Seminar 1

Put the data files in the same folder as your Jupyter Notebook files if there are data files.

We learn coding by imitation. Therefore, we start by copying example codes and run them. Based on the outputs, comments, and the codes, we understand what the codes need and what the codes produce. Then we can modify the codes and apply them to new data for solving new problems.

1. **Install Python and learn about Jupyter Notebook by reading the materials.**
2. **Try basic Python operations: Start a new Jupyter Notebook and copy the following codes one (line/part) by one, followed by press the keys Shift + Enter.**

```
# Python Basics


# This file helps you review basic Python operations. You can learn more details on a free e-book

# https://problemsolvingwithpython.com/


## Lists


an_empty_list = list()

an_empty_list


a_list = [1, 2.3, 'a', True]

a_list


a_list[1]

# prints 2.3


a_list[1] = 2.5

# a_list is now [1, 2.5, 'a', True]

a_list



a_list[1:3]
```

```
# prints [2.5, 'a']


a_list[1:4]


a_list[2:3]


a_list[::-1]
# returns the reverse of the list: [True, 'a', 2.3, 1]


a_list.append(5)
# a_list is now [1, 2.5, 'a', True, 5]
a_list


len(a_list)
# prints 5


del a_list[0]
# a_list is now [2.5, 'a', True, 5]
a_list


a_list += [1, 'b']
# a_list is now [2.5, 'a', True, 5, 1, 'b']
a_list



a, b, c, d, e, f = [2.5, 'a', True, 5, 1, 'b']
# a now is 2.5, b is 'a' and so on
a
```

Defining functions

```
def half(x):
    return x/2.0
```

```
half(10)
```

```
half(18)
```

```
half(29)
```

```
half(12345)
```

Numpy

```
import numpy as np
```

```
a = np.arange(15)
a
```

```
a = a.reshape(3,5)
a
```

```
a = np.arange(15).reshape(3, 5)
a
```

```
a.shape
```

```
a.ndim
```

```
a.size

c=np.zeros((3, 4))
c

d=np.ones((2, 3, 4))
d

a.min()

a.sum()

a.max()

a[0]

a[1]

a[2]
```

3. **Get company stock data: Start a new Jupyter Notebook and copy the following codes one (line/part) by one, followed by press the keys Shift + Enter.**

```
# Get company stock data

# This cell is only needed to be run once.
! pip install yahooquery

from yahooquery import Ticker

aapl = Ticker('aapl')
```

```
aapl.history()


# If no arguments are provided, as above, default values will be supplied for both period and interval, which are ytd and 1d, respectively. Additional arguments you can provide to the method are start and end. Start and end dates can be either strings with a date format of yyyy-mm-dd or as a datetime.datetime object.


aapl.history(period='max')


aapl.history(start='2019-05-01')  # Default end date is now


# Period options = 1d, 5d, 1mo, 3mo, 6mo, 1y, 2y, 5y, 10y, ytd, max

# Interval options = 1m, 2m, 5m, 15m, 30m, 60m, 90m, 1h, 1d, 5d, 1wk, 1mo, 3mo


df=aapl.history(start='2009-05-01', end='2018-12-31')

df


# Save data in memory to disk

df.to_csv('Data_Stock_Date.csv')

df.to_csv('Data_Stock_No_Date.csv', index=False)


# read data from disk to memory

import pandas as pd

df = pd.read_csv('Data_Stock_Date.csv')

df



df = pd.read_csv('Data_Stock_No_Date.csv')

df
```

```
aapl.esg_scores


aapl.key_stats


aapl.summary_profile


aapl.institution_ownership


aapl.fund_ownership


# The following methods take a frequency argument.  If nothing is provided, annual data will be returned.  To return quarterly data, pass "q" as an argument.


aapl.balance_sheet()  # Defaults to Annual


aapl.balance_sheet(frequency="q")


aapl.cash_flow()


aapl.income_statement()
```