8) Functions

- **Defining functions in Python.**

A **function** in Python is a reusable block of code that performs a specific task. It helps reduce repetition and makes your code more organized.

In Python, you define one using def name(params): followed by an indented block. You call it by writing name(args)

They support code reuse, avoiding duplication Mathematical Foundation

Basic Structure of a Function

1. Basic Structure of a Function
2. def keyword – tells Python you're defining a function
3. Function name – should follow naming rules (no spaces, can't start with numbers)
4. Parameters (optional) – values passed into the function
5. Colon : – indicates the start of the function block
6. Indented code block – contains the logic of the function
7. return (optional) – sends a result back to the caller

```
def function_name(parameter1, parameter2):
    # Function body
    result = parameter1 + parameter2
    return result
```

Types of Functions:

1. Built-in Functions: Provided by Python (len(), print()).
2. User-defined Functions: Created using def.
3. Lambda Functions: Anonymous, one-line functions using the lambda keyword.

- **Different types of functions: with/without parameters, with/without return values.**

  Functions in Python can be categorized into four main types depending on:

  - Whether they accept parameters
  - Whether they return a value

1. **Functions Without Parameters and Without Return Values**

    - Purpose: Perform a task without needing input or returning a result.

```
def function_name():
    # code block
    print("This is a simple function.")
```

2. **Functions With Parameters and Without Return Values.**

    - Purpose: Accept input, process it, but don’t return anything.

```
def function_name(parameter1, parameter2):
    # code block using parameters
    print("Result is:", parameter1 + parameter2)
```

3. **Functions Without Parameters and With Return Values**

    - These functions do not take any input (no parameters).

- But they perform a task and return a value using the return statement.

```
def function_name():
    # logic inside
    return value
```

4. Functions With Parameters and With Return Values

- These functions accept inputs (parameters).
- They process those inputs and return a result using the return statement.

```
def function_name(param1, param2):
    # process input parameters
    return result
```

- **Anonymous functions (lambda functions).**

An anonymous function is a function without a name. In Python, these are created using the lambda keyword and are commonly referred to as lambda functions.

Syntax:

```
lambda arguments: expression
```

- Anonymous functions are functions without a name.
- In Python, these are created using the keyword lambda.
- They are generally short, one-line functions used for simple operations.
- Lambda functions can take any number of arguments but can only have one expression.
- The expression is automatically returned (no need to use return).