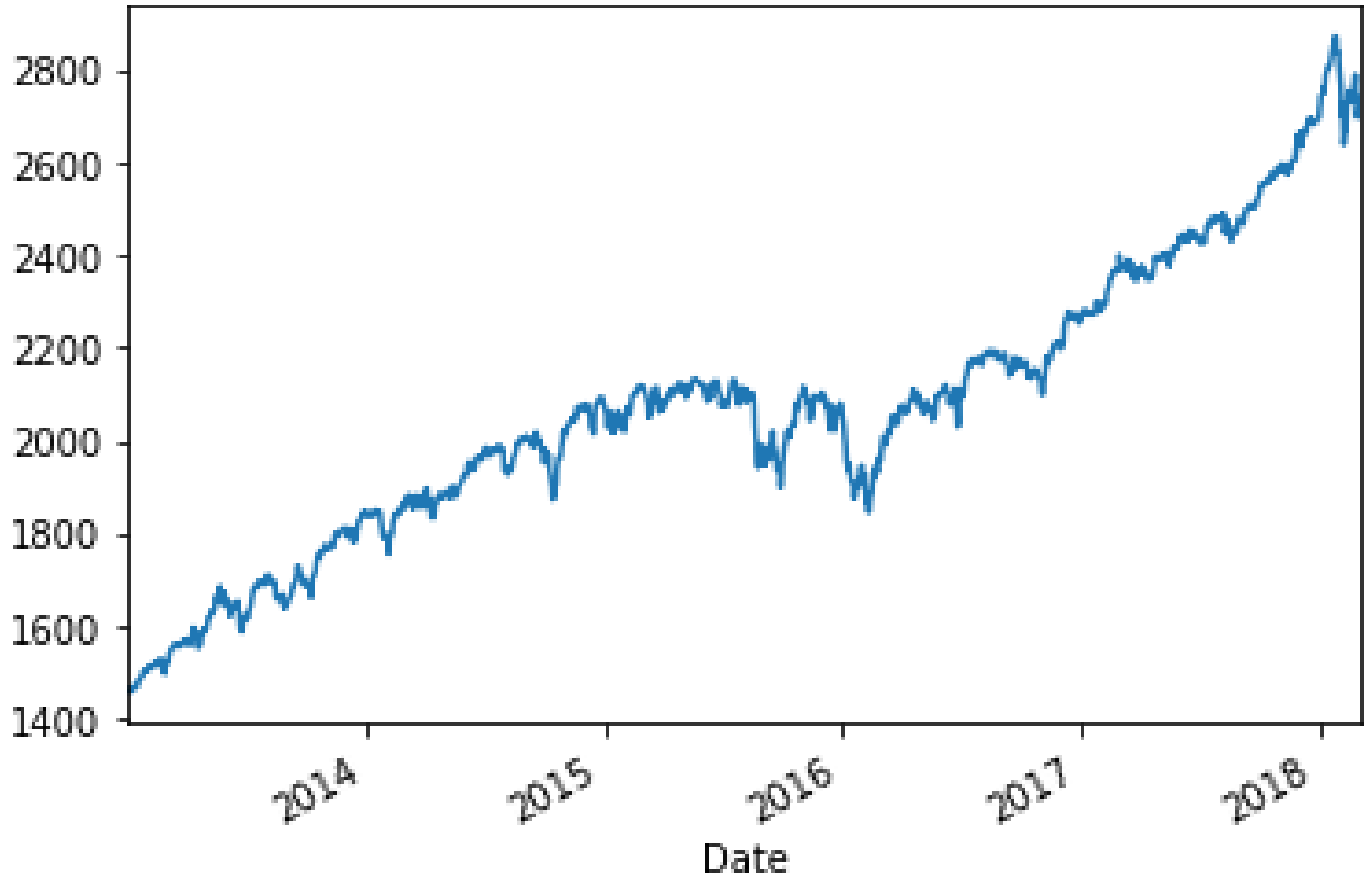
Representing time with datetimes

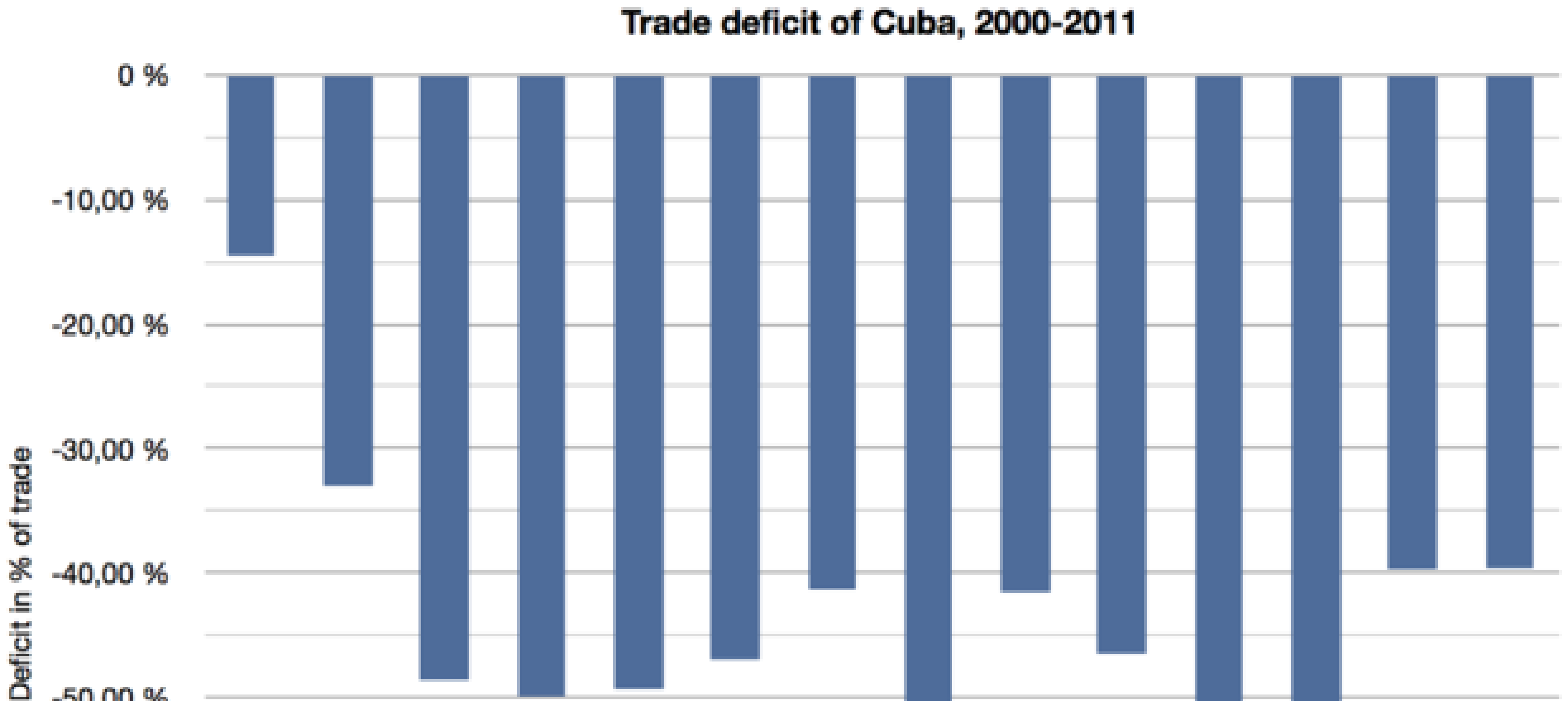
INTERMEDIATE PYTHON FOR FINANCE

Kennedy Behrman
Data Engineer, Author, Founder

Datetimes


2800
2600
2400
2200
2000
1800
1600
1400
2014
2015
2016
2017
2018
Date

Datetimes



Trade deficit of Cuba, 2000-2011
0 %
-10,00 %
-20,00 %
-30,00 %
-40,00 %
-50,00 %
Deficit in % of trade

