This exercise will require you to pull some data from the Qunadl API. Qaundl is currently the most widely used aggregator of financial market data.

As a first step, you will need to register a free account on the http://www.quandl.com website.

After you register, you will be provided with a unique API key, that you should store:

In [15]:
```
# Store the API key as a string - according to PEP8, constants are always named in all upper case
API_KEY = '9fLHcK67YYGzdQ2kNtHs'
```

Qaundl has a large number of data sources, but, unfortunately, most of them require a Premium subscription. Still, there are also a good number of free datasets.

For this mini project, we will focus on equities data from the Frankfurt Stock Exhange (FSE), which is available for free. We'll try and analyze the stock prices of a company called Carl Zeiss Meditec, which manufactures tools for eye examinations, as well as medical lasers for laser eye surgery: https://www.zeiss.com/meditec/int/home.html. The company is listed under the stock ticker AFX_X.

You can find the detailed Quandl API instructions here: https://docs.quandl.com/docs/time-series

While there is a dedicated Python package for connecting to the Quandl API, we would prefer that you use the *requests* package, which can be easily downloaded using *pip* or *conda*. You can find the documentation for the package here: http://docs.python-requests.org/en/master/

Finally, apart from the *requests* package, you are encouraged to not use any third party Python packages, such as *pandas*, and instead focus on what's available in the Python Standard Library (the *collections* module might come in handy: https://pymotw.com/3/collections/). Also, since you won't have access to DataFrames, you are encouraged to us Python's native data structures - preferably dictionaries, though some questions can also be answered using lists. You can read more on these data structures here: https://docs.python.org/3/tutorial/datastructures.html

Keep in mind that the JSON responses you will be getting from the API map almost one-to-one to Python's dictionaries. Unfortunately, they can be very nested, so make sure you read up on indexing dictionaries in the documentation provided above.

In [16]:
```
# First, import the relevant modules
import requests
import pandas as pd
import json
```

In [17]:
```
# Now, call the Quandl API and pull out a small sample of the data (only one day) to get a glimpse
# into the JSON structure that will be returned
r = requests.get('https://www.quandl.com/api/v3/datasets/FSE/AFX_X.json?api_key=9fLHcK67YYGzdQ2kNtHs
&start_date=2017-01-02&end_date=2017-12-29')
r
```

Out[17]:
```
<Response [200]>
```

These are your tasks for this mini project:

1. Collect data from the Franfurt Stock Exchange, for the ticker AFX_X, for the whole year 2017 (keep in mind that the date format is YYYY-MM-DD).
2. Convert the returned JSON object into a Python dictionary.
3. Calculate what the highest and lowest opening prices were for the stock in this period.
4. What was the largest change in any one day (based on High and Low price)?
5. What was the largest change between any two days (based on Closing Price)?
6. What was the average daily trading volume during this year?
7. (Optional) What was the median trading volume during this year. (Note: you may need to implement your own function for calculating the median.)

1 Collected Data From the API

In [18]:
```
a = r.json()
```

2 Converting json into python dict

In [19]:
```
data = dict(a)
```

In [20]:
```
df = pd.DataFrame(data['dataset']['data'])
df
```

Out[20]:

