Higher Order Functions in Python

geeksforgeeks.org/higher-order-functions-in-python

January 20, 2020

A function is called **Higher Order Function** if it contains other functions as a parameter or returns a function as an output i.e, the functions that operate with another function are known as Higher order Functions. It is worth knowing that this higher order function is applicable for functions and methods as well that takes functions as a parameter or returns a function as a result. Python too supports the concepts of higher order functions.

Properties of higher-order functions:

- A function is an instance of the Object type.
- You can store the function in a variable.
- You can pass the function as a parameter to another function.
- You can return the function from a function.
- You can store them in data structures such as hash tables, lists, ...

Functions as objects

In Python, a function can be assigned to a variable. This assignment does not call the function, instead a reference to that function is created. Consider the below example, for better understanding.

Example:

filter_none
edit

play_arrow

brightness_4

```
def shout(text):
return text.upper()

print (shout( 'Hello' ))

yell = shout

print (yell( 'Hello' ))
```

Output:

```
HELLO
HELLO
```

In the above example, a function object referenced by shout and creates a second name pointing to it, yell.

Passing Function as an argument to other function

Functions are like objects in Python, therefore, they can be passed as argument to other functions. Consider the below example, where we have created a function greet which takes a function as an argument.

Example:

filter_none
edit

play_arrow

brightness_4

```
def shout(text):
return text.upper()

def whisper(text):
return text.lower()

def greet(func):
greeting = func("Hi, I am created by a function \
passed as an argument.")
print (greeting)

greet(shout)
greet(whisper)
```

Output:

```
HI, I AM CREATED BY A FUNCTION PASSED AS AN ARGUMENT.
hi, i am created by a function passed as an argument.
```

Returning function

As functions are objects, we can also return a function from another function. In the below example, the create_adder function returns adder function.

Example:

filter_none
edit

play_arrow

brightness_4

```
def create_adder(x):
def adder(y):
return x + y

return adder

add_15 = create_adder( 15 )

print (add_15( 10 ))
```

Output:

25

Decorators

Decorators are the most common use of higher-order functions in Python. It allows programmers to modify the behavior of function or class. Decorators allow us to wrap another function in order to extend the behavior of wrapped function, without permanently modifying it. In Decorators, functions are taken as the argument into another function and then called inside the wrapper function.

Syntax:

```
@gfg_decorator
def hello_decorator():
    .
    .
    .
```

The above code is equivalent to –

```
def hello_decorator():
    .
    .
    .

hello_decorator = gfg_decorator(hello_decorator)
```

In the above code, `gfg_decorator` is a callable function, will add some code on the top of some another callable function, `hello_decorator` function and return the wrapper function.

Example:

filter_none
edit

play_arrow

brightness_4

```
def hello_decorator(func):

def inner1():
print ( "Hello, this is before function execution" )

func()

print ( "This is after function execution" )

return inner1

def function to be used():
print ( "This is inside the function !!" )

function_to_be_used = hello_decorator(function_to_be_used)

function_to_be_used()
```

Output:

Hello, this is before function execution
This is inside the function !!
This is after function execution

Note: For more information, refer to Decorators in Python.

---

nikhilaggarwal3
Check out this Author's contributed articles.

If you like GeeksforGeeks and would like to contribute, you can also write an article using contribute.geeksforgeeks.org or mail your article to contribute@geeksforgeeks.org. See your article appearing on the GeeksforGeeks main page and help other Geeks.

Please Improve this article if you find anything incorrect by clicking on the "Improve Article" button below.