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

**Topic 48 - Using Functions as Variables**

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

**What**

Functions in Python **act like variables**—they can be **stored**, **passed around**, and even **used directly in expressions**. Instead of assigning the result of a function call to a separate variable, you can use the function itself wherever a variable might appear.

**Why**

Treating functions as variables allows:

* **Condensed, readable code** by directly using function calls in expressions.
* **Flexibility**: Functions can be nested or combined in a single line, making complex calculations more streamlined.

**How**

1. **Basic Example**:  
   Instead of using separate variables to store function results, call the functions directly in an expression.

python

Copy code

def add\_numbers(first\_number, second\_number):

return first\_number + second\_number

def subtract\_numbers(first\_number, second\_number):

return first\_number - second\_number

# Traditional way

result\_of\_adding = add\_numbers(1, 2)

result\_of\_subtracting = subtract\_numbers(3, 2)

sum\_of\_results = result\_of\_adding + result\_of\_subtracting

print(sum\_of\_results) # Output: 4

# Using functions as variables in one line

sum\_of\_results = add\_numbers(1, 2) + subtract\_numbers(3, 2)

print(sum\_of\_results) # Output: 4

1. **Using Functions Directly in Expressions**  
   You can call functions directly inside print statements, comparisons, or other function calls.

python

Copy code

print(add\_numbers(5, 10)) # Output: 15

1. **Nesting Functions**  
   Functions can be **nested** within expressions as needed. For example:

python

Copy code

result = add\_numbers(subtract\_numbers(10, 5), 3)

print(result) # Output: 8

**Things to Remember**

* Functions, like variables, **hold values**. You can use them wherever values are accepted.
* **Order matters** when combining or nesting functions. Python evaluates from the innermost function outward.
* Be careful with **complex expressions**; readability is as important as conciseness.

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