**DESARROLLO DE LA PRACTICA**

**A). Sistema de manejo de archivos:**

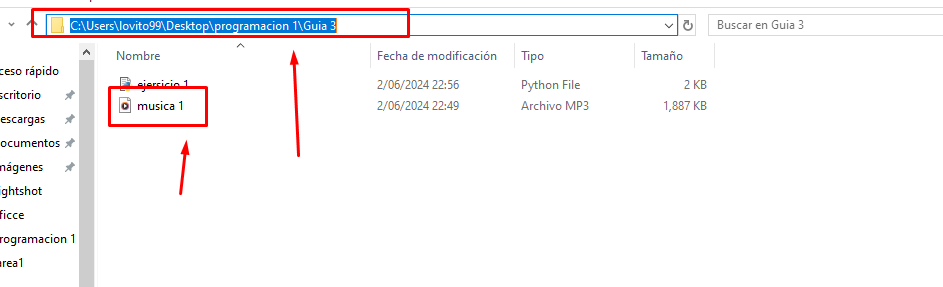
Instrucciones

Implementar un módulo para listar archivos de una

carpeta de una extensión dada. Ejemplo, listar todos los archivos

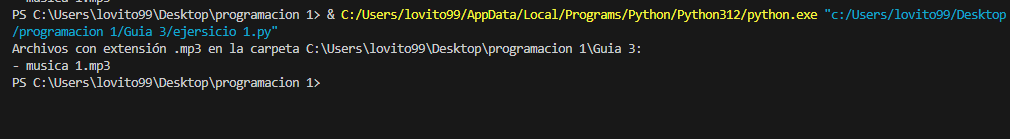
.mp3. (usar expresiones lambda para el filtrado)

* **Crear un módulo que liste archivos en una carpeta con una extensión específica.**



* **Utilizar expresiones lambda para el filtrado**

archivos = listadeArchivos(ruta\_carpeta, extension)

* **Resultado**
* **Código de implementación**

**from** **pathlib** **import** Path

**def** **ListadeArchivos**(RutaCarpeta, extension):

"""

Lista los archivos con una extensión específica en la carpeta dada.

Args:

RutaCarpeta (str): Ruta de la carpeta donde buscar los archivos.

extension (str): Extensión de los archivos a listar (por ejemplo, ".mp3").

Returns:

list: Lista de archivos con la extensión especificada.

"""

# Obtener todos los archivos en la carpeta dada

archivos = [archivo **for** archivo **in** Path(RutaCarpeta).iterdir() **if** archivo.is\_file()]

# Filtrar archivos por la extensión dada usando una expresión lambda

ArchivosFiltrados = list(filter(**lambda** archivo: archivo.suffix == extension, archivos))

**return** ArchivosFiltrados

# Ejemplo de uso

RutaCarpeta = Path(r"C:\Users\lovito99\Desktop\programacion 1\Guia 3")

extension = ".mp3" # Cambia esta extensión según sea necesario

archivos = ListadeArchivos(RutaCarpeta, extension)

**print**(f"Archivos con extensión {extension} en la carpeta {RutaCarpeta}:")

**for** archivo **in** archivos:

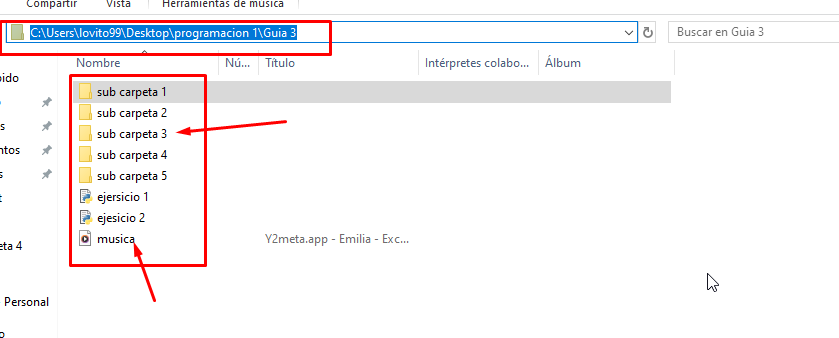
**print**("- " + archivo.name)

**Ejercicio 2:**

**Extender el ejercicio anterior, realizando una búsqueda recursiva en todo el contenido de la carpeta (incluyendo las subcarpetas).**

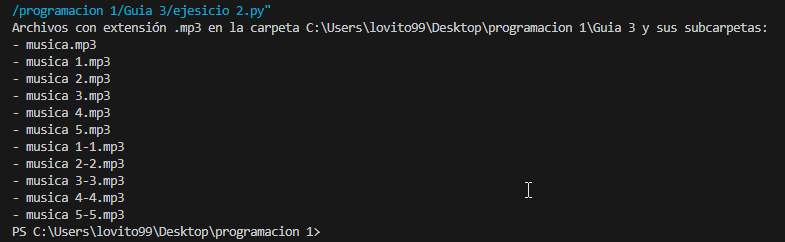
* **Path(RutaCarpeta).rglob(f"\*{extension}"):**

Este método busca de manera recursiva todos los archivos que coinciden con el patrón dado (en este caso, archivos con la extensión especificada) dentro de la carpeta y sus subcarpetas.



