Decoradores en Python

Los decoradores son funciones que reciben como argumento otras funciones y dan como resultado otras funciones, se usan para modificar el comportamiento de otras funciones y que estas puedan ser reutilizadas, además de que funcionan para acortar el código.

Se usan también para añadir funciones a otras funciones, como un ejemplo se tiene un sofá en el que se le van añadiendo cosas para que se vea mejor o para que se pueda hacer algo más que solo sentarse en él.

Los decoradores se basan en tres fundamentos principales:

Funciones dentro de otras funciones:

Podemos definir una función2 dentro de otra función y llamarla desde la función principal, pero no podremos llamar a esta función2 desde otro lado ya que está dentro de la función principal.

Retornado de funciones:

Usando este método vamos a poder usar la función2 dentro de la otra desde otras partes del código usando una variable.

Funciones como argumentos:

Al usar una función podemos definir una variable como un argumento, también podemos asignar una función 2 como argumento de una función principal, así nos ahorramos el paso de definir una variable más para ejecutar la función 2.

Estructura de un decorador:

Un decorador está compuesto de las tres ideas anteriores en donde debemos definir una función como argumento de otra, definir una función dentro de la principal y retornar a esta misma, además de que en la función interior ejecuta la función definida como argumento.

```python
def decorador(funcion):
    def nueva_funcion(x,y):
        print("es par?")
        c = funcion(x,y)
        print("si, es par")
        return c
    return nueva_funcion
```

Esta es una forma general en la que podemos encontrar un decorador, podemos ver que al definir una función principal damos como un argumento una función, después definimos una función adicional donde escribimos las nuevas funciones que queremos ver llamando a la función del argumento y por ultimo retornando a la función interna manteniendo el alineado correspondiente; para insertarlo en una función debemos colocar un @ con el nombre del decorador sobre la función a decorar

```python
@decorador
def suma(x,y):
    print(2+4)
    return x + y

suma(2,4)
```

De esta forma al ejecutar el código podemos presenciar las funciones añadidas.

DECORADO CON FUNCIONES DE PARAMETROS:

Es muy común encontrar códigos que necesiten parámetros para realizar operaciones y el decorado básico no permite que se guarden estos parámetros por lo que se debe añadir el concepto de *args **kwargs el cual permite que se guarden los parámetros variables.