Funções decoradoras e decoradores

terça-feira, 13 de setembro de 2022 20:10

É possivel colocar uma função em uma variável para a variável ser executada igual a função.

Exemplo:

```
def mensagem(msg):
    print(msg)

variavel = mensagem
variavel("Geronimo")
```

Podemos também criar uma função dentro da outra para jogar dentro de uma variável esse função retornada.

```
def func():
    def mensagem():
        print('alo')
    return mensagem

variavel = func()
variavel()
```

Podemos também usar uma função como parâmetro e jogar dentro da função escrava.

Exemplo:

```
def func(funcao):
    def escravo():
        funcao()
    return escravo

def fala():
    print("oi denovo")

variavel = func(fala)
variavel()
```

então eu posso usar a função fala como variávell para que seja decorado o valor da função.

Exemplo:

```
def func(funcao):
    def escravo():
        print('Agora eu estou decorada')
        funcao()
    return escravo

def fala():
    print("oi denovo")

fala = func(fala)
fala()
```

então ao inves de decorar usando uma variavel com o mesmo nome da função
que queremos decorar podemos usar um @função
Exemplo:

```
def func(funcao):
    def escravo():
        print('Agora eu estou decorada')
        funcao()
    return escravo
@func
def fala():
    print("oi denovo")

fala()
```

então agora a função fala() está decorada na func

logo se queremos passar mais de um função para decorar e nelas estiverem
argumentos que na função escrava do decorador não tenha usamos o *args e
o **kwargs

Exemplo:

```
def func(funcao):
    def escravo(*args,**kwargs):
        print('Agora eu estou decorada')
        funcao(*args,**kwargs)
    return escravo
@func
def fala():
    print("oi denovo")

@func
def valor(msg):
    print(msg)

fala()
valor(1)
```

Logo a função decoradora pode receber outra função com ou sem parâmetro.

Função de velocidade:

```python
from time import time
from time import sleep


def velocidade(funcao):
    def interna(*args, **kwargs):
        start_time = time()
        resultado = funcao(*args, **kwargs)
        end_time = time()
        tempo = (end_time - start_time) * 1000
        print(f'\nA função {funcao.__name__} '
              f'levou {tempo:.2f}ms para executar.')
        return resultado
    return interna
```

NÂO SE DEVE PASSAR OBJETOS MUTAVEIS COMO PARÂMETROS PORÉM SE PASSADO DEVEMOS USAR DESSE JEITO:

```python
def lista_de_clientes(clientes_iteravel, lista=None):
    if lista is None:
        lista = []
    lista.extend(clientes_iteravel)
    return lista


lista_clientes_1 = ['Fabrício']
clientes1 = lista_de_clientes(['João', 'Maria', 'Eduardo'], lista_clientes_1)
clientes2 = lista_de_clientes(['Marcos', 'Jonas', 'Zico'])
clientes3 = lista_de_clientes(['José'])

print(clientes1)
print(clientes2)
```