Pandas DataFrame -- Practice Code

In [1]:
```
import numpy as np
import pandas as pd
from pandas import Series, DataFrame
from numpy.random import randn
```

In [2]:
```
df1 = pd.read_clipboard()
df1
```

Out[2]:

	Date	Open	High	Low	Close	Volume	Adj Close*
0	Oct 9, 2015	110.00	112.28	109.49	112.12	52,533,800	112.12
1	Oct 8, 2015	110.19	110.19	108.21	109.50	61,698,500	109.50
2	Oct 7, 2015	111.74	111.77	109.41	110.78	46,602,600	110.78
3	Oct 6, 2015	110.63	111.74	109.77	111.31	48,196,800	111.31
4	Oct 5, 2015	109.88	111.37	109.07	110.78	51,723,100	110.78
5	Oct 2, 2015	108.01	111.01	107.55	110.38	57,560,400	110.38
6	Oct 1, 2015	109.07	109.62	107.31	109.58	63,748,000	109.58
7	Sep 30, 2015	110.17	111.54	108.73	110.30	66,105,000	110.30
8	Sep 29, 2015	112.83	113.51	107.86	109.06	73,135,900	109.06
9	Sep 28, 2015	113.85	114.57	112.44	112.44	51,723,900	112.44
10	Sep 25, 2015	116.44	116.69	114.02	114.71	55,842,200	114.71
11	Sep 24, 2015	113.25	115.50	112.37	115.00	49,810,600	115.00
12	Sep 23, 2015	113.63	114.72	113.30	114.32	35,645,700	114.32
13	Sep 22, 2015	113.38	114.18	112.52	113.40	49,809,000	113.40
14	Sep 21, 2015	113.67	115.37	113.66	115.21	46,554,300	115.21
15	Sep 18, 2015	112.21	114.30	111.87	113.45	73,419,000	113.45
16	Sep 17, 2015	115.66	116.49	113.72	113.92	63,462,700	113.92
17	Sep 16, 2015	116.25	116.54	115.44	116.41	36,910,000	116.41
18	Sep 15, 2015	115.93	116.53	114.42	116.28	43,004,100	116.28

In [3]:
```
df1.columns
```

Out[3]:
```
Index([u'Date', u'Open', u'High', u'Low', u'Close', u'Volume', u'A
dj Close*'], dtype='object')
```

In [4]:
```
df1.head()
```

Out[4]:

	Date	Open	High	Low	Close	Volume	Adj Close*
0	Oct 9, 2015	110.00	112.28	109.49	112.12	52,533,800	112.12
1	Oct 8, 2015	110.19	110.19	108.21	109.50	61,698,500	109.50
2	Oct 7, 2015	111.74	111.77	109.41	110.78	46,602,600	110.78
3	Oct 6, 2015	110.63	111.74	109.77	111.31	48,196,800	111.31
4	Oct 5, 2015	109.88	111.37	109.07	110.78	51,723,100	110.78

In [5]:
```
df1.tail()
```

Out[5]:

	Date	Open	High	Low	Close	Volume	Adj Close*
14	Sep 21, 2015	113.67	115.37	113.66	115.21	46,554,300	115.21
15	Sep 18, 2015	112.21	114.30	111.87	113.45	73,419,000	113.45
16	Sep 17, 2015	115.66	116.49	113.72	113.92	63,462,700	113.92
17	Sep 16, 2015	116.25	116.54	115.44	116.41	36,910,000	116.41
18	Sep 15, 2015	115.93	116.53	114.42	116.28	43,004,100	116.28

In [6]:
```
df1.Date
```

Out[6]:
```
0      Oct 9, 2015
1      Oct 8, 2015
2      Oct 7, 2015
3      Oct 6, 2015
4      Oct 5, 2015
5      Oct 2, 2015
6      Oct 1, 2015
7     Sep 30, 2015
8     Sep 29, 2015
9     Sep 28, 2015
10    Sep 25, 2015
11    Sep 24, 2015
12    Sep 23, 2015
13    Sep 22, 2015
14    Sep 21, 2015
15    Sep 18, 2015
16    Sep 17, 2015
17    Sep 16, 2015
18    Sep 15, 2015
Name: Date, dtype: object
```

In [7]:
```
df1.Volume
```

Out[7]:
```
0     52,533,800
1     61,698,500
2     46,602,600
3     48,196,800
4     51,723,100
5     57,560,400
6     63,748,000
7     66,105,000
8     73,135,900
9     51,723,900
10    55,842,200
11    49,810,600
12    35,645,700
13    49,809,000
14    46,554,300
15    73,419,000
16    63,462,700
17    36,910,000
18    43,004,100
Name: Volume, dtype: object
```

