Named Parameters, Default Parameters, and Flexible Numbers of Parameters

---

Named Parameters

Named Parameters allow you to specify parameters in any order when calling a function:

```
def PrintInfo(name, age, friends):
    print(name, age, friends)

PrintInfo(age = 10, name = 'Lucy', friends = [])
```

This is useful for functions with many parameters, where it's hard to remember the order they have to be in, and in combination with default parameters.

Default Parameters

Default parameters allow you to set a default value for if the function is called without providing that parameter.

```
def PrintInfo(name = None, age = None, friends = None):
    print(name, age, friends)

# You can now call this function with only some of the parameters

PrintInfo(age = 23)     => None, 23, None
PrintInfo('Ann', 30)    => 'Ann', 30, None
PrintInfo('Bob')        => 'Bob', None, None
```

Flexible Numbers of Parameters

You can allow a function be called with a completely variable number of parameters (like `print()`) by giving one parameter a name beginning with an asterisk:

```
def PrintMultiple(*items):
    for item in items:
        print(item)

PrintMultiple(1, 2, 3, 4, 5)
```

The parameter comes into the function as a tuple.