**Delegados, funciones lambda, métodos anónimos y funciones de orden superior**

**Ejersicio 1**

Dadas dos listas de números del mismo tamaño, filtra los pares de números donde el primer número es mayor que el segundo. Luego, suma cada par de números filtrados y finalmente reduce la lista de sumas tomando el máximo valor.

**Entender y desarrollar**

* Importación de la función reduce desde functools:

Importamos la función reduce del módulo **functools**. reduce se utiliza para aplicar una función a los elementos de una lista secuencialmente y devolver un solo valor.

* Definición de la función leer lista:

leer\_lista(descripcion): Esta función recibe una descripción como argumento que se utiliza para indicar al usuario qué tipo de lista debe ingresar.

**while True::** Inicia un bucle infinito que continuará hasta que se ejecute una instrucción return que devuelva una lista válida.

try:: Comienza un bloque de código que puede producir una excepción.

lista = list(map(int, input(f"Introduce los números de la {descripcion} separados por espacios: ").split())): **Solicita al usuario que ingrese números separados por espacios, luego divide la entrada en una lista de cadenas y la convierte en una lista de enteros utilizando map**. Esto se hace dentro de un bloque try para manejar la posibilidad de que el usuario ingrese valores no enteros.

except ValueError:: **Captura la excepción ValueError que ocurre si el usuario ingresa valores que no pueden convertirse en enteros**

print("Error: Introduce solo números enteros."): Imprime un mensaje de error si se produce una excepción.

return lista: Devuelve la lista válida ingresada por el usuario.

* Lectura de las listas desde el teclado:

Se utiliza la función leer\_lista dos veces para leer las dos listas de números enteros ingresadas por el usuario.

* Verificación de que las listas tienen el mismo tamaño:

if len(list1) != len(list2): Verifica si las dos listas tienen la misma longitud.

print("Error: Las listas deben tener el mismo tamaño."): Imprime un mensaje de error si las listas no tienen la misma longitud.

* Creación de todas las combinaciones posibles de pares:

TodosPares = [(x + y, x, y) for x in list1 for y in list2]: **Crea una lista de todas las combinaciones posibles de pares (x, y) donde x es un elemento de list1 e y es el elemento correspondiente en list2. Cada par incluye la suma de x e y, x y y mismos**.

* Filtrado de pares donde x es mayor o igual que y:

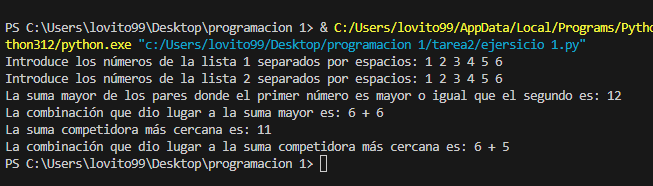
ValordeSumas = [(ValorSumado, x, y) for ValorSumado, x, y in TodosPares if x >= y]: Filtra los pares de la lista TodosPares para incluir solo aquellos donde el elemento x es mayor o igual que el elemento y. Cada par se almacena junto con su suma en una nueva lista llamada ValordeSumas.

* Ordenamiento de la lista de sumas en orden descendente:

sorted\_ValordeSumas = sorted(ValordeSumas, reverse=True): Ordena la lista ValordeSumas en orden descendente según la suma de los pares.

* Encontrar la suma mayor y la suma competidora más cercana:

Se busca la suma mayor y, si existe, la suma competidora más cercana en la lista ordenada **sorted\_ValordeSumas.** Si la suma competidora más cercana existe, se almacenan tanto la suma como los elementos x e y correspondientes.



* **Implementación del código**

from functools import reduce

*# Función para leer una lista de números enteros desde el teclado*

def leer\_lista(descripcion):

while True:

try:

lista = list(map(int, input(f"Introduce los números de la {descripcion} separados por espacios: ").split()))

return lista

except ValueError:

print("Error: Introduce solo números enteros.")

*# Leer las dos listas desde el teclado*

list1 = leer\_lista("lista 1")

list2 = leer\_lista("lista 2")

*#X=elementos de linsta 1*

*#Y=elementos de las lista 2*

*# Verificar que las listas tienen el mismo tamaño ( de elementos)*

if len(list1) != len(list2):

print("Error: Las listas deben tener el mismo tamaño.")

else:

*# Paso 1: Crear todas las combinaciones posibles de pares (x, y) donde x es un elemento de list1 e y es el elemento correspondiente en list2*

TodosPares = [(x + y, x, y) for x in list1 for y in list2]

*# Paso 2: Filtrar pares donde x es mayor o igual que y*

ValordeSumas = [(ValorSumado, x, y) for ValorSumado, x, y in TodosPares if x >= y]

*# Paso 3: Encontrar el máximo valor de la lista de sumas*

*# sorted (ordenar la listas segun condicion de cada uno)*

if ValordeSumas:

sorted\_ValordeSumas = sorted(ValordeSumas, reverse=True)

SumaMaxima, maximoX, maximoY = sorted\_ValordeSumas[0]

*# Encontrar la suma competidora más cercana*

SumaPenultima = None

for ValorSumado, x, y in sorted\_ValordeSumas[1:]:

if ValorSumado < SumaMaxima:

SumaPenultima = ValorSumado

break

print("La suma mayor de los pares donde el primer número es mayor o igual que el segundo es:", SumaMaxima)

print("La combinación que dio lugar a la suma mayor es:", maximoX, "+", maximoY)

if SumaPenultima is not None:

print("La suma competidora más cercana es:", SumaPenultima)

for ValorSumado, x, y in sorted\_ValordeSumas[1:]:

if ValorSumado == SumaPenultima:

print("La combinación que dio lugar a la suma competidora más cercana es:", x, "+", y)

break

else:

print("No hay suma competidora más cercana.")

else:

print("No hay pares donde el primer número sea mayor o igual que el segundo.")