Python_Functions

January 15, 2025

1 Python Functions:

1.1 Presented by Abhisek Sarkar

1.1.1 as20ms091@iiserkol.ac.in

Function: A block of code that performs a specific task.

1.2 What is Python Function?

Python function is a block of code that can be executed multiple times from different parts of the program. * only runs when it is called.

- Data can be passed, known as parameters, into a function.
- A function can return data as a result.

Creating a Function In Python a function is defined using the `def` keyword:

[4]:
```
def function():
  print("Hello World!")
```

Calling a Function To call a function, use the function name followed by parenthesis:

[5]:
```
def function():
  print("Hello World!")

function()
```

```
Hello World!
```

Arguments In computing, arguments are values that are passed to a function when it is called. These values are used within the function to perform specific operations or calculations.

- **Argument:** The input value that the function receives.
- **Parameter:** A variable that represents an argument within the function's definition.

Key Points:

- Arguments allow functions to be more versatile and reusable.
- You can pass multiple arguments to a function by separating them with commas.
- The order of arguments matters when calling a function.

Example:

```
def greet(name):
  """This function greets the person passed in as a parameter."""
  print("Hello, " + name + ". Good morning!")

greet('Abhisek')  # Output: Hello, Abhisek. Good morning!
```

In this example, `name` is the parameter, and `'Abhisek'` is the argument. The `greet` function uses the argument to personalize the greeting.

In short: From a function's perspective:

- A **parameter** is the variable listed inside the parentheses in the function definition.
- An **argument** is the value that is sent to the function when it is called.

Number of Arguments: When you define a function, you usually specify how many input values (arguments) it needs to work correctly. To use that function, you must provide exactly that number of arguments.

For example:

If a function is designed to take two pieces of information, you must give it two values when you "call" it. You can't use it with only one value or with three values.

This ensures that the function receives the necessary information to perform its intended task accurately.

[6]:
```
def my_function(first_name, last_name):
  print(first_name + " " + last_name)

my_function("Abhisek", "Sarkar")
```

```
Abhisek Sarkar
```

- This function expects 2 arguments, and gets 2 arguments.
- If the function is called with 1 or 3 arguments, there will be error:

If we pass 3 arguments?

```
def my_function(first_name, last_name):
  print(first_name + " " + last_name)

my_function("Michael", "Joseph", "Jackson")
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[7], line 4
      1 def my_function(first_name, last_name):
      2   print(first_name + " " + last_name)
----> 4 my_function("Michael", "Joseph", "Jackson")
```

```
TypeError: my_function() takes 2 positional arguments but 3 were given
```

1.2.1 Arbitrary Arguments, *args

What if we want to pass a variable number of arguments to a function? We can use the `*args` If the number of arguments is **unknown**, add a `*` before the parameter name.

[8]:
```
def goal_scorer(*players):
  print("The highest goal scorer in football is " + players[2])

goal_scorer("Maradona", "Messi", "Ronaldo", "Anthony", "Nicolas Jackson")
```

```
The highest goal scorer in football is Ronaldo
```