Datetimes

```
from datetime import datetime
```

These objects have attributes representing the year, month, date, hour, second, microsecond, and timezone.

```
black_monday = datetime(1987, 10, 19)
print(black_monday)
```

In order to create a datetime object, you must at least provide the year, month, and date.

```
datetime.datetime(1987, 10, 19, 0, 0)
```

Datetime now

```
datetime.now()
```

```
datetime.datetime(2019, 11, 6, 3, 48, 30, 886713)
```

Datetime from string

```
black_monday_str = "Monday, October 19, 1987. 9:30 am"
format_str = "%A, %B %d, %Y. %I:%M %p"
datetime.datetime.strptime(black_monday_str, format_str)
```

string-parse-time

```
datetime.datetime(1987, 10, 19, 9, 30)
```

Datetime from string

Year

- `%y` Without century (01, 02, ..., 98, 99)
- `%Y` With century (0001, 0002, ..., 1998, 1999, ..., 9999)

Month

- `%b` Abbreviated names (Jan, Feb, ..., Nov, Dec)
- `%B` Full names (January, February, ... November, December)
- `%m` As numbers (01, 02, ..., 11, 12)

Day of Month

- `%d` (01, 02, ..., 30, 31)

Datetime from string

Weekday

- `%a` Abbreviated name (Sun, ... Sat)
- `%A` Full name (Sunday, ... Saturday)
- `%w` Number (0, ..., 6)

Hour

- `%H` 24 hour (00, 01, ... 23)
- `%I` 12 hour (01, 02, ... 12)
- `%M` (01, 02, ..., 59)

Datetime from string

Seconds

- `%S` (00, 01, ... 59)

Micro-seconds

- `%f` (000000, 000001, ... 999999)

AM/PM

- `%p` (AM, PM)

Datetime from string

`%m` Months

`%M` Minutes

Datetime from string

```
"1837-05-10"
```

`%Y`

`%m`

`%d`

```
"%Y-%m-%d"
```

Datetime from string

```
"Friday, 17 May 01"
```

`%A`

`%d`

`%B`

`%y`

```
"%A, %d %B %y"
```

String from datetime

```
dt.strftime(format_string)
```

string-from-time

String from datetime

```
great_depression_crash = datetime.datetime(1929, 10, 29)
great_depression_crash
```

```
datetime.datetime(1929, 10, 29, 0, 0)
```

```
great_depression_crash.strftime("%a, %b %d, %Y")
```

```
'Tue, Oct 29, 1929'
```

Let's practice!

INTERMEDIATE PYTHON FOR FINANCE

Working with datetimes

INTERMEDIATE PYTHON FOR FINANCE

Kennedy Behrman
Data Engineer, Author, Founder

Datetime attributes

```python
now.year
now.month
now.day
```

```
2019
11
13
```

```python
now.hour
now.minute
now.second
```

```
22
34
56
```

Comparing datetimes

equals `==`

less than `<`

more than `>`

Comparing datetimes

```
from datetime import datetime
asian_crisis = datetime(1997, 7, 2)
world_mini_crash = datetime(1997, 10, 27)
```

```
asian_crisis > world_mini_crash
```

```
False
```

```
asian_crisis < world_mini_crash
```

```
True
```

Comparing datetimes

```
asian_crisis = datetime(1997, 7, 2)
world_mini_crash = datetime(1997, 10, 27)
```

```
text = "10/27/1997"
format_str =  "%m/%d/%Y"
sell_date = datetime.strptime(text, format_str)
```

```
sell_date == world_mini_crash
```

```
True
```