In [8]:
```
df1.columns
```

Out[8]:
```
Index([u'Date', u'Open', u'High', u'Low', u'Close', u'Volume', u'A
dj Close*'], dtype='object')
```

In [9]:
```
df1[['Date', 'High', 'Low', 'Close']]
```

Out[9]:

	Date	High	Low	Close
0	Oct 9, 2015	112.28	109.49	112.12
1	Oct 8, 2015	110.19	108.21	109.50
2	Oct 7, 2015	111.77	109.41	110.78
3	Oct 6, 2015	111.74	109.77	111.31
4	Oct 5, 2015	111.37	109.07	110.78
5	Oct 2, 2015	111.01	107.55	110.38
6	Oct 1, 2015	109.62	107.31	109.58
7	Sep 30, 2015	111.54	108.73	110.30
8	Sep 29, 2015	113.51	107.86	109.06
9	Sep 28, 2015	114.57	112.44	112.44
10	Sep 25, 2015	116.69	114.02	114.71
11	Sep 24, 2015	115.50	112.37	115.00
12	Sep 23, 2015	114.72	113.30	114.32
13	Sep 22, 2015	114.18	112.52	113.40
14	Sep 21, 2015	115.37	113.66	115.21
15	Sep 18, 2015	114.30	111.87	113.45
16	Sep 17, 2015	116.49	113.72	113.92
17	Sep 16, 2015	116.54	115.44	116.41
18	Sep 15, 2015	116.53	114.42	116.28

In [10]:
```
df1.head()
```

Out[10]:

	Date	Open	High	Low	Close	Volume	Adj Close*
0	Oct 9, 2015	110.00	112.28	109.49	112.12	52,533,800	112.12
1	Oct 8, 2015	110.19	110.19	108.21	109.50	61,698,500	109.50
2	Oct 7, 2015	111.74	111.77	109.41	110.78	46,602,600	110.78
3	Oct 6, 2015	110.63	111.74	109.77	111.31	48,196,800	111.31
4	Oct 5, 2015	109.88	111.37	109.07	110.78	51,723,100	110.78

In [11]:
```
df1['Exchange'] = 'Nasdaq'
df1
```

Out[11]:

	Date	Open	High	Low	Close	Volume	Adj Close*	Exchange
0	Oct 9, 2015	110.00	112.28	109.49	112.12	52,533,800	112.12	Nasdaq
1	Oct 8, 2015	110.19	110.19	108.21	109.50	61,698,500	109.50	Nasdaq
2	Oct 7, 2015	111.74	111.77	109.41	110.78	46,602,600	110.78	Nasdaq
3	Oct 6, 2015	110.63	111.74	109.77	111.31	48,196,800	111.31	Nasdaq
4	Oct 5, 2015	109.88	111.37	109.07	110.78	51,723,100	110.78	Nasdaq
5	Oct 2, 2015	108.01	111.01	107.55	110.38	57,560,400	110.38	Nasdaq
6	Oct 1, 2015	109.07	109.62	107.31	109.58	63,748,000	109.58	Nasdaq
7	Sep 30, 2015	110.17	111.54	108.73	110.30	66,105,000	110.30	Nasdaq
8	Sep 29, 2015	112.83	113.51	107.86	109.06	73,135,900	109.06	Nasdaq
9	Sep 28, 2015	113.85	114.57	112.44	112.44	51,723,900	112.44	Nasdaq
10	Sep 25, 2015	116.44	116.69	114.02	114.71	55,842,200	114.71	Nasdaq
11	Sep 24, 2015	113.25	115.50	112.37	115.00	49,810,600	115.00	Nasdaq
12	Sep 23, 2015	113.63	114.72	113.30	114.32	35,645,700	114.32	Nasdaq
13	Sep 22, 2015	113.38	114.18	112.52	113.40	49,809,000	113.40	Nasdaq
14	Sep 21, 2015	113.67	115.37	113.66	115.21	46,554,300	115.21	Nasdaq
15	Sep 18, 2015	112.21	114.30	111.87	113.45	73,419,000	113.45	Nasdaq
16	Sep 17, 2015	115.66	116.49	113.72	113.92	63,462,700	113.92	Nasdaq
17	Sep 16, 2015	116.25	116.54	115.44	116.41	36,910,000	116.41	Nasdaq
18	Sep 15, 2015	115.93	116.53	114.42	116.28	43,004,100	116.28	Nasdaq

