```python
# http://pandas-datareader.readthedocs.io/en/latest/remote_data.html
# http://blog.csdn.net/xtfge0915/article/details/52938740
import pandas_datareader.data as web
import datetime
import pandas as pd
# ##############################################################################
# #
# #    write a file
# #
# ##############################################################################
f = open("sample.txt","a+")
contents = "hello, word"
f.write(contents)
f.close()

# ############################
try:
    f = open("sample.txt","w")
    contents = "Hello, world"
    f.write(contents)
except IOError as error:
    print(str(error))
finally:
    f.close()
# ##############################
with open("sample.txt","a+") as out:
    contents = "hello, world"
    out.write(contents)
# ################################

#
start = datetime.datetime(2016, 1, 1)
end = datetime.datetime(2017, 3, 26)
#
# Get AAPL stock prices and show it

AAPL = web.DataReader("AAPL","yahoo",start, end)
print(AAPL.head(5))
print(type(AAPL))
print(AAPL.tail(10))
print(AAPL["Adj Close"].head())
print(AAPL["Adj Close"].head())
print(AAPL.index)

try:
    AAPL.to_csv(r"C:\Users\kooli\Desktop\out.cvs")
except IOError as error:
    print(str(error))
    exit()
```

```
50
51
52 try:
53     data = pd.read_csv(r"C:\Users\kooli\Desktop\out.cvs")
54     print(type(data))
55     print(data.tail(3))
56 except IOError as error:
57     print(str(error))
58
59 AAPL = data
60
61 import matplotlib.pyplot as plt
62 import numpy as np
63 # # ************************draw a simple graph
    ****************************************
64
65 np.random.seed(1000)
66 y = np.random.standard_normal(20)
67 x = range(len(y))
68 plt.plot(x,y)
69 plt.show()
70
71
72 # # ******************************draw stock  in one graph
    *******************************
73 #
74 plt.plot(AAPL.index,AAPL["Adj Close"],marker="o",linestyle = "dashed")
75 plt.show()
76 import pylab
77 pylab.rcParams['figure.figsize'] = (10, 5)
78 AAPL["Adj Close"].plot(grid = True)
79 plt.show()
80 # # ************************************************************************************************
   **
81 # #
82 # # # ******************************draw stock in two graphs
    *********************************
83 fig = plt.figure()
84 ax_price = fig.add_subplot(1,2,1)
85 ax_volume = fig.add_subplot(1,2,2)
86
87 ax_price.plot(AAPL.index, AAPL["Adj Close"])
88 ax_price.set_xticklabels(AAPL.index, rotation = 30, fontsize="small")
89 ax_price.legend(loc = "best")
90
91 ax_volume.plot(AAPL.index,AAPL['Volume'])
92 ax_volume.set_xticklabels([str(x)[0:11] for x in AAPL.index], rotation = -30,
   fontsize="small")
93 ax_volume.set_title("AAPL Volume Trends")
```

```
94 ax_volume.legend(loc = "best")
95 plt.show()
96 # # ***********************************************************************************************
   *******
97 #
98 # # ******************************draw stocks
    ************************************************
99
100 all_data = {}
101
102 for ticker in ['AAPL','IBM','GOOG']:
103     all_data[ticker] = web.DataReader(ticker, 'yahoo', start, end)
104
105 print(all_data)
106
107 import pickle
108
109 f = open(r"stocks.txt","wb")
110
111 try:
112     pickle.dump(all_data,f)
113 except IOError as error:
114     print(str(error))
115 finally:
116     f.close()
117
118
119 f = open(r"stocks.txt", "rb")
120 try:
121     all_data = pickle.load(f)
122 except IOError as error:
123     print(str(error))
124
125
126
127 # print(all_data)
128
129 from pandas import DataFrame
130
131 price = DataFrame({tic: data['Adj Close'] for tic, data in all_data.items()})
132 volume = DataFrame({tic: data['Volume'] for tic, data in all_data.items()})
133
134 # print("price = \n", price.tail(5))
135 # print("price AAPL= \n", price['AAPL'])
136 # print("price AAPL= \n", price.AAPL)
137 # print("volume = \n", volume.tail(5))
138
139
140 import matplotlib.pyplot
```

```
141
142 fig = plt.figure()
143 ax = fig.add_subplot(1,1,1)
144 ax.plot(price.index,price['AAPL'], linestyle= "--", label="apple")
145 ax.plot(price['IBM'].index,price['IBM'].values, linestyle= '-.' ,label="IBM")
146 # ax.plot(price['GOOG'].index,price['GOOG'].values)
147 ax.legend(loc = "best")
148 plt.show()
149
150 print(type(price))
151 # price.plot()
152 price["IBM"].plot()
153 plt.show()
154
155 # # ***********************************************************************************
    *******
156 #
157 returns = price.pct_change()
158 print(returns.tail(10))
159
160 # # The corr method of Series computes the correlation of the overlapping, non-NA,
161 # # aligned-by-index values in two Series. Relatedly, cov computes the covariance:
162 cov = returns.AAPL.corr(returns.IBM)
163 print(cov)
164
165 # # DataFrame's corr and cov methods, on the other hand,
166 # # return a full correlation or covariance matrix as a DataFrame, respectively:
167
168 print(returns.corr())
169 cor = returns.corr()
170 print(cor.idxmin())
171
172
173 print(returns.cov())
174
175 # # Using DataFrame's corrwith method, you can compute pairwise correlations
    between a DataFrame's columns or rows with another Series or DataFrame.
176 # # Passing a Series returns a Series with the correlation value computed for each
    column:
177 aSeries = returns.corrwith(returns.GOOG)
178 print("GOOG\n",aSeries)
179 aSeries.order()
180 print("GOOG\n",aSeries.order(ascending=False))
181
182 # #Passing a DataFrame computes the correlations of matching column names.
183 # # Here I compute correlations of percent changes with volume:
184 print("returns with volume\n",returns.corrwith(volume))
```