* **Filtrado de archivos:**

La lista de archivos se filtra para asegurarse de que solo se incluyan archivos (y no directorios).



* **Implementación del código**

**from** **pathlib** **import** Path

**def** **ListadeArchivosRecursivo**(RutaCarpeta, extension):

"""

Lista los archivos con una extensión específica en la carpeta dada y sus subcarpetas.

Args:

RutaCarpeta (str): Ruta de la carpeta donde buscar los archivos.

extension (str): Extensión de los archivos a listar (por ejemplo, ".mp3").

Returns:

list: Lista de archivos con la extensión especificada.

"""

# Obtener todos los archivos en la carpeta dada y sus subcarpetas de manera recursiva

archivos = [archivo **for** archivo **in** Path(RutaCarpeta).rglob(f"\*{extension}") **if** archivo.is\_file()]

**return** archivos

# Ejemplo de uso

RutaCarpeta = Path(r"C:\Users\lovito99\Desktop\programacion 1\Guia 3")

extension = ".mp3" # Cambia esta extensión según sea necesario

archivos = ListadeArchivosRecursivo(RutaCarpeta, extension)

**print**(f"Archivos con extensión {extension} en la carpeta {RutaCarpeta} y sus subcarpetas:")

**for** archivo **in** archivos:

* **print**("- " + archivo.name)

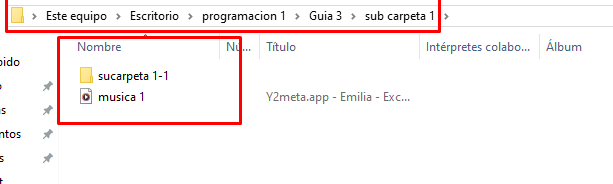
**Ejercicio 3:**

**Imprimir el contenido jerarquizado (en forma de árbol) de una carpeta como se muestra en la imagen (solo referencial): para ello usar recursividad.**

* **Función ListadeArchivo:**

Busca archivos recursivamente en una carpeta y todas sus subcarpetas.

Devuelve una lista de archivos con la extensión especificada.



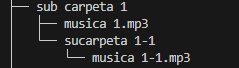
* **Función ImprimirArbol:**

Imprime el contenido de una carpeta en forma de árbol jerarquizado, filtrando por la extensión de archivo proporcionada.

RutaCarpeta: La ruta de la carpeta cuya estructura queremos imprimir.

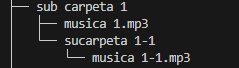
extension: Extensión de los archivos a listar.

prefijo: Prefijo utilizado para mostrar la estructura jerárquica. Comienza vacío y se actualiza en cada llamada recursiva.



Obtener y ordenar el contenido de la carpeta:

contenido = sorted(carpeta.iterdir(), key=lambda x: (x.is\_file(), x.name)) obtiene y ordena el contenido de la carpeta, listando primero directorios y luego archivos.



* **Implementación del código**



**from** **pathlib** **import** Path

**def** **ListadeArchivos**(RutaCarpeta, extension):

"""

Lista los archivos y directorios con una extensión específica en la carpeta dada y sus subcarpetas.

Args:

RutaCarpeta (str): Ruta de la carpeta donde buscar los archivos.

extension (str): Extensión de los archivos a listar (por ejemplo, ".mp3").

Returns:

list: Lista de archivos y directorios con la extensión especificada.

"""

archivos = [archivo **for** archivo **in** Path(RutaCarpeta).rglob(f"\*{extension}")]

**return** archivos

**def** **ImprimirArbol**(RutaCarpeta, extension, prefijo=""):

"""

Imprime el contenido de una carpeta en forma de árbol jerarquizado, filtrando por extensión de archivo.

Args:

RutaCarpeta (str): Ruta de la carpeta cuya estructura queremos imprimir.

extension (str): Extensión de los archivos a listar (por ejemplo, ".mp3").

prefijo (str): Prefijo utilizado para mostrar la estructura jerárquica.

"""

carpeta = Path(RutaCarpeta)

contenido = sorted(carpeta.iterdir(), key=**lambda** x: (x.is\_file(), x.name))

total\_elementos = len(contenido)

**for** indice, elemento **in** enumerate(contenido):

es\_ultimo = (indice == total\_elementos - **1**)

prefijo\_actual = prefijo + ("└── " **if** es\_ultimo **else** "├── ")

**if** elemento.is\_file() **and** elemento.suffix == extension:

**print**(prefijo\_actual + elemento.name)

**elif** elemento.is\_dir():

**print**(prefijo\_actual + elemento.name)

prefijo\_nuevo = prefijo + (" " **if** es\_ultimo **else** "│ ")

ImprimirArbol(elemento, extension, prefijo\_nuevo)

# Ejemplo de uso

RutaCarpeta = Path(r"C:\Users\lovito99\Desktop\programacion 1\Guia 3")

extension = ".mp3" # Cambia esta extensión según sea necesario

ImprimirArbol(RutaCarpeta, extension)