins': 0.19240999,
 'grossMargins': 0.48248002,
 'operatingCashflow': 3520999936,
 'revenueGrowth': 0.488,
 'operatingMargins': 0.22198,
 'ebitda': 4055000064,
 'targetLowPrice': 107,
 'recommendationKey': 'buy',
 'grossProfits': 7929000000,
 'freeCashflow': 3122749952,
 'targetMedianPrice': 150,
 'currentPrice': 119.22,
 'earningsGrowth': -0.454,
 'currentRatio': 2.024,
 'returnOnAssets': 0.21327,
 'numberOfAnalystOpinions': 38,
 'targetMeanPrice': 152.02,
 'debtToEquity': 9.764,
```

```
'returnOnEquity': 0.47428,
'targetHighPrice': 200,
'totalCash': 3608000000,
'totalDebt': 732000000,
'totalRevenue': 16433999872,
'totalCashPerShare': 3.008,
'financialCurrency': 'USD',
'revenuePerShare': 13.548,
'quickRatio': 1.49,
'recommendationMean': 2.2,
'exchange': 'NMS',
'shortName': 'Advanced Micro Devices, Inc.',
'longName': 'Advanced Micro Devices, Inc.',
'exchangeTimezoneName': 'America/New_York',
'exchangeTimezoneShortName': 'EDT',
'isEsgPopulated': False,
'gmtOffSetMilliseconds': '-14400000',
'quoteType': 'EQUITY',
'symbol': 'AMD',
'messageBoardId': 'finmb_168864',
'market': 'us_market',
'annualHoldingsTurnover': None,
'enterpriseToRevenue': 8.525,
'beta3Year': None,
'enterpriseToEbitda': 34.551,
'52WeekChange': 0.51966953,
'morningStarRiskRating': None,
'forwardEps': 4.72,
'revenueQuarterlyGrowth': None,
'sharesOutstanding': 1627360000,
'fundInceptionDate': None,
'annualReportExpenseRatio': None,
'totalAssets': None,
'bookValue': 6.211,
'sharesShort': 27776129,
'sharesPercentSharesOut': 0.0171,
'fundFamily': None,
'lastFiscalYearEnd': 1640390400,
'heldPercentInstitutions': 0.52896,
'netIncomeToCommon': 3161999872,
'trailingEps': 2.57,
'lastDividendValue': 0.005,
'SandP52WeekChange': 0.15217662,
'priceToBook': 19.194977,
'heldPercentInsiders': 0.00328,
'nextFiscalYearEnd': 1703462400,
'yield': None,
'mostRecentQuarter': 1640390400,
'shortRatio': 0.24,
'sharesShortPreviousMonthDate': 1644883200,
'floatShares': 1193798619,
'beta': 1.848425,
'enterpriseValue': 140104957952,
'priceHint': 2,
'threeYearAverageReturn': None,
'lastSplitDate': 966902400,
```

```
'lastSplitFactor': '2:1',
'legalType': None,
'lastDividendDate': 798940800,
'morningStarOverallRating': None,
'earningsQuarterlyGrowth': -0.453,
'priceToSalesTrailing12Months': 11.805638,
'dateShortInterest': 1647302400,
'pegRatio': 0.99,
'ytdReturn': None,
'forwardPE': 25.258476,
'lastCapGain': None,
'shortPercentOfFloat': 0.0171,
'sharesShortPriorMonth': 88709340,
'impliedSharesOutstanding': 0,
'category': None,
'fiveYearAverageReturn': None,
'previousClose': 123.23,
'regularMarketOpen': 123.04,
'twoHundredDayAverage': 116.6998,
'trailingAnnualDividendYield': 0,
'payoutRatio': 0,
'volume24Hr': None,
'regularMarketDayHigh': 125.66,
'navPrice': None,
'averageDailyVolume10Day': 102167370,
'regularMarketPreviousClose': 123.23,
'fiftyDayAverage': 115.95,
'trailingAnnualDividendRate': 0,
'open': 123.04,
'toCurrency': None,
'averageVolume10days': 102167370,
'expireDate': None,
'algorithm': None,
'dividendRate': None,
'exDividendDate': 798940800,
'circulatingSupply': None,
'startDate': None,
'regularMarketDayLow': 118.59,
'currency': 'USD',
'trailingPE': 46.389107,
'regularMarketVolume': 99476946,
'lastMarket': None,
'maxSupply': None,
'openInterest': None,
'marketCap': 194013855744,
'volumeAllCurrencies': None,
'strikePrice': None,
'averageVolume': 102428813,
'dayLow': 118.59,
'ask': 117.24,
'askSize': 1100,
'volume': 99476946,
'fiftyTwoWeekHigh': 164.46,
'fromCurrency': None,
'fiveYearAvgDividendYield': None,
'fiftyTwoWeekLow': 72.5,
```

