**Functions**

Definition: A block of organized, reusable code used to perform a single, related action.

Syntax:

def function\_name(parameters):

# function body

return [expression]

Parameters: Inputs to the function (can be optional).

Return Statement: Sends back a result from the function (optional).

Function Types

Built-in Functions: Predefined functions in Python like print(), len(), etc.

User-defined Functions: Functions defined by the user for specific tasks.

Function Characteristics

Reusability: Functions can be called multiple times within a program.

Modularity: Encapsulating functionality into functions enhances code organization.

Abstraction: Users can utilize functions without knowing the internal implementation.

Arguments in Functions

Positional Arguments: Defined by their position in the function call.

Keyword Arguments: Associated with parameter names in the function call.

Default Arguments: Parameters with preset default values.

Variable Scope

Global Variables: Defined outside functions, accessible throughout the program.

Local Variables: Defined within a function, accessible only within that function.

Lambda Functions

Anonymous Functions: Small, unnamed functions defined using the lambda keyword.

Syntax:

lambda arguments: expression

Recursion

Function calling itself: Recursive functions solve problems in smaller instances.

Base Case: Condition where the function stops calling itself.

Function Decorators

Modify or extend behavior of functions: Functions that take another function and extend its functionality.

Docstrings

Documentation Strings: Description of a function's purpose, usage, and parameters using triple quotes '''...'''.

Passing Functions as Arguments

Functions can be passed as arguments to other functions.