Python Introduction: Functions

Functions help extract out common code blocks.

Let's define a function `print_greeting()`.

In [1]:

```
def print_greeting():
    print("Hi there, how are you?")
    print("Long time no see.")
```

And call it:

In [2]:

```
print_greeting()
```

```
Hi there, how are you?
Long time no see.
```

That's a bit impersonal.

In [3]:

```
def print_greeting(name):
    print("Hi there, {0}, how are you?".format(name))
    print("Long time no see.")
```

In [4]:

```
print_greeting("Andreas")
```

```
Hi there, Andreas, how are you?
Long time no see.
```

But we might not know their name.

(And we just changed the interface of `print_greeting`!)

In [6]:

```
def print_greeting(name="my friend"):
    print("Hi there, {0}, how are you?".format(name))
    print("Long time no see.")
```

In [7]:

```
print_greeting("Andreas")
print_greeting()
```

```
Hi there, Andreas, how are you?
Long time no see.
Hi there, my friend, how are you?
Long time no see.
```

Function parameters work like variables.

So what does this do?

In [8]:

```
def my_func(my_list):
    my_list.append(5)

l = [1,2,3]
my_func(l)
print(l)
```

```
[1, 2, 3, 5]
```

Can be very surprising!

Define a better function `my_func_2`:

In [9]:

```
def my_func_2(my_list):
    return my_list + [5]
```

In [10]:

```
l = [1,2,3]
l2 = my_func_2(l)
print(l)
print(l2)
```

```
[1, 2, 3]
[1, 2, 3, 5]
```