```
'bid': 117.24,
'tradeable': False,
'dividendYield': None,
'bidSize': 900,
'dayHigh': 125.66,
'regularMarketPrice': 119.22,
'preMarketPrice': 116.98,
'logo_url': 'https://logo.clearbit.com/amd.com'}
```

Question 1 Use the key `'country'` to find the country the stock belongs to, remember it as it will be a quiz question.

In [24]:
```
amd_info['country']
```

Out[24]:
```
'United States'
```

Question 2 Use the key `'sector'` to find the sector the stock belongs to, remember it as it will be a quiz question.

In [25]:
```
amd_info['sector']
```

Out[25]:
```
'Technology'
```

Question 3 Obtain stock data for AMD using the `history` function, set the `period` to max. Find the `Volume` traded on the first day (first row).

In [26]:
```
amd_share_price_data = amd.history(period="max")
```

In [27]:
```
amd_share_price_data.head()
```

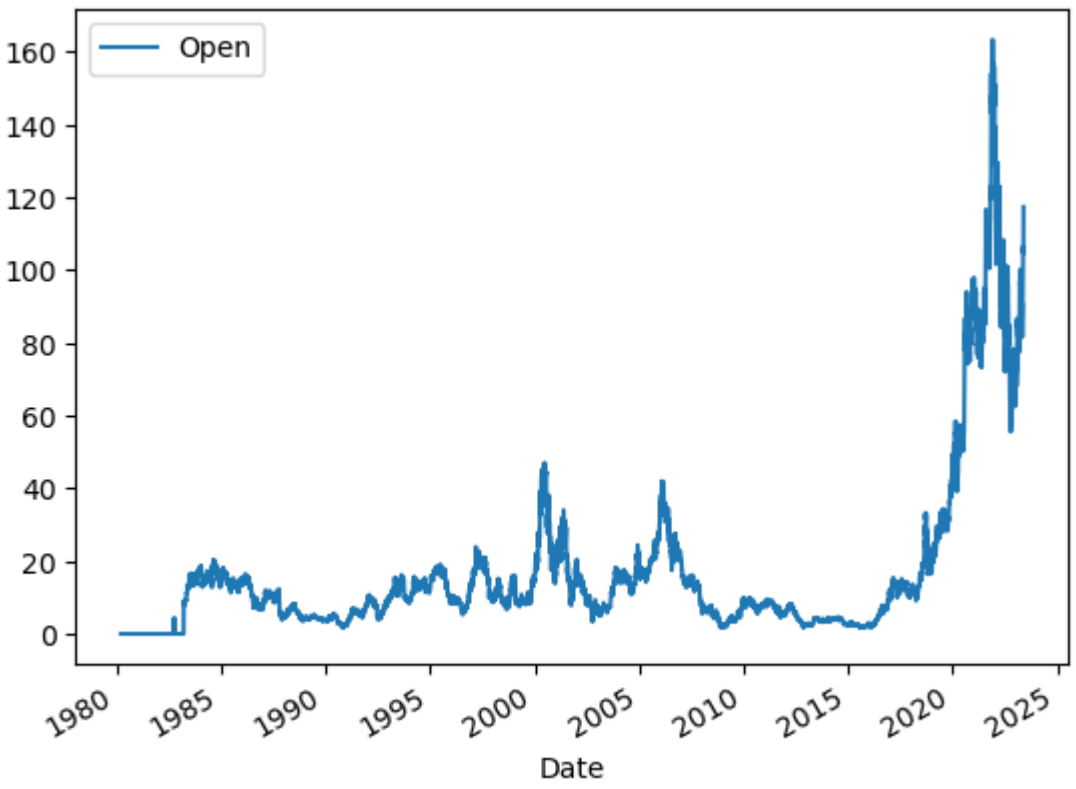
Out[27]:

Date	Open	High	Low	Close	Volume	Dividends	Stock Splits
1980-03-17 00:00:00-05:00	0.0	3.302083	3.125000	3.145833	219600	0.0	0.0
1980-03-18 00:00:00-05:00	0.0	3.125000	2.937500	3.031250	727200	0.0	0.0
1980-03-19 00:00:00-05:00	0.0	3.083333	3.020833	3.041667	295200	0.0	0.0
1980-03-20 00:00:00-05:00	0.0	3.062500	3.010417	3.010417	159600	0.0	0.0
1980-03-21 00:00:00-05:00	0.0	3.020833	2.906250	2.916667	130800	0.0	0.0

In [28]:
```
amd_share_price_data.reset_index(inplace=True)
```

In [29]:
```
amd_share_price_data.plot(x="Date", y="Open")
```

Out[29]: <AxesSubplot:xlabel='Date'>



About the Authors:

Joseph Santarcangelo has a PhD in Electrical Engineering, his research focused on using machine learning, signal processing, and computer vision to determine how videos impact human cognition. Joseph has been working for IBM since he completed his PhD.

Azim Hirjani

Change Log

Date (YYYY-MM-DD)	Version	Changed By	Change Description
2020-11-10	1.1	Malika Singla	Deleted the Optional part
2020-08-27	1.0	Malika Singla	Added lab to GitLab
