Decorators in Python

decorator is a design pattern that adds additional responsibilities to an object dynamically. In Python, a function is the first-order object. So, a decorator in Python adds additional responsibilities/functionalities to a function dynamically without modifying a function.

In Python, a function can be passed as an argument to another function. It is also possible to define a function inside another function, and a function can return another function.

a decorator in Python is a function that receives another function as an argument. The behavior of the argument function is extended by the decorator without actually modifying it. The decorator function can be applied over a function using the @decorator syntax.

```
def mydecorator(fn):
    fn()
    print('How are you?')



def greet():
    print('Hello! ', end='')

mydecorator(greet)
```

```
def mydecorator(fn):
    def inner_function():
        fn()
        print('How are you?')
    return inner_function


@mydecorator
def greet():
    print('Hello! ', end='')

greet()
```