Recall:

Objects

Methods

Common methods of:

- `list`
- `str`

Tuples

You've heard these before?

- quadruple
- quintuple
- sextuple
- septuple

Mathematically:

$$n\text{-tuple}$$

Python tuple

An ordered collection

So... what's different about this vs a list?

An *immutable* ordered collection

```
l = [1, 2, 3]
t = (1, 2, 3)

l[1] = 42
t[1] = 42
```

Where do we use tuples?

Storing simple sequences

Returning multiple things from functions

- let's write a `divide(num, denom)` function!
- other functions you might bump into

More tuple syntax

Single-element tuples:

```
x = (1,)
```

***Destructuring* tuples:**

```
p = get_a_cartesian_point()
x, y = p

# or just:
x, y = get_a_cartesian_point()
```

Helpful for explicitly stating expected number of elements!

matplotlib example

```
>>> help(plt.subplots)
Help on function subplots in module matplotlib.pyplot:

subplots(nrows=1, ncols=1, sharex=False, sharey=False, squeeze=True, s
None, gridspec_kw=None, **fig_kw)
    Create a figure and a set of subplots

    This utility wrapper makes it convenient to create common layouts
    subplots, including the enclosing figure object, in a single call.
    [...]

    Returns
    -------
    fig : :class:`matplotlib.figure.Figure` object

    ax : Axes object or array of Axes objects.
    [...]
```

Returns _____ values: ____**fig**____**ax**____

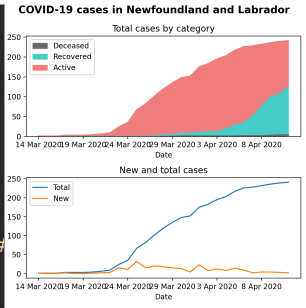
Using subplots

```
fig, ax = plt.subplots(2, # ...
fig.suptitle('COVID-19 cases in # ...

ax[0].set_title('New and total cases')
ax[0].plot(total)
ax[0].plot(new)
ax[0].legend(['Total', 'New'])

ax[1].set_title('Total cases by category')
ax[1].stackplot(range(len(total)), deaths, #
ax[1].legend(['Deceased', 'Recovered', # ..

plt.show()
```



Data: http://bit.ly/covid19-nl

Code: https://gist.github.com/trombonehero/b7b2ec2667dab2bf3bb09399984a8046

Using tuples

Just like lists:

- iteration
- `count()` method
- `index()` method
- ... but that's all!

Argument tuples

Slightly more advanced usage

```
def foo(*args):
    for a in args:
        print(a)

foo(1, 2, 3)
```

Or even:

```
def multiply(x, y):
    return x * y

t = (1, 2)
```

Summary

Tuples

- simple ordered *immutable* collections
- syntax
- usage