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

**Topic 43 - Functions: Passing Information with Keyword Arguments**

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

**What**

In Python, when calling a function, you can pass arguments by specifying each one with a **keyword**. This lets you match each argument to a parameter by name, rather than by order, making the function call more readable and flexible.

**Why**

Keyword arguments allow for greater flexibility in the order of arguments and improve the readability of your code, making it easy to understand which value is assigned to which parameter.

**How**

1. **Using Keyword Arguments**:
   * You can specify each argument with a **keyword** (parameter name) and an **equal sign**, followed by the value you want to pass.
   * The function definition will match each value with the corresponding parameter based on the keyword name, regardless of order.

python

Copy code

def say\_names\_of\_couple(husband\_name, wife\_name):

print("The names of the couple are " + husband\_name + " and " + wife\_name)

# Calling with keyword arguments

say\_names\_of\_couple(husband\_name="Bill", wife\_name="Zelda")

# Output: The names of the couple are Bill and Zelda

1. **Order Doesn't Matter**:
   * With keyword arguments, the **order** in which you pass them doesn’t matter.
   * You can specify the parameters in any order, as long as you use the correct keywords.

python

Copy code

say\_names\_of\_couple(wife\_name="Zelda", husband\_name="Bill")

# Output: The names of the couple are Bill and Zelda

1. **Mixing Positional and Keyword Arguments**:
   * You can use positional arguments along with keyword arguments, but **positional arguments** must come **before** any keyword arguments in the function call.

python

Copy code

def introduce\_person(name, age):

print(f"{name} is {age} years old.")

introduce\_person("Alice", age=30) # Valid

# Output: Alice is 30 years old.

* + The following will cause an error because the positional argument comes after the keyword argument:

python

Copy code

introduce\_person(age=30, "Alice") # Invalid

**Things to Remember**

* Keyword arguments allow you to pass arguments to a function in any order.
* Always ensure positional arguments come before keyword arguments if mixing the two.
* Using keyword arguments can make your function calls more descriptive and easier to understand.

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