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

**Topic 46 - Functions: Dealing with an Unknown Number of Arguments**

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

**What**

In Python, functions can accept an **unknown number of arguments**. There are two ways to handle this:

1. Using \*args for an unknown number of **positional arguments**.
2. Using \*\*kwargs for an unknown number of **keyword arguments**.

**Why**

Handling unknown arguments allows functions to be **more flexible** and capable of accepting **varying inputs** based on the needs of the caller. It’s useful in cases where you might not know in advance how many arguments will be passed.

**How**

1. **Accepting Unknown Keyword Arguments (\*\*kwargs)**
   * Use two asterisks \*\* before a parameter name to collect **all additional keyword arguments** into a dictionary.
   * Each additional argument will be stored as a **key-value pair** in the dictionary.

python

Copy code

def display\_result(winner, score, \*\*other\_info):

print("The winner was " + winner)

print("The score was " + score)

for key, value in other\_info.items():

print(key + ": " + value)

# Example call with extra keyword arguments

display\_result(winner="Real Madrid", score="1-0", overtime="yes", injuries="none")

# Output:

# The winner was Real Madrid

# The score was 1-0

# overtime: yes

# injuries: none

1. **Accepting Unknown Positional Arguments (\*args)**
   * Use a single asterisk \* before a parameter name to collect **all additional positional arguments** into a tuple.
   * Each additional argument will be stored in the tuple in the order it was passed.

python

Copy code

def display\_nums(first\_num, second\_num, \*opt\_nums):

print(first\_num)

print(second\_num)

print(opt\_nums)

# Example call with extra positional arguments

display\_nums(100, 200, 300, 400, 500)

# Output:

# 100

# 200

# (300, 400, 500)

**Things to Remember**

* **Order matters**: Regular parameters come first, followed by \*args (if used), and then \*\*kwargs.
* **Data type differences**: \*args collects positional arguments into a tuple, while \*\*kwargs collects keyword arguments into a dictionary.

Using \*args and \*\*kwargs enables functions to handle a wide variety of argument combinations, enhancing their flexibility and adaptability.

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