Difference between datetimes

- Compare with `<` , `>` , or `==` .
- Subtraction returns a `timedelta` object.
- `timedelta` attributes: weeks, days, minutes, seconds, microseconds

Difference between datetimes

```
delta = world_mini_crash - asian_crisis
```

```
type(delta)
```

```
datetime.timedelta
```

```
delta.days
```

```
117
```

Creating relative datetimes

```
dt
```

```
datetime.datetime(2019, 1, 14, 0, 0)
```

```
datetime(dt.year, dt.month, dt.day - 7)
```

```
datetime.datetime(2019, 1, 7, 0, 0)
```

```
datetime(dt.year, dt.month, dt.day - 15)
```

```
ValueError                                Traceback (most recent call last)
<ipython-input-28-804001f45cdb> in <module>()
-> 1 datetime(dt.year, dt.month, dt.day - 15)
ValueError: day is out of range for month
```

Creating relative datetimes

```
delta = world_mini_crash - asian_crisis
type(delta)
```

```
datetime.timedelta
```

Creating relative datetimes

```python
from datetime import timedelta
```

```python
offset = timedelta(weeks = 1)
offset
```

```
datetime.timedelta(7)
```

```python
dt - offset
```

```
datetime.datetime(2019, 1, 7, 0, 0)
```

Creating relative datetimes

```
offset = timedelta(days=16)
dt - offset
```

```
datetime.datetime(2018, 12, 29, 0, 0)
```

```
cur_week = last_week + timedelta(weeks=1)
# Do some work with date
# set last week variable to cur week and repeat
last_week = cur_week
```

```
source_dt = event_dt - timedelta(weeks=4)
# Use source datetime to look up market factors
```

Let's practice!

INTERMEDIATE PYTHON FOR FINANCE

Dictionaries

INTERMEDIATE PYTHON FOR FINANCE

Kennedy Behrman
Data Engineer, Author, Founder

Lookup by index

```
my_list = ['a','b','c','d']
```

```
  0    1    2    3
['a','b','c','d']
```

```
my_list[0]
```

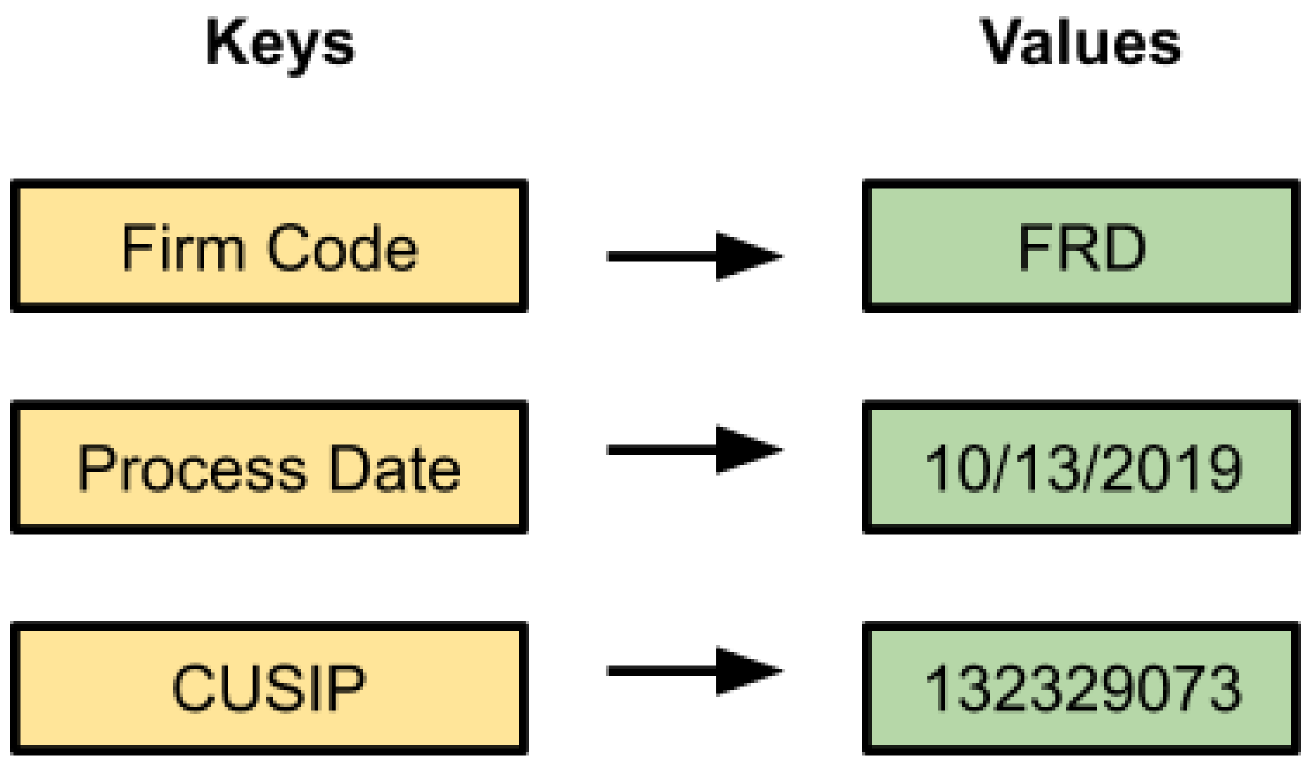
```
'a'
```

```
my_list.index('c')
```

```
2
```