	0	1	2	3	4	5	6	7	8	9	10
0	2017-12-29	51.76	51.94	51.45	51.76	NaN	34640.0	1792304.0	None	None	None
1	2017-12-28	51.65	51.82	51.43	51.60	NaN	40660.0	2099024.0	None	None	None
2	2017-12-27	51.45	51.89	50.76	51.82	NaN	57452.0	2957018.0	None	None	None
3	2017-12-22	51.05	51.50	50.92	51.32	NaN	71165.0	3641949.0	None	None	None
4	2017-12-21	51.16	51.52	50.90	51.40	NaN	120649.0	6179433.0	None	None	None
5	2017-12-20	51.88	52.04	51.20	51.27	NaN	50587.0	2610258.0	None	None	None
6	2017-12-19	52.73	52.73	51.07	51.66	NaN	137313.0	7102361.0	None	None	None
7	2017-12-18	52.37	52.75	51.61	52.62	NaN	129733.0	6770499.0	None	None	None
8	2017-12-15	52.70	52.70	51.64	52.01	NaN	204080.0	10596319.0	None	None	None
9	2017-12-14	53.11	53.54	52.15	52.67	NaN	132981.0	7016953.0	None	None	None
10	2017-12-13	52.64	53.35	52.48	53.09	NaN	128434.0	6801159.0	None	None	None
11	2017-12-12	52.29	53.10	51.82	52.43	NaN	87911.0	4615924.0	None	None	None
12	2017-12-11	52.28	52.45	51.26	52.14	NaN	71817.0	3724193.0	None	None	None
13	2017-12-08	51.50	52.83	51.28	52.12	NaN	109157.0	5690648.0	None	None	None
14	2017-12-07	50.89	51.47	50.81	51.47	NaN	48123.0	2463848.0	None	None	None
15	2017-12-06	50.80	51.11	50.39	50.89	NaN	88730.0	4504075.0	None	None	None
16	2017-12-05	51.21	51.38	50.40	51.25	NaN	83023.0	4231971.0	None	None	None
17	2017-12-04	49.50	51.23	49.50	51.14	NaN	94385.0	4800027.0	None	None	None
18	2017-12-01	49.52	50.49	49.17	49.86	NaN	101733.0	5065932.0	None	None	None
19	2017-11-30	48.64	49.84	48.28	49.70	NaN	123019.0	6085171.0	None	None	None
20	2017-11-29	49.64	49.64	48.70	48.75	NaN	67342.0	3292223.0	None	None	None
21	2017-11-28	49.09	49.89	49.03	49.25	NaN	42669.0	2107358.0	None	None	None
22	2017-11-27	49.13	49.73	48.96	49.20	NaN	102180.0	5055762.0	None	None	None
23	2017-11-24	49.11	49.41	48.87	49.11	NaN	50350.0	2472842.0	None	None	None
24	2017-11-23	48.80	49.46	48.45	49.20	NaN	38834.0	1909352.0	None	None	None
25	2017-11-22	48.40	49.61	48.39	48.80	NaN	91142.0	4478093.0	None	None	None
26	2017-11-21	47.25	48.59	46.78	48.39	NaN	78502.0	3782098.0	None	None	None
27	2017-11-20	46.57	47.38	46.54	47.04	NaN	97252.0	4563515.0	None	None	None
28	2017-11-17	47.03	47.15	46.80	46.84	NaN	54107.0	2540820.0	None	None	None
29	2017-11-16	47.09	47.23	46.55	47.03	NaN	89373.0	4195732.0	None	None	None
...	...	...	...	...	...	...	...	...	...	...	...
225	2017-02-10	36.65	37.50	36.57	37.06	NaN	115843.0	4291017.0	None	None	None
226	2017-02-09	36.20	36.25	35.77	36.25	NaN	67781.0	2445428.0	None	None	None
227	2017-02-08	35.98	36.14	35.84	36.05	NaN	39731.0	1431205.0	None	None	None
228	2017-02-07	35.56	36.05	35.36	35.89	NaN	67410.0	2410818.0	None	None	None
229	2017-02-06	36.06	36.15	35.60	35.64	NaN	41911.0	1496794.0	None	None	None
230	2017-02-03	36.02	36.20	35.73	36.10	NaN	40705.0	1464712.0	None	None	None
231	2017-02-02	35.95	36.20	35.70	36.07	NaN	54279.0	1953176.0	None	None	None
232	2017-02-01	34.75	36.00	34.75	35.94	NaN	85137.0	3038172.0	None	None	None
233	2017-01-31	35.24	35.24	34.56	34.56	NaN	63371.0	2199583.0	None	None	None
234	2017-01-30	35.38	35.59	34.95	35.15	NaN	69603.0	2457762.0	None	None	None
235	2017-01-27	34.83	35.43	34.81	35.30	NaN	69657.0	2444913.0	None	None	None
236	2017-01-26	35.07	35.58	34.80	34.89	NaN	64103.0	2249375.0	None	None	None
237	2017-01-25	34.42	34.86	34.03	34.83	NaN	56240.0	1947147.0	None	None	None
238	2017-01-24	34.00	34.35	33.85	34.22	NaN	48797.0	1666086.0	None	None	None
239	2017-01-23	34.04	34.12	33.62	34.06	NaN	55333.0	1877957.0	None	None	None
240	2017-01-20	34.54	34.59	34.05	34.17	NaN	80246.0	2743474.0	None	None	None
241	2017-01-19	35.04	35.04	34.42	34.50	NaN	73105.0	2526731.0	None	None	None
242	2017-01-18	35.04	35.51	34.80	34.90	NaN	65931.0	2311608.0	None	None	None
243	2017-01-17	35.06	35.19	34.79	34.99	NaN	39195.0	1369857.0	None	None	None
244	2017-01-16	34.85	35.24	34.56	35.07	NaN	47879.0	1678679.0	None	None	None
245	2017-01-13	34.98	34.98	34.60	34.85	NaN	59367.0	2065534.0	None	None	None
246	2017-01-12	35.38	35.38	34.31	34.90	NaN	163860.0	5703427.0	None	None	None
247	2017-01-11	34.95	36.00	34.84	35.42	NaN	123530.0	4369079.0	None	None	None
248	2017-01-10	34.80	34.98	34.46	34.91	NaN	43976.0	1528055.0	None	None	None
249	2017-01-09	35.29	35.35	34.43	34.67	NaN	62225.0	2157182.0	None	None	None
250	2017-01-06	34.91	35.21	34.91	35.04	NaN	27507.0	964046.0	None	None	None
251	2017-01-05	35.02	35.20	34.73	35.06	NaN	48412.0	1692326.0	None	None	None
252	2017-01-04	35.48	35.51	34.75	35.19	NaN	54408.0	1906810.0	None	None	None
253	2017-01-03	35.90	35.93	35.34	35.48	NaN	70618.0	2515473.0	None	None	None
254	2017-01-02	34.99	35.94	34.99	35.80	NaN	44700.0	1590561.0	None	None	None

