Functions in Python

In Python, a function is a reusable block of code that performs a specific task. Functions help in structuring code, making it more readable, maintainable, and reusable. Python provides both built-in functions (like print(), len(), and sum()) and allows users to define custom functions.

Defining a Function

A function in Python is defined using the def keyword, followed by the function name and parentheses containing optional parameters. The function body is indented and typically includes a return statement.

```
def greet(name):
    return f"Hello, {name}!"


print(greet("Alice"))  # Output: Hello, Alice!
```

Function Parameters and Arguments

- **Positional Arguments:** Passed in the order defined in the function.
- **Keyword Arguments:** Passed with parameter names, allowing flexibility.
- **Default Arguments:** Have default values if not provided.
- **Arbitrary Arguments:** *args for multiple positional arguments and **kwargs for multiple keyword arguments.

```
def add(a, b=10):  # Default argument
    return a + b


print(add(5))  # Output: 15
```

Lambda Functions

Python supports anonymous functions using lambda:

```
square = lambda x: x * x

print(square(4))  # Output: 16
```

Scope and Return Values

- Variables inside a function have **local scope** unless declared global.
- Functions can return multiple values using tuples.

Higher-Order Functions

Functions can be passed as arguments, supporting functional programming.

```python
def apply(func, value):
    return func(value)


print(apply(lambda x: x * 2, 5))  # Output: 10
```

Functions in Python enhance modularity, improve readability, and facilitate reusability, making them essential for efficient programming.