DAY 12 **[08 - JULY - 2025]**

FUNCTIONS IN PYTHON

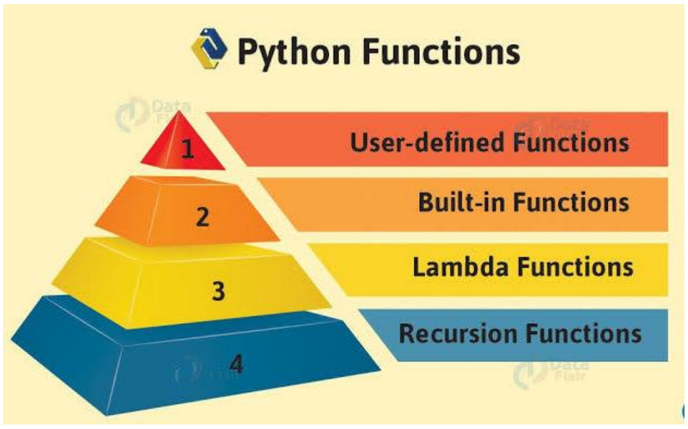
Python Functions is a block of statements that does a specific task. The idea is to put some commonly or repeatedly done task together and make a function so that instead of writing the same code again and again for different inputs, we can do the function calls to reuse code contained in it over and over again.

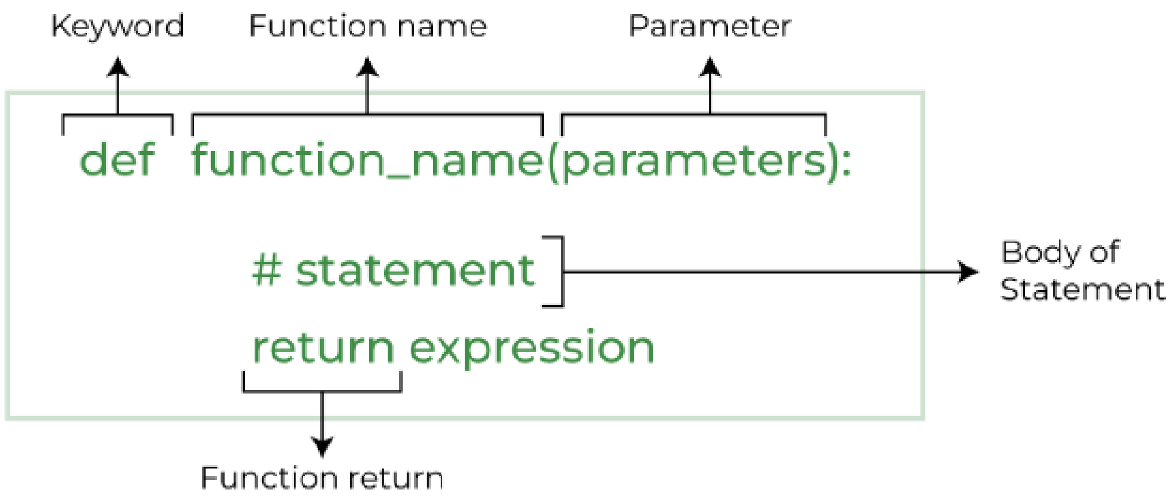
Benefits of Using Functions

- Code Reuse
- Reduced code length
- Increased redability of code

Python Function Declaration

The syntax to declare a function is:





Types of functions:

- Built-in function: These are Standard functions in Python that are available to use.
- User-defined function: We can create our own functions based on our requirements.
- Lambda Function: a small, anonymous function defined without a name, often used for concise, single-expression operations.
- Recursive Function: a function that calls itself within its own code.

Arguments in Functions

Arguments are values passed to a function when it is called. They can be categorized as follows:

1. Positional Arguments: These are arguments that are passed to a function in a specific order.

2. Keyword Arguments: These are arguments passed to a function by explicitly naming each parameter and its corresponding value.

3. Default Arguments: These are arguments that assume a default value if no value is provided during the function call

4. Variable-length Arguments:

**args* (Non-keyword arguments): Allows a function to accept any number of positional arguments.

***kwargs* (Keyword arguments): Allows a function to accept any number of keyword arguments

Arbitrary keyword arguments allow a function to accept an unspecified number of keyword arguments. In Python, this is done using **kwargs in the function definition. kwargs stands for "keyword arguments" and allows you to handle named arguments that are not specified in advance.

In Python, *args and* *kwargs are used to allow functions to accept an arbitrary number of arguments.

Python *args The special syntax *args in function definitions is used to pass a variable number of arguments to a function. It is used to pass a non-keyworded, variable-length argument list.

Python **kwargs The special syntax* *kwargs in function definitions is used to pass a variable length argument list. We use the name kwargs with the double star **.

```
print("Args example")
def fun(*args):            #non keyword arguments
    return sum(args)

print(fun(1, 2, 3, 4))
print(fun(5, 10, 15))
```

```python
print("\mkwargs example")
def fun(**kwargs):      #keyword arguments
    for k, val in kwargs.items():
        print(k, val)

fun(a=1, b=2, c=3)
```

```
Output:

Args example
10
30

kwargs example
a 1
b 2
c 3
```