Creating DF from a directory

In [12]:
```
scores = [92, 88, 95, 85, 98]
students = ['Gary', 'Alex', 'Kris', 'Tom', 'Cathy']
```

In [13]:
```
data = { 'Student':students, 'Score':scores }
data
```

Out[13]:
```
{'Score': [92, 88, 95, 85, 98],
 'Student': ['Gary', 'Alex', 'Kris', 'Tom', 'Cathy']}
```

In [14]:
```
df2 = DataFrame( data )
df2
```

Out[14]:

	Score	Student
0	92	Gary
1	88	Alex
2	95	Kris
3	85	Tom
4	98	Cathy

In [15]:
```
ind = "A B C D E".split()
cols = "col1 col2 col3 col4 col5".split()

x = []
for i in range(25):
    x.append( np.random.randint( 1, 100 ) )

x = np.array( x )
x = x.reshape(5,5)
x
```

Out[15]:
```
array([[83, 91, 26, 75,  7],
       [91, 37, 56, 99, 31],
       [ 8, 95, 61,  4, 79],
       [78, 62, 53, 49, 94],
       [47, 22, 35,  7, 51]])
```

In [17]:
```
df3 = DataFrame( x, index=ind, columns=cols )
df3
```

Out[17]:

	col1	col2	col3	col4	col5
A	83	91	26	75	7
B	91	37	56	99	31
C	8	95	61	4	79
D	78	62	53	49	94
E	47	22	35	7	51

In [18]:
```
new_ind = "A B C D E F G".split()
```

In [19]:
```
df4 = df3.reindex( new_ind, fill_value = 0 )
df4
```

Out[19]:

	col1	col2	col3	col4	col5
A	83	91	26	75	7
B	91	37	56	99	31
C	8	95	61	4	79
D	78	62	53	49	94
E	47	22	35	7	51
F	0	0	0	0	0
G	0	0	0	0	0

In [20]:
```
new_cols = "col1 col2 col3 col4 col5 col6".split()
```

In [21]:
```
df5 = df4.reindex( columns=new_cols,  fill_value = 0 )
df5
```

Out[21]:

	col1	col2	col3	col4	col5	col6
A	83	91	26	75	7	0
B	91	37	56	99	31	0
C	8	95	61	4	79	0
D	78	62	53	49	94	0
E	47	22	35	7	51	0
F	0	0	0	0	0	0
G	0	0	0	0	0	0

Data Selections

In [22]:
```
df5
```

Out[22]:

	col1	col2	col3	col4	col5	col6
A	83	91	26	75	7	0
B	91	37	56	99	31	0
C	8	95	61	4	79	0
D	78	62	53	49	94	0
E	47	22	35	7	51	0
F	0	0	0	0	0	0
G	0	0	0	0	0	0

In [24]:
```
df5['col1']
```

Out[24]:
```
A    83
B    91
C     8
D    78
E    47
F     0
G     0
Name: col1, dtype: int64
```

In [25]:
```
df5[ [ 'col3', 'col5'] ]
```

Out[25]:

	col3	col5
A	26	7
B	56	31
C	61	79
D	53	94
E	35	51
F	0	0
G	0	0

In [26]:
```
df5
```

Out[26]:

	col1	col2	col3	col4	col5	col6
A	83	91	26	75	7	0
B	91	37	56	99	31	0
C	8	95	61	4	79	0
D	78	62	53	49	94	0
E	47	22	35	7	51	0
F	0	0	0	0	0	0
G	0	0	0	0	0	0

In [28]:
```
df5[ df5['col4'] < 20 ]
```

Out[28]:

	col1	col2	col3	col4	col5	col6
C	8	95	61	4	79	0
E	47	22	35	7	51	0
F	0	0	0	0	0	0
G	0	0	0	0	0	0

In [30]:
```
df5
```

Out[30]:

	col1	col2	col3	col4	col5	col6
A	83	91	26	75	7	0
B	91	37	56	99	31	0
C	8	95	61	4	79	0
D	78	62	53	49	94	0
E	47	22	35	7	51	0
F	0	0	0	0	0	0
G	0	0	0	0	0	0

