POSITIONAL AND KEYWORD ARGUMENTS

Positional Arguments

Most common way of assigning arguments to parameters: via the order in which they are passed

i.e. their position

```
def my_func(a, b):
    # code …
```

```
my_func(10, 20)    → a = 10, b = 20

my_func(20, 10)    → a = 20, b = 10
```

Default Values

A positional arguments can be made optional by specifying a default value for the corresponding parameter

```
def my_func(a, b=100):
    # code …
```

```
my_func(10, 20)      → a = 10, b = 20

my_func(5)           → a = 5, b = 100
```

Consider a case where we have three arguments, and we want to make one of them optional:

```
def my_f[illegible]c(a[illegible] b=100, c):
    # cod[illegible]…
```

How would we call this function without specifying a value for the second parameter?

```
my_fu[illegible](5[illegible]5) ???
```

If a positional parameter is defined with a default value

every positional parameter after it

must also be given a default value

Default Values

```
def my_func(a, b=5, c=10):
    # code …
```

```
my_func(1)          → a = 1, b = 5, c = 10

my_func(1, 2)       → a = 1, b = 2, c = 10

my_func(1, 2, 3)    → a = 1, b = 2, c = 3
```

But what if we want to specify the 1st and 3rd arguments, but omit the 2nd argument?

i.e. we want to specify values for a and c, but let b take on its default value?

→ Keyword Arguments (named arguments)

```
my_func(a=1, c=2)       → a = 1, b = 5, c = 2

my_func(1, c=2)         → a = 1, b = 5, c = 2
```

Keyword Arguments

Positional arguments can, optionally, be specified by using the parameter name

whether or not the parameters have default values

```
def my_func(a, b, c)        my_func(1, 2, 3)

                            my_func(1, 2, c=3)
                                                        → a=1, b=2, c=3
                            my_func(a=1, b=2, c=3)

                            my_func(c=3, a=1, b=2)
```

But once you use a named argument, all arguments thereafter must be named too

```
my_func(c=1, 2, 3) ✗

my_func(1, b=2, 3) ✗        my_func(1, b=2, c=3) ✓
                            my_func(1, c=3, b=2) ✓
```

Keyword Arguments

All arguments after the first named (keyword) argument, must be named too

Default arguments may still be omitted

```
def my_func(a, b=2, c=3)


my_func(1)              → a=1, b=2, c=3

my_func(a=1, b=5)       → a=1, b=5, c=3

my_func(c=0, a=1)       → a=1, b=2, c=0
```

Code