Functions

Reference:

- https://docs.python.org/3/tutorial/controlflow.html#defining-functions
- https://docs.python.org/3/tutorial/controlflow.html#more-on-defining-functions
- https://docs.python.org/3/tutorial/controlflow.html#default-argument-values
- https://docs.python.org/3/tutorial/controlflow.html#keyword-arguments

Use a function to define your own custom, re-usable operation. Like in other languages, Python functions must first be defined before they can be invoked (or called).

Define a function:

```
def do_stuff(): # NOTE: the trailing parentheses are required
    print("DOING STUFF HERE!")
```

Invoke the function:

```
do_stuff() # NOTE: the trailing parentheses are important. If they are
omitted, the function will be accessed but not be invoked
```

If you try to invoke a function before or without defining it, you will see an error like `NameError: name 'do_stuff' is not defined`.

Parameters (Input Values)

Some functions accept parameters which can be passed to the function during its invocation. A function's parameters must be configured during the function's definition.

Single Parameter

Define a function with a parameter:

```
def do_stuff_with_param(message):
    print("----------")
    print(message)
    print("----------")
```

In this case, `message` is the name of the function's parameter. Invoke it like so:

```
do_stuff_with_param("HELLO!")
#> ---------
#> HELLO!
#> ---------

do_stuff_with_param("HELLO AGAIN!")
#> ---------
#> HELLO AGAIN!
#> ---------
```

```
m = "HELLO!"
do_stuff_with_param(m)
#> ---------
#> HELLO!
#> ---------

s = "HELLO AGAIN!"
do_stuff_with_param(s)
#> ---------
#> HELLO AGAIN!
#> ---------
```

Multiple Parameters

Defining a function with multiple parameters:

```
def do_stuff_with_params(message, first_name, last_name):
    print("'" + message + "', says " + first_name + " " + last_name)
```

The order of the parameters during function definition corresponds with the order they should be passed during function invocation. In this case, `message`, `first_name` and `last_name` are the names of the function's parameters, in that order. So we can invoke the function like so:

```
do_stuff_with_params("HELLO THERE", "Ophelia", "Clark")
#> 'HELLO THERE', says Ophelia Clark
```

It is possible to pass the parameters in a different order, by explicitly specifying the parameter name during function invocation:

```
do_stuff_with_params(first_name= "Ophelia", message="HELLO THERE",
last_name="Clark")
#> 'HELLO THERE', says Ophelia Clark
```

Return Values

In order to have a function return some value when invoked, use the `return` keyword:

```
def calculate_area(length, height):
  length * height

area = calculate_area(4, 2)
print(area) #> None
```

```
def calculate_area(length, height):
  return length * height

area = calculate_area(4, 2)
print(area) #> 8
```