Boolean Data Frame

In [31]:
```
df5 < 50
```

Out[31]:

	col1	col2	col3	col4	col5	col6
A	False	False	True	False	True	True
B	False	True	False	False	True	True
C	True	False	False	True	False	True
D	False	False	False	True	False	True
E	True	True	True	True	False	True
F	True	True	True	True	True	True
G	True	True	True	True	True	True

In [33]:
```
df5
```

Out[33]:

	col1	col2	col3	col4	col5	col6
A	83	91	26	75	7	0
B	91	37	56	99	31	0
C	8	95	61	4	79	0
D	78	62	53	49	94	0
E	47	22	35	7	51	0
F	0	0	0	0	0	0
G	0	0	0	0	0	0

In [34]:
```
df5.ix['E']
```

Out[34]:
```
col1    47
col2    22
col3    35
col4     7
col5    51
col6     0
Name: E, dtype: int64
```

Dropping Rows & Columns

In [40]:
```
df5
```

Out[40]:

	col1	col2	col3	col4	col5	col6
A	83	91	26	75	7	0
B	91	37	56	99	31	0
C	8	95	61	4	79	0
D	78	62	53	49	94	0
E	47	22	35	7	51	0
F	0	0	0	0	0	0
G	0	0	0	0	0	0

In [41]:
```
df5.drop('F')
```

Out[41]:

	col1	col2	col3	col4	col5	col6
A	83	91	26	75	7	0
B	91	37	56	99	31	0
C	8	95	61	4	79	0
D	78	62	53	49	94	0
E	47	22	35	7	51	0
G	0	0	0	0	0	0

In [42]:
```
df5
```

Out[42]:

	col1	col2	col3	col4	col5	col6
A	83	91	26	75	7	0
B	91	37	56	99	31	0
C	8	95	61	4	79	0
D	78	62	53	49	94	0
E	47	22	35	7	51	0
F	0	0	0	0	0	0
G	0	0	0	0	0	0

Data Alignment

In [43]:
```
df5.drop('col5', axis=1 )
```

Out[43]:

	col1	col2	col3	col4	col6
A	83	91	26	75	0
B	91	37	56	99	0
C	8	95	61	4	0
D	78	62	53	49	0
E	47	22	35	7	0
F	0	0	0	0	0
G	0	0	0	0	0

In [44]:
```
df6 = DataFrame(np.arange(4).reshape(2,2), columns=list('AB'), inde
x=list('xy') )
df6
```

Out[44]:

	A	B
x	0	1
y	2	3

In [45]:
```
df7 = DataFrame(np.arange(9).reshape(3,3), columns=list('ABC'), ind
ex=list('xyz') )
df7
```

Out[45]:

	A	B	C
x	0	1	2
y	3	4	5
z	6	7	8

In [46]:
```
df6 + df7
```

Out[46]:

	A	B	C
x	0	2	NaN
y	5	7	NaN
z	NaN	NaN	NaN

In [47]:
```
df6.add( df7, fill_value=0)
```

Out[47]:

	A	B	C
x	0	2	2
y	5	7	5
z	6	7	8

Operations on DFs

In [48]:
```
df5
```

Out[48]:

	col1	col2	col3	col4	col5	col6
A	83	91	26	75	7	0
B	91	37	56	99	31	0
C	8	95	61	4	79	0
D	78	62	53	49	94	0
E	47	22	35	7	51	0
F	0	0	0	0	0	0
G	0	0	0	0	0	0

In [50]:
```
df8 = df5.drop( [ 'F', 'G'])
df9 = df8.drop( ['col4', 'col6'], axis=1 )
df9
```

Out[50]:

	col1	col2	col3	col5
A	83	91	26	7
B	91	37	56	31
C	8	95	61	79
D	78	62	53	94
E	47	22	35	51

In [51]:
```
df9.sum()
```

Out[51]:
```
col1    307
col2    307
col3    231
col5    262
dtype: int64
```

In [52]:
```
df9.sum( axis = 1 )
```

Out[52]:
```
A    207
B    215
C    243
D    287
E    155
dtype: int64
```

In [53]:
```
df9.max()
```

Out[53]:
```
col1    91
col2    95
col3    61
col5    94
dtype: int64
```

In [54]:
```
df9
```

Out[54]:

	col1	col2	col3	col5
A	83	91	26	7
B	91	37	56	31
C	8	95	61	79
D	78	62	53	94
E	47	22	35	51

In [56]:
```
df9.idxmax()
```

Out[56]:
```
col1    B
col2    C
col3    C
col5    D
dtype: object
```

In [57]:
```
df9
```

Out[57]:

	col1	col2	col3	col5
A	83	91	26	7
B	91	37	56	31
C	8	95	61	79
D	78	62	53	94
E	47	22	35	51

