DECORADORES

En programación, un decorador es una técnica o patrón que permite cambiar o agregar funcionalidad a una función o método existente a una función o método sin cambiar su código fuente. Los decoradores son útiles para ampliar la funcionalidad de las funciones de forma modular y reutilizable. Decorador se refiere específicamente a una función que acepta otra función como entrada y devuelve una nueva función de salida que imita o modifica el comportamiento de la función original.

Los beneficios de los decoradores son:

1. **La modularidad:**
   Esto nos permite separar las preocupaciones y organizar el código de una manera más limpia y estructurada. Para incorporar la funcionalidad adicional directamente en el código fuente de la función, puede aplicar decoradores a las funciones.

2. **La reutilización:**
   Una vez definido un decorador, se puede usar varias veces sin tener la necesidad de duplicar el código, lo que facilita agregar funciones relacionadas en varios lugares del programa. puede usarlo para varias funciones sin tener que escribir código duplicado

3. **Mas comprensible:** Los decoradores hacen el código más comprensible, porque al eliminar esa necesidad de agregar más cuerpo a la función lo hace un código mas centrado en la tarea principal de la función

El uso de decoradores es común en Python en una variedad de contextos, incluida la ejecución medición de tiempo, la validación de datos y registro de eventos

```python
# Definir un decorador que agrega un mensaje antes y después de la función
def mensaje_decorador(funcion):
    def encapsular():
        print("Antes de ejecutar la función")
        funcion()
        print("Después de ejecutar la función")
    return encapsular

# Aplicar el decorador a una función
@mensaje_decorador
def saludar():
    print("Hola, mundo!")

# Llamar a la función decorada
saludar()
```

```
Antes de ejecutar la función
Hola, mundo!
Después de ejecutar la función
```

Jonathan David Fraga Narváez