Python TUTORIAL

Python Functions

Summary: in this tutorial, you'll learn to develop Python functions by using the `def` keyword.

What is a function

A function is a named code block that performs a job or returns a value.

Why do you need functions in Python

Sometimes, you need to perform a task multiple times in a program. And you don't want to copy the code for that same task all over places.

To do so, you wrap the code in a function and use this function to perform the task whenever you need it.

For example, whenever you want to display a value on the screen, you need to call the `print()` function. Behind the scene, Python runs the code inside the `print()` function to display a value on the screen.

In practice, you use functions to divide a large program into smaller and more manageable parts. The functions will make your program easier to develop, read, test, and maintain.

The `print()` function is one of many built-in functions in Python. It means that these functions are available everywhere in the program.

In this tutorial, you'll learn how to define user-defined Python functions.

Defining a Python function

Here's a simple function that shows a greeting:

```
def greet():
    """ Display a greeting to users """
    print('Hi')
```

This example shows the simplest structure of a function. A function has two main parts: a function definition and body.

1) Function definition

A function definition starts with the `def` keyword and the name of the function (`greet`).

If the function needs some information to do its job, you need to specify it inside the parentheses `()` . The `greet` function in this example doesn't need any information, so its parentheses are empty.

The function definition always ends in a colon (`:`).

2) Function body

All the indented lines that follow the function definition make up the function's body.

The text string surrounded by triple quotes is called a docstring (https://www.pythontutorial.net/python-basics/python-function-docstrings/) . It describes what the function does. Python uses the docstring to generate documentation for the function automatically.

The line `print('Hi')` is the only line of actual code in the function body. The `greet()` function does one task: `print('Hi')` .

Calling a function

When you want to use a function, you need to call it. A function call instructs Python to execute the code inside the function.

To call a function, you write the function's name, followed by the information that the function needs in parentheses.

The following example calls the `greet()` function. Since the `greet()` function doesn't need any information, you need to specify empty parentheses like this:

```
greet()
```

If you run the program, it'll show a greeting on the screen:

```
Hi
```

Passing information to Python functions

Suppose that you want to greet users by their names. To do it, you need to specify a name in parentheses of the function definition as follows:

```
def greet(name):
```

The `name` is called a function parameter or simply parameter.

When you add a parameter to the function definition, you can use it as a variable inside the function body:

```
def greet(name):
    print(f"Hi {name}")
```

And you can access the `name` parameter only within the body of the `greet()` function, not the outside.

When you call a function with a parameter, you need to pass the information. For example:

```
greet('John')
```

Output:

```
Hi John
```

The value that you pass into a function is called an argument. In this example `'John'` is an argument.

Also, you can call the function by passing a variable into it:

```
first_name = 'Jane'
greet(first_name)
```

In this example, the `first_name` variable is also the argument of the `greet()` function.

Parameters vs. Arguments

It's important to distinguish between the parameters and arguments of a function.

A parameter of a piece of information that a function needs. And you specify the parameter in the function definition. For example, the `greet()` function has a parameter called `name` .

An argument is a piece of data that you pass into the function. For example, the text string `'John'` or the variable `jane` is the function argument.

Sometimes, you'll see that the term parameters and arguments are used interchangeably.

Returning a value

A function can perform a task like the `greet()` function. Or it can return a value. The value that a function returns is called a **return value**.

To return a value from a function, you use the `return` statement inside the function body.

```
return value
```

The following example modifies the `greet()` function to return a greeting instead of displaying it on the screen:

```
def greet(name):
    return f"Hi {name}"
```

When you call the `greet()` function, you can assign its return value to a variable:

```
greeting = greet('John')
```

And show it on the screen:

```
print(greeting)
```

The new `greet()` function is better than the old one because it doesn't depend on the `print()` function.

Later, you can reuse the `greet()` function in other applications. For example, you can use it in a web application to greet users after they log in.

Python functions with multiple parameters

A function can have zero, one, or multiple parameters.

The following example defines a function called `sum()` that calculates the sum of two numbers:

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

```
total = sum(10,20)
print(total)
```

Output:

```
30
```

In this example, the `sum()` function has two parameters `a` and `b`, and returns the sum of them.

When a function has multiple parameters, you need to use a comma to separate them.

When you call the function, you need to pass all the arguments. If you pass more or fewer arguments to the function, you'll get an error.

In the following function call, a will be 10 and b will be 20 inside the function body.

```
total = sum(10, 20)
```

Summary

- A Python function is a reusable named block of code that performs a task or returns a value.
- Use the `def` keyword to define a new function. A function consists of function definition and body.
- A function can have zero or more parameters. If a function has one or more parameters, you need to pass the same number of arguments into it.
- A function can perform a job or return a value. Use the `return` statement to return a value from a function.