255 rows × 11 columns

In [21]:
```
name = data['dataset']['column_names']
name
```

Out[21]:
```
['Date',
 'Open',
 'High',
 'Low',
 'Close',
 'Change',
 'Traded Volume',
 'Turnovre',
 'Last Price of the Day',
 'Daily Traded Units',
 'Daily Turnovre']
```

In [22]:
```
df.columns = name
df.head()
```

Out[22]:

	Date	Open	High	Low	Close	Change	Traded Volume	Turnovre	Last Price of the Day	Daily Traded Units	Daily Turnovre
0	2017-12-29	51.76	51.94	51.45	51.76	NaN	34640.0	1792304.0	None	None	None
1	2017-12-28	51.65	51.82	51.43	51.60	NaN	40660.0	2099024.0	None	None	None
2	2017-12-27	51.45	51.89	50.76	51.82	NaN	57452.0	2957018.0	None	None	None
3	2017-12-22	51.05	51.50	50.92	51.32	NaN	71165.0	3641949.0	None	None	None
4	2017-12-21	51.16	51.52	50.90	51.40	NaN	120649.0	6179433.0	None	None	None

3 Heighest stock price

In [23]:
```
df.High.max()
```

Out[23]:
```
53.54
```

3.1 Lowest stock price

In [24]:
```
df.Low.min()
```

Out[24]:
```
33.62
```

4 Largest change in one day

In [25]:
```
diff = df.High - df.Low
diff.max()
```

Out[25]:
```
2.8100000000000023
```

5 Mean of Tradevolume

In [26]:
```
df['Traded Volume'].mean()
```

Out[26]:
```
89124.33725490196
```

6 Median of Trade volume

In [27]:
```
df['Traded Volume'].median()
```

Out[27]:
```
76286.0
```

7 Largest change between any two days

In [28]:
```
df['change'] = df.Close.diff()
df.change.max()
```

Out[28]:
```
2.559999999999995
```

In [ ]:
```

```