Signature

- Calls to the function will look like this (with the same name and number of arguments). Example: `double(3)`.
- When you call `double`, the argument can be any expression. (The name `x` doesn't affect calls.)
- In the body of the function, `x` is the name of the argument, as if the body included the code `x = <the first argument>`.

```
# Our first function definition

def double(x):
    """ Double x """
    return 2*x
```

Documentation ("docstring")

- Text that describes what the function does.
- Can be any string, traditionally triple-quoted so it can span several lines.
- Traditionally, the first line describes what the function does, briefly.
- Subsequent lines can give more detail and examples.
- Running `double?` will show this text, just like `max?` will show the documentation for the built-in function `max`.

Body

- All the code in here runs each time you call the function.
- The special statement `return` tells Python what the value of each call to this function is: it's the value of the expression after `return`.
- For example, the value of `double(3)` is `6`. (Remember, when the argument is `3`, it's like the body starts with `x = 3`.)
- Often, the body will have multiple lines of code that build up to computing the `return`ed value. You can write any Python code here that you could write anywhere else.

Indentation

- Each line of code in the body is indented (that is, it's preceded by spaces).
- Traditionally, we use 2 or 4 spaces. They only need to be consistent.
- This tells Python that those lines are part of the body.
- The function's body ends at any unindented line.

```python
def biggest_difference(array_x):
    """Find the biggest difference in absolute value between two adjacent elements of array_x."""
    diffs = np.diff(array_x)
    absolute_diffs = abs(diffs)
    return max(absolute_diffs)

some_numbers = make_array(2, 4, 5, 6, 4, -1, 1)
biggest_difference(some_numbers)
```

```python
def biggest_difference(array_x):
    """Find the biggest difference in absolute value between two adjacent elements of array_x."""
    diffs = np.diff(array_x)
    absolute_diffs = abs(diffs)
    return max(absolute_diffs)

some_numbers = make_array(2, 4, 5, 6, 4, -1, 1)
biggest_difference(some_numbers)
```

```python
def biggest_difference(array_x):
    """Find the biggest difference in absolute value between two adjacent elements of array_x."""
    diffs = np.diff(array_x)
    absolute_diffs = abs(diffs)
    return max(absolute_diffs)

some_numbers = make_array(2, 4, 5, 6, 4, -1, 1)
biggest_difference(some_numbers)
```

```python
def biggest_difference(array_x):
    """Find the biggest difference in absolute value between two adjacent elements of array_x."""
    diffs = np.diff(array_x)
    absolute_diffs = abs(diffs)
    return max(absolute_diffs)

some_numbers = make_array(2, 4, 5, 6, 4, -1, 1)
biggest_difference(some_numbers)
```

```python
def biggest_difference(array_x):
    """Find the biggest difference in absolute value between two adjacent elements of array_x."""
    diffs = np.diff(array_x)
    absolute_diffs = abs(diffs)
    return max(absolute_diffs)

some_numbers = make_array(2, 4, 5, 6, 4, -1, 1)
biggest_difference(some_numbers)
```

Names defined in scope

```
def biggest_difference(array_x):
    """Find the biggest difference in absolute value between two adjacent elements of array_x."""
    diffs = np.diff(array_x)
    absolute_diffs = abs(diffs)
    return max(absolute_diffs)

some_numbers = make_array(2, 4, 5, 6, 4, -1, 1)
big_diff = biggest_difference(some_numbers)
print("The biggest difference is", big_diff)
```

The function `biggest_difference` is defined.

```
def biggest_difference(array_x):
    """Find the biggest difference in absolute value between two adjacent elements of array_x."""
    diffs = np.diff(array_x)
    absolute_diffs = abs(diffs)
    return max(absolute_diffs)

some_numbers = make_array(2, 4, 5, 6, 4, -1, 1)
big_diff = biggest_difference(some_numbers)
print("The biggest difference is", big_diff)
```

The array `some_numbers` is defined.

```
biggest_difference = <a function>
```

```
def biggest_difference(array_x):
    """Find the biggest difference in absolute value between two adjacent elements of array_x."""
    diffs = np.diff(array_x)
    absolute_diffs = abs(diffs)
    return max(absolute_diffs)

some_numbers = make_array(2, 4, 5, 6, 4, -1, 1)
big_diff = biggest_difference(some_numbers)
print("The biggest difference is", big_diff)
```

Our function is called. Before this line finishes, Python executes its body…

```
biggest_difference = <a function>
some_numbers = [2, 4, …]
```

```
def biggest_difference(array_x):
    """Find the biggest difference in absolute value between two adjacent elements of array_x."""
    diffs = np.diff(array_x)
    absolute_diffs = abs(diffs)
    return max(absolute_diffs)

some_numbers = make_array(2, 4, 5, 6, 4, -1, 1)
big_diff = biggest_difference(some_numbers)
print("The biggest difference is", big_diff)
```

The argument is given the name `array_x`. The function's first line does nothing.

```
biggest_difference = <a function>
array_x = [2, 4, …]
```

```
def biggest_difference(array_x):
    """Find the biggest difference in absolute value between two adjacent elements of array_x."""
    diffs = np.diff(array_x)
    absolute_diffs = abs(diffs)
    return max(absolute_diffs)

some_numbers = make_array(2, 4, 5, 6, 4, -1, 1)
big_diff = biggest_difference(some_numbers)
print("The biggest difference is", big_diff)
```

The array `diffs` is defined.

```
biggest_difference = <a function>
array_x = [2, 4, …]
```

```
def biggest_difference(array_x):
    """Find the biggest difference in absolute value between two adjacent elements of array_x."""
    diffs = np.diff(array_x)
    absolute_diffs = abs(diffs)
    return max(absolute_diffs)

some_numbers = make_array(2, 4, 5, 6, 4, -1, 1)
big_diff = biggest_difference(some_numbers)
print("The biggest difference is", big_diff)
```

The array `absolute_diffs` is defined.

```
biggest_difference = <a function>
array_x = [2, 4, …]
diffs = [2, 1, 1, -2, …]
```

```
def biggest_difference(array_x):
    """Find the biggest difference in absolute value between two adjacent elements of array_x."""
    diffs = np.diff(array_x)
    absolute_diffs = abs(diffs)
    return max(absolute_diffs)

some_numbers = make_array(2, 4, 5, 6, 4, -1, 1)
big_diff = biggest_difference(some_numbers)
print("The biggest difference is", big_diff)
```

The value of `max(absolute_diffs)` is computed and becomes the value of the call `biggest_difference(some_numbers)`.

```
biggest_difference = <a function>
array_x = [2, 4, …]
diffs = [2, 1, 1, -2, …]
absolute_diffs = [2, 1, 1, 2, …]
```

```
def biggest_difference(array_x):
    """Find the biggest difference in absolute value between two adjacent elements of array_x."""
    diffs = np.diff(array_x)
    absolute_diffs = abs(diffs)
    return max(absolute_diffs)

some_numbers = make_array(2, 4, 5, 6, 4, -1, 1)
big_diff = biggest_difference(some_numbers)
print("The biggest difference is", big_diff)
```

The function call is done, so `array_x`, `diffs`, and `absolute_diffs` disappear. `some_numbers` reappears, and `big_diff` is defined as the value of the call. Finally, the `print` statement happens.

```
biggest_difference = <a function>
some_numbers = [2, 4, …]
big_diff = 5
```