Trisha

AI/ML Enthusiast | Intern @CodroidHub | First-Year Engineering Student

Artificial Intelligence/Machine Learning UpSkills Notebook

From Basics to Real-World — Starts Your ML Journey

Agenda — Functions in Python

In this notebook, you will learn:

What is a Function?

- Why we use functions
- Benefits of functions

How to Define a Function

- `def` keyword
- Function name, parentheses, colon
- Indentation for function body

Calling a Function

- How to run (invoke) a function

Parameters and Arguments

- What goes inside the parentheses
- Passing data to a function

Return Statement

- How to send a result back

Built-in Functions

- Examples: `print()`, `len()`, `type()`

User-defined Functions

- Creating your own

Summary By the end, you'll know how to write, call, and use functions in Python!

Agenda — Functions in Python

Functions help you organize your code into **reusable blocks**.
They make your code cleaner, reduce repetition, and are easy to debug.

Let's learn how to create and use functions in Python!

What is a Function?

A **function** is a block of code that runs **only when you call it**.

- It can take **inputs** (called **parameters**)
- It can **return an output** (using `return`)

Why use functions?

- Reuse code
- Make code readable
- Split big problems into small parts

How to Define a Function

Steps :

- `def` keyword tells Python you are creating a function.
- `function_name` can be any valid name you choose.
- `()` is where you put parameters (data you pass in). If there are none, you still need the parentheses.
- `:` means the start of the function body.
- The code block inside must be indented (usually 4 spaces or a tab).

Syntax:

```
def function_name(parameters):
    # code block
```

Example:

In [11…
```
def greet():
    print("Hello! Welcome to Python functions.")
```

Calling a Function

To **run** the function, write its name **with parentheses**.

Example:

In [5]:
```
# Define the function
def greet():
    print("Hello! Welcome to Python functions.")

# Call the function
greet()
```

```
Hello! Welcome to Python functions.
```

Parameters and Arguments

- **Parameter:** Variable inside the function definition.
- **Argument:** Value you pass to the function when calling it.

Example:

In [12…
```
def greet(name):
    print("Hello,", name)

greet("Isha")
greet("Trisha")
```

```
Hello, Isha
Hello, Trisha
```

Return Statement

A function can **return a value** using `return`.

Example:

In [13…
```
def add(a, b):
```

```
    return a + b

result = add(3, 5)
print("Sum:", result)
```

```
Sum: 8
```

Built-in Functions

Python has many functions ready to use:

- `print()` → Display output
- `len()` → Length of a string, list, etc.
- `type()` → Check data type

Examples:

In [15…
```
text = "Python"

print(len(text))   # 6
```

```
6
```

In [14…
```
print(type(text))  # <class 'str'>
```

```
<class 'str'>
```

Another User-defined Function

Let's create a function to check if a number is **even**.

Example:

In [16…
```
def is_even(number):
    if number % 2 == 0:
        return True
    else:
        return False

print(is_even(4))   # True
print(is_even(7))   # False
```

```
True
False
```

Summary

- Functions help you reuse code.
- Define a function with `def`.
- Call a function using its name + `()`.
- Pass data with **parameters/arguments**.
- Use `return` to send a value back.

Keep practicing — functions make your programs powerful, clear, and reusable!

Connect @
Mail (Trisha Dhiman): dhimantrisha1812@gmail.com
Contact: +91-9729832340

Trisha Dhiman

AI/ML Enthusiast | Intern @CodroidHub | First-Year Engineering Student