Preguntas de reto:

1. Escribir un ejemplo de una lambda function de Python

# Función lambda que calcula el cuadrado de un número

square = lambda x: x \*\* 2

# Utilizando la función lambda para calcular el cuadrado de un número

print(square(5)) # Output: 25

Otro ejemplo:

# Función lambda que suma dos números

addition = lambda a, b: a + b

# Utilizando la función lambda para sumar dos números

result = addition(3, 5)

print(result) # Output: 8

1. Un ejemplo y utilización de lookup table

# Definir la tabla de búsqueda para convertir grados Celsius a Fahrenheit

Una tabla de búsqueda, o "lookup table" en inglés, es una estructura de datos que se utiliza para asociar claves con valores. Es comúnmente utilizada en programación para almacenar información que necesita ser accedida rápidamente mediante una clave.

tabla\_conversion = {

-20: -4,

-15: 5,

-10: 14,

-5: 23,

0: 32,

5: 41,

10: 50,

15: 59,

20: 68,

25: 77,

30: 86,

35: 95,

40: 104

}

# Función para buscar el valor Fahrenheit correspondiente a una temperatura en Celsius

def celsius\_a\_fahrenheit(celsius):

if celsius in tabla\_conversion:

return tabla\_conversion[celsius]

else:

return "Temperatura no encontrada en la tabla de conversión"

# Ejemplo de uso de la función

temperatura\_celsius = 20

temperatura\_fahrenheit = celsius\_a\_fahrenheit(temperatura\_celsius)

print(f"{temperatura\_celsius} grados Celsius son equivalentes a {temperatura\_fahrenheit} grados Fahrenheit.")

1. Retornar un n-valor de la secuencia fibonacci con recursividad

def fibonacci(n):

if n <= 1:

return n

else:

return fibonacci(n-1) + fibonacci(n-2)

# Solicitar al usuario el valor de n

n = int(input("Ingresa el valor de n para la secuencia Fibonacci: "))

# Imprimir el n-ésimo valor de la secuencia Fibonacci

print("El", n, "ésimo valor de la secuencia Fibonacci es:", fibonacci(n))

1. Un ejemplo de un quick sort

def quicksort(arr):

if len(arr) <= 1:

return arr

else:

pivot = arr[0]

less\_than\_pivot = [x for x in arr[1:] if x <= pivot]

greater\_than\_pivot = [x for x in arr[1:] if x > pivot]

return quicksort(less\_than\_pivot) + [pivot] + quicksort(greater\_than\_pivot)

# Ejemplo de uso:

arr = [3, 6, 8, 10, 1, 2, 1]

sorted\_arr = quicksort(arr)

print("Arreglo original:", arr)

print("Arreglo ordenado:", sorted\_arr)