

Functions

ESS 116 | Fall 2024

Prof. Henri Drake, Prof. Jane Baldwin, and Prof. Michael Pritchard

(Modified from Ethan Campbell and Katy Christensen's materials for UW's Ocean 215)

What we'll cover in this lesson

1. Functions and arguments

Functions and arguments

Function name An “argument” or “parameter”



len ([6 , 8 , 7 , 5])



The function “returns” or “evaluates to” the integer 4

Some functions act on a target

The “target” of the function



Function name



`'python' . upper ()`



The function returns `'PYTHON'`

Values returned by functions can be stored in a variable

The “target” of the function

Function name



`'python'.upper()`

```
new_string = 'python'.upper()
```

Some functions don't return anything

```
numbers = [ 6 , 8 , 7 , 5 ]
```

```
numbers.sort()
```

This function returns nothing at all!


It simply modifies `numbers` “in-place,” which becomes `[5 , 6 , 7 , 8]`.

Some functions have named arguments

“Named” or “keyword” argument

Argument value

`numbers.sort(reverse=True)`



Now, `numbers` will be sorted in reverse order: `[8 , 7 , 6 , 5]`.

Named arguments always have a default value

The “default” value of `reverse`



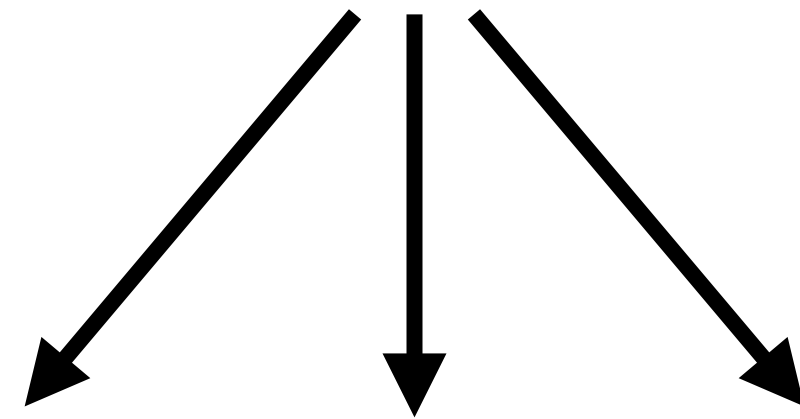
```
numbers.sort(reverse=False)
```

is equivalent to:

```
numbers.sort()
```


Functions can have **both** positional and named arguments

“Positional” arguments have a fixed order



function_name(**arg1**, **arg2**, **arg3**, ..., **named_arg1**=default1,
named_arg2=default2, ...)

“Named” arguments can be provided in any order,
but they must follow any positional arguments