**Step 1: Define a Function with Variable-Length Arguments**

def average(\*t):

* The \*t syntax allows the function to accept any number of positional arguments.
* t is treated as a **tuple**, meaning all arguments passed to the function will be stored inside a tuple.

**Step 2: Calculate the Average**

avg = sum(t) / len(t)

* The sum(t) function adds up all the numbers inside the tuple t.
* The len(t) function counts how many numbers are in the tuple.
* The division sum(t) / len(t) calculates the **average** of the numbers.

**Step 3: Round the Result**

return round(avg, 2)

* The round(avg, 2) function rounds the average to **2 decimal places** for better readability.
* The function then **returns** the rounded average.

**Step 4: Calling the Function with Different Arguments**

result1 = average(32, 44, 23, 12, 67, 54, 87, 65, 90, 23, 11)

* Here, the function is called with **11 numbers**.
* The function calculates their sum and divides by **11** to get the average.

result2 = average(45, 77, 87, 23, 52, 21, 89, 8, 43, 59, 11, 24, 43)

* The function is called again with **13 numbers**.
* The sum is divided by **13** to compute the average.

**Step 5: Printing the Results**

print(f"Average is: {result1}")

print(f"Average is: {result2}")

* The calculated averages are displayed using an **f-string**, which makes printing formatted strings easier.

**Example Output**

Average is: 49.09

Average is: 45.85

* The output depends on the input numbers, but it will always show the average rounded to 2 decimal places.

**Why Use Variable-Length Arguments?**

* If we use normal parameters like average(a, b, c), we **must** pass exactly 3 values.
* With \*args, we can pass **any number of values**.
* This makes the function more **flexible** and **reusable**.