FUNCTIONS AND MODULES:

A FUNCTION IS A BLOCK OF REUSABLE CODE THAT PERFORM SPECIFIC TASK

- User defined
- Built in

User defined: A user-defined function is a function that is created by the user (or programmer) to perform a specific task. Unlike built-in functions, which are provided by the programming language, user-defined functions allow you to create reusable blocks of code that can be called multiple times throughout your program.

Built in:A built-in function is a function that is already provided by a programming language or its standard library. These functions are ready to be used in your code without needing to define them yourself. They perform common operations like mathematical calculations, input/output handling, data manipulation, and more.

Argumentation:

argumentation refers to the process of passing values (arguments) to functions when they are called. These arguments provide input to the function, and the function processes them to return the desired result.

Types of argumentation:

- Positional argument
- Keyword argument
- Default argument
- Variable length argument

1. **Positional Arguments**:

- **Definition**: These are the most common type of arguments. The values are passed to the function in the order (position) in which the parameters are defined in the function signature.

Example:
python
Copy

```
def add(a, b):
    return a + b

print(add(3, 5))  # Output: 8 (3 is assigned to a, 5 to b)
```

-

3. Keyword Arguments:

- **Definition**: These arguments are passed to a function by explicitly naming the parameter and assigning a value to it. They are not dependent on the order in which they are passed.

2. **Default Arguments**:

- **Definition**: These are arguments that have default values specified in the function definition. If no value is provided for these parameters, the default value is used.

Example:
 python
Copy
4. Variable-Length Arguments (Arbitrary Arguments):

- **Definition**: These allow a function to accept an arbitrary number of arguments. There are two types of variable-length arguments in Python

- **(Non-keyword Variable-Length Arguments)**: Allows passing a variable number of non-keyword arguments (i.e., a tuple).

Example:
 python
Copy

```
def greet(*names):
    for name in names:
        print(f"Hello, {name}")

greet("Alice", "Bob", "Charlie")  # Accepts any
number of positional arguments
```

  ○

- **`**kwargs` (Keyword Variable-Length Arguments)**: Allows passing a variable number of keyword arguments (i.e., a dictionary).

Example:
python
Copy

```
def greet_with_info(**info):
    for key, value in info.items():
        print(f"{key}: {value}")
```

- `greet_with_info(name="Alice", age=30, city="New York")`