In [58]:
```
df9.cumsum()
```

Out[58]:

	col1	col2	col3	col5
A	83	91	26	7
B	174	128	82	38
C	182	223	143	117
D	260	285	196	211
E	307	307	231	262

In [59]:
```
df9.describe()
```

Out[59]:

	col1	col2	col3	col5
count	5.000000	5.000000	5.000000	5.000000
mean	61.400000	61.400000	46.200000	52.400000
std	34.195029	32.222663	14.956604	35.210794
min	8.000000	22.000000	26.000000	7.000000
25%	47.000000	37.000000	35.000000	31.000000
50%	78.000000	62.000000	53.000000	51.000000
75%	83.000000	91.000000	56.000000	79.000000
max	91.000000	95.000000	61.000000	94.000000

In [60]:
```
nd = np.nan
A = [1, 2, 3]
B = [4, nd, 6]
C = [nd, 8, nd]
D = [nd, nd, nd]

df10 = DataFrame( [A, B, C, D])
df10
```

Out[60]:

	0	1	2
0	1	2	3
1	4	NaN	6
2	NaN	8	NaN
3	NaN	NaN	NaN

In [61]:
```
df10.dropna()
```

Out[61]:

	0	1	2
0	1	2	3

In [63]:
```
df10.dropna( how='all')
```

Out[63]:

	0	1	2
0	1	2	3
1	4	NaN	6
2	NaN	8	NaN

In [64]:
```
df10
```

Out[64]:

	0	1	2
0	1	2	3
1	4	NaN	6
2	NaN	8	NaN
3	NaN	NaN	NaN

In [65]:
```
df10.dropna(thresh=2)
```

Out[65]:

	0	1	2
0	1	2	3
1	4	NaN	6

In [66]:
```
df10.dropna(thresh=3)
```

Out[66]:

	0	1	2
0	1	2	3

In [67]:
```
df10
```

Out[67]:

	0	1	2
0	1	2	3
1	4	NaN	6
2	NaN	8	NaN
3	NaN	NaN	NaN

In [68]:
```
df10.fillna( 100 )
```

Out[68]:

	0	1	2
0	1	2	3
1	4	100	6
2	100	8	100
3	100	100	100

In [69]:
```
df10
```

Out[69]:

	0	1	2
0	1	2	3
1	4	NaN	6
2	NaN	8	NaN
3	NaN	NaN	NaN

In [70]:
```
df10.fillna( 0, inplace=True )
df10
```

Out[70]:

	0	1	2
0	1	2	3
1	4	0	6
2	0	8	0
3	0	0	0

Multi-level Indexing on DFs

In [72]:
```
ind1 = "A A A B B B".split()
ind2  = "a b c a b c".split()
cols1 = "C1 C2 C2 C3 C3".split()
cols2 = "col1 col2 col3 col4 col5".split()

x = []
for i in range(30):
    x.append( np.random.randint(1, 100) )

x = np.array( x)       # Converting a List to an Array
x = x.reshape(6, 5)    # Reshapping Array to a matrix
x
```

Out[72]:
```
array([[52, 65, 78, 82, 65],
       [77, 94, 20, 36, 13],
       [97,  4, 23, 76,  4],
       [80, 52, 89, 76, 89],
       [81, 41, 38, 42,  1],
       [33, 46, 37, 32, 81]])
```

In [73]:
```
df11 = DataFrame( x, index=[ind1, ind2], columns=[cols1, cols2 ])
df11
```

Out[73]:

		C1	C2		C3	
		col1	col2	col3	col4	col5
A	a	52	65	78	82	65
	b	77	94	20	36	13
	c	97	4	23	76	4
B	a	80	52	89	76	89
	b	81	41	38	42	1
	c	33	46	37	32	81

In [74]:
```
df11['C2']
```

Out[74]:

		col2	col3
A	a	65	78
	b	94	20
	c	4	23
B	a	52	89
	b	41	38
	c	46	37

In [75]:
```
df11[['C1', 'C3']]
```

Out[75]:

		C1	C3	
		col1	col4	col5
A	a	52	82	65
	b	77	36	13
	c	97	76	4
B	a	80	76	89
	b	81	42	1
	c	33	32	81

