ACTIVIDAD 1

LAMBDA, DECORADORES Y GENERADORES

MARCO ANTONIO VARGAS LIRA

Este programa en Python tiene como propósito **leer, filtrar, procesar y analizar temperaturas** registradas en distintas ciudades mexicanas. Además, incorpora una herramienta de auditoría que **mide el tiempo que tarda en ejecutarse la función principal** encargada del procesamiento de datos.

De manera general, el programa:

1. Genera un conjunto de datos con temperaturas de distintas ciudades.

2. Filtra solo aquellas que tienen temperaturas altas (mayores o iguales a 30 °C).

3. Ordena esas temperaturas de mayor a menor.

4. Genera mensajes de alerta para cada ciudad con calor extremo.

5. Calcula la temperatura promedio de esas alertas.

6. Finalmente, muestra los resultados y el tiempo total de ejecución.

El diseño del programa demuestra el uso de **decoradores, generadores, funciones lambda y funciones de orden superior**, que son herramientas clave de la programación funcional en Python.

La información del programa se obtiene a través de una función llamada `leer_temperaturas`.
Esta función no devuelve una lista completa, sino que **usa la palabra clave `yield`**, lo cual la convierte en un **generador**.

Un generador en Python entrega los datos uno a uno, en lugar de cargar todos en la memoria de golpe.
Esto resulta más eficiente cuando se trabaja con grandes cantidades de información.

El generador contiene una lista con siete ciudades y sus respectivas temperaturas.
Por ejemplo:

- Ciudad de México con 26 °C
- Monterrey con 34 °C
- Mérida con 40 °C
  y así sucesivamente.

Cada vez que el programa solicita un nuevo dato, el generador devuelve una tupla (por ejemplo, `("Monterrey", 34)`), hasta recorrer toda la lista.

```
from functools import reduce

import time
```

```python
# Decorador

def auditar_funcion(func):

    def envoltura(*args, **kwargs):

        print("\nEjecutando:", func.__name__)

        inicio = time.time()

        resultado = func(*args, **kwargs)

        fin = time.time()

        print("Duración:", round(fin - inicio, 5), "segundos")

        return resultado

    return envoltura



# Generador de datos

def leer_temperaturas():

    datos = [

        ("CDMX", 26),

        ("Monterrey", 34),

        ("Toluca", 19),

        ("Cancún", 38),

        ("Guadalajara", 31),

        ("Puebla", 28),

        ("Mérida", 40)

    ]

    for ciudad in datos:

        yield ciudad
```

```python
@auditar_funcion
def procesar_datos():
    # Convertir el generador en lista
    temperaturas = list(leer_temperaturas())

    # Filtrar temperaturas altas (>=30)
    altas = list(filter(lambda x: x[1] >= 30, temperaturas))

    # Ordenar de mayor a menor temperatura
    ordenadas = sorted(altas, key=lambda x: x[1], reverse=True)

    # Transformar en mensajes de alerta
    alertas = list(map(lambda x: f"Alerta de calor en {x[0]}: {x[1]}°C", ordenadas))

    # Calcular promedio de temperaturas
    if len(ordenadas) > 0:
        promedio = reduce(lambda acc, x: acc + x[1], ordenadas, 0) / len(ordenadas)
    else:
        promedio = 0

    # Mostrar resultados
    print("\nAlertas ordenadas:")
    for alerta in alertas:
        print(alerta)
```

```
    print(f"\nTemperatura promedio de alertas: {promedio:.1f}°C")


# Programa principal

if __name__ == "__main__":

    procesar_datos()
```

https://github.com/themarco11/Python-INTERMEDIO.git