**7. Functions and Methods**

1. **Defining and calling functions in Python**.

Function is a Block of Code That we can use again and again **Types of Function :**

1. Built in Function ( print(), input() )
2. User Define

Functions are fundamental building blocks in Python that allow you to organize and reuse code. Here's a comprehensive explanation of how to define and call functions in Python.

# Basic Syntax :

def function\_name(parameters):

"""docstring (optional)"""

# function body return value # optional **Calling Functions** greet("Alice") # Output: Hello, Alice!

# Function Components

1. def keyword: Starts the function definition
2. Function name: Follows Python naming conventions

(lowercase\_with\_underscores)

1. Parameters: Variables listed in parentheses (can be empty)
2. Docstring: Optional documentation string (first statement in function)
3. Function body: Indented block of code
4. return statement: Optional, exits function and returns value(s)

# Scope Rules

* Local scope: Variables defined inside a function
* Enclosing scope: For nested functions
* Global scope: Variables defined at module level
* Built-in scope: Python's built-in names

# Function arguments (positional, keyword, default)

Python provides flexible ways to pass arguments to functions. Understanding these different argument types is essential for writing clean, maintainable code.

## 1. Positional Arguments

**Definition**: Arguments passed in the exact order of the function's parameters.

Characteristics:

* Matched based on position (order matters)
* Most basic and common argument type
* Must be passed before any keyword arguments

## 2. Keyword Arguments

**Definition**: Arguments passed with an explicit parameter name.

Characteristics:

* Matched by parameter name rather than position
* Can be passed in any order
* Make function calls more readable
* Must come after positional arguments (if both are used)

## 3. Default Arguments

**Definition**: Parameters that assume a default value if no argument is provided.

Characteristics:

* Defined in the function definition
* Must appear after non-default parameters
* Allow functions to be called with fewer arguments
* Should use immutable defaults (avoid mutable defaults like lists/dicts unless intended)

# Scope of variables in Python

## 1. Local Scope (Function-level)

Variables defined inside a function have local scope and are only accessible within that function.

## 2. Enclosing Scope (Non-local)

For nested functions, variables from the outer function are accessible in the inner function (but not modifiable unless declared as nonlocal).

## 3. Global Scope (Module-level)

Variables defined at the top level of a module (outside all functions) have global scope and are accessible throughout the module.

## 4. Built-in Scope

Python's built-in names (like print, len, str, etc.) have the widest scope and are available everywhere.

# Built-in methods for strings, lists, etc

Python provides many useful built-in methods for working with strings, lists, dictionaries, tuples, and sets. Here's a comprehensive overview:

## 1.String Methods

Python strings are immutable sequences of Unicode characters with these key method categories:

## Case Manipulation Methods

* upper()/lower(): Convert case
* title(): Title case conversion
* capitalize(): Capitalize first character
* swapcase(): Swap cases

## Search and Validation Methods

* find()/index(): Locate substrings
* startswith()/endswith(): Check prefixes/suffixes
* isalpha()/isdigit()/isalnum(): Character type checks **Transformation Methods**
* replace(): Substring replacement
* split()/rsplit(): String splitting
* join(): Sequence concatenation
* format(): String formatting

## 2.List Methods

Lists are mutable sequences with these fundamental operations: **Modification Methods**

* append(): Add single element
* extend(): Add multiple elements
* insert(): Insert at position
* remove()/pop(): Element removal

## Organizational Methods

* sort(): In-place sorting
* reverse(): Order reversal
* copy(): Shallow copying

## Information Methods

* index(): Position finding
* count(): Occurrence counting

**3.Dictionary Methods**

Dictionaries are mutable mappings with these core operations:

## Access Methods

* get(): Safe key access
* keys()/values()/items(): View objects
* setdefault(): Get with default insertion

## Modification Methods

* update(): Multiple key updates  pop()/popitem(): Item removal
* clear(): Complete emptying

## 4.Tuple Methods

Immutable sequences with limited functionality:

* index(): Position finding
* count(): Occurrence counting