Getting Stock Prices from Yahoo and Plotting

In [76]:
```
import pandas.io.data as pdweb
import datetime as dt
```

In [77]:
```
prices = pdweb.get_data_yahoo(['AAPL'], start=dt.datetime(2015, 1, 1),
                              end=dt.datetime(2015,9,30))['Adj Close']
prices.head()
```

Out[77]:

	AAPL
Date	
2015-01-02	107.958556
2015-01-05	104.917190
2015-01-06	104.927067
2015-01-07	106.398374
2015-01-08	110.486441

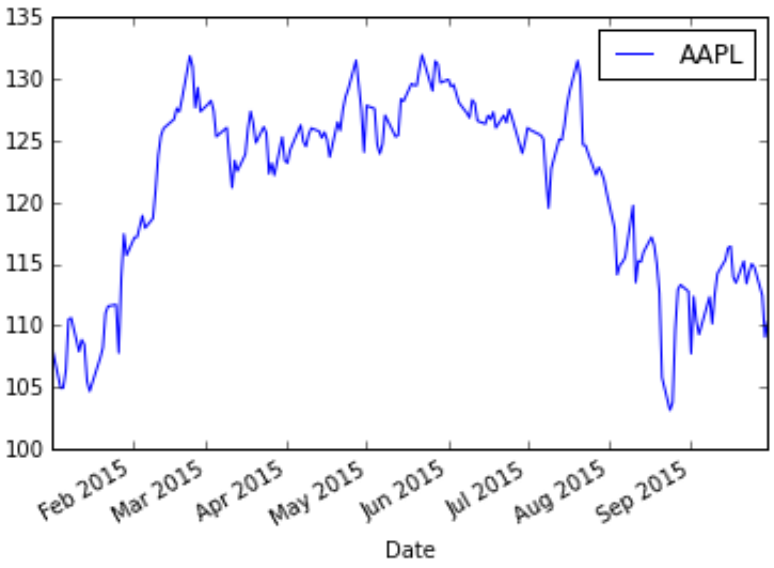
In [78]:
```
prices.tail()
```

Out[78]:

	AAPL
Date	
2015-09-24	115.000000
2015-09-25	114.709999
2015-09-28	112.440002
2015-09-29	109.059998
2015-09-30	110.300003

In [79]:
```
%matplotlib inline
prices.plot()
```

Out[79]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x10c4facd0>
```



Plotting with multiple scurities -- Apple and Netflix Prices

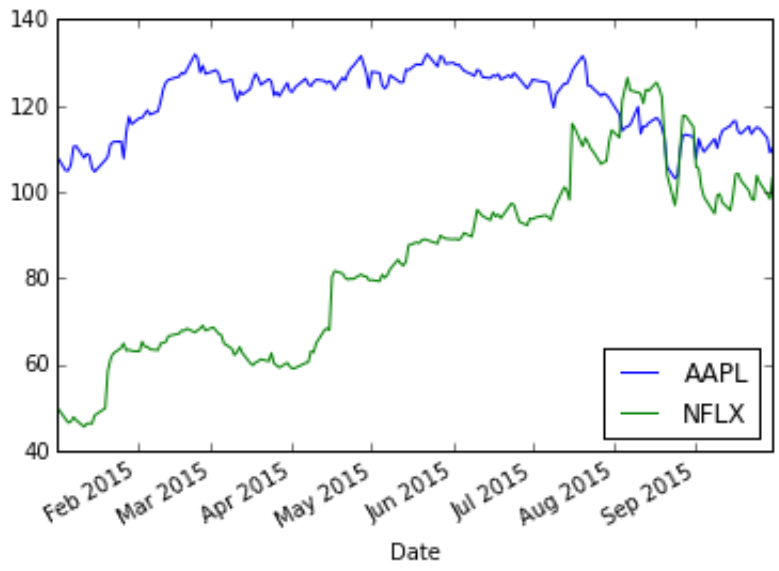
In [80]:
```
prices2 = pdweb.get_data_yahoo(['AAPL','NFLX'], start=dt.datetime(2
015, 1, 1),
                               end=dt.datetime(2015,9,30))['Adj Clos
e']
prices2.head()
```

Out[80]:

	AAPL	NFLX
Date		
2015-01-02	107.958556	49.848572
2015-01-05	104.917190	47.311428
2015-01-06	104.927067	46.501427
2015-01-07	106.398374	46.742859
2015-01-08	110.486441	47.779999

In [81]:
```
%matplotlib inline
prices2.plot()
```

Out[81]: <matplotlib.axes._subplots.AxesSubplot at 0x10c53f2d0>



In [ ]: