DEFINING A FUNCTION

```
def function_name(arguments):
```

Indicates you're defining a new function

An intuitive name to call you function by

Examples:
- *avg*
- *usd_converter*

Variable names for the function inputs, separated by commas

Functions have the same **naming rules** and best practices as variables – use 'snake_case'!

DEFINING A FUNCTION

```
def function_name(arguments):
    do this
```

Code block for the function that uses the arguments to perform a specific task

DEFINING A FUNCTION

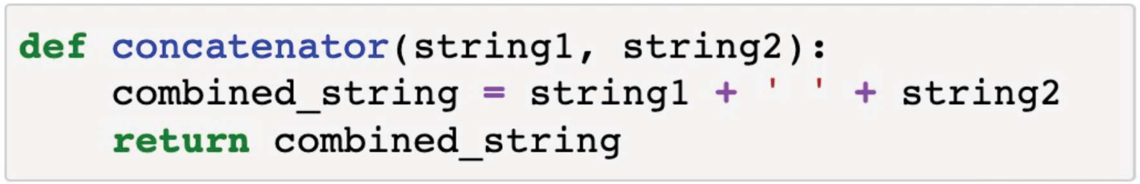
```
def function_name(arguments):
    do this
    return output
```

Ends the function
(without it, the function returns None)

Values to return
(usually variables)

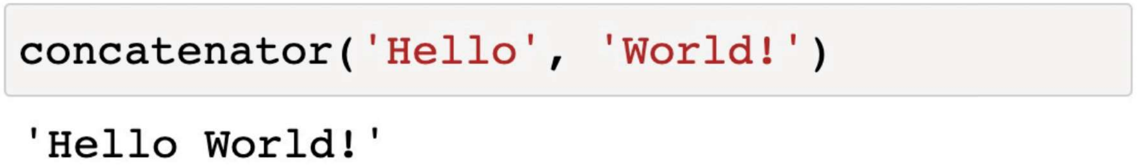
DEFINING A FUNCTION

EXAMPLE *Defining a function that concatenates two words, separated by a space*



```python
def concatenator(string1, string2):
    combined_string = string1 + ' ' + string2
    return combined_string
```

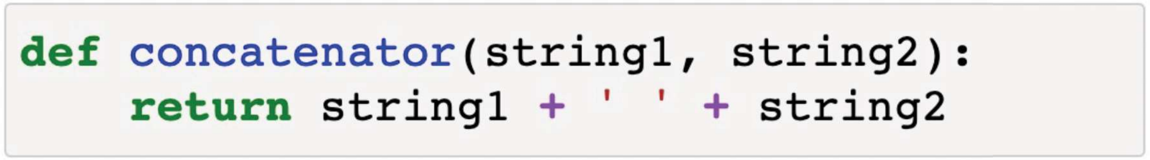
Here we're defining a function called ***concatenator****, which accepts two arguments, combines them with a space , and returns the result*



```python
concatenator('Hello', 'World!')
```

```
'Hello World!'
```

When we call this function with two string arguments, the combined string is returned



```python
def concatenator(string1, string2):
    return string1 + ' ' + string2
```

Note that we don't need a code block before return, here we're combining strings in the return statement