**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Topic 45 - Functions: Mixing Positional and Keyword Arguments**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**What**

In Python functions, you can combine **positional arguments** and **keyword arguments** when calling a function. Positional arguments map to parameters by their position, while keyword arguments use a key-value pairing to map to parameters by name.

**Why**

Mixing positional and keyword arguments allows for **more readable** and **flexible** function calls. You can pass essential arguments positionally and specify additional arguments using keywords, especially if the function has default parameters.

**How**

1. **Using Both Positional and Keyword Arguments**
   * In the function call, **positional arguments must come before keyword arguments**.
   * If you break this rule, Python will throw an error.

python

Copy code

def give\_greeting(greeting, first\_name):

print(greeting + ", " + first\_name)

give\_greeting("Hello there", first\_name="Al")

# Output: Hello there, Al

1. **Including Default Values in the Mix**
   * Functions can have **default values for parameters**, which become optional in the function call.
   * Default parameters must come after non-default positional parameters in the function definition.

python

Copy code

def give\_greeting(greeting, first\_name, flattering\_nickname=" the wonder boy"):

print(greeting + ", " + first\_name + flattering\_nickname)

give\_greeting("Hello there", first\_name="Al")

# Output: Hello there, Al the wonder boy

1. **Using Lists and Dictionaries as Arguments**
   * Lists and dictionaries can be passed as arguments to functions, allowing for complex data manipulation within the function.

python

Copy code

customers = {

0: {"first name": "John", "last name": "Ogden", "address": "301 Arbor Rd."},

1: {"first name": "Ann", "last name": "Sattermyer", "address": "PO Box 1145"},

2: {"first name": "Jill", "last name": "Somers", "address": "3 Main St."},

}

def find\_something(dict, inner\_dict, target):

print(dict[inner\_dict][target])

# Example call

find\_something(customers, 2, "last name")

# Output: Somers

**Things to Remember**

* Positional arguments come before keyword arguments in function calls.
* Parameters with default values must come last in the function definition.
* Lists, dictionaries, strings, and numbers can all be passed as arguments to functions.

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**