Lookup by key

Dictionaries



Representation

```
{ 'key-1':'value-1', 'key-2':'value-2', 'key-3':'value-3'}
```

Creating dictionaries

```python
my_dict = {}
my_dict
```

```
{}
```

```python
my_dict = dict()
my_dict
```

```
{}
```

Creating dictionaries

```
ticker_symbols = {'AAPL':'Apple', 'F':'Ford', 'LUV':'Southwest'}
print(ticker_symbols)
```

```
{'AAPL':'Apple', 'F':'Ford', 'LUV':'Southwest'}
```

```
ticker_symbols = dict([['APPL','Apple'],['F','Ford'],['LUV','Southwest']])
print(ticker_symbols)
```

```
{'AAPL':'Apple', 'F':'Ford', 'LUV':'Southwest'}
```

Adding to dictionaries

```
ticker_symbols['XON'] = 'Exxon'
ticker_symbols
```

```
{'APPL': 'Apple', 'F': 'Ford', 'LUV': 'Southwest', 'XON': 'Exxon'}
```

```
ticker_symbols['XON'] = 'Exxon OLD'
ticker_symbols
```

```
{'APPL': 'Apple','F': 'Ford','LUV': 'Southwest','XON': 'Exxon OLD'}
```

Accessing values

```python
ticker_symbols['F']
```

```
'Ford'
```

Accessing values

```
ticker_symbols['XOM']
```

```
KeyError                                  Traceback (most recent call last)
<ipython-input-6-782fbf617bf7> in <module>()
-> 1 ticker_symbols['XOM']

   KeyError: 'XOM'
```

Accessing values

```
company = ticker_symbols.get('LUV')
print(company)
```

```
'Southwest'
```

```
company = ticker_symbols.get('XOM')
print(company)
```

```
None
```

```
company = ticker_symbols.get('XOM', 'MISSING')
print(company)
```

```
'MISSING'
```

Deleting from dictionaries

```
ticker_symbols
```

```
{'APPL': 'Apple', 'F': 'Ford', 'LUV': 'Southwest', 'XON': 'Exxon OLD'}
```

```
del(ticker_symbols['XON'])
```

```
ticker_symbols
```

```
{'APPL': 'Apple', 'F': 'Ford', 'LUV': 'Southwest'}
```

Let's practice!

INTERMEDIATE PYTHON FOR FINANCE