**PLAYLIST**

*def* agregar\_cancion(*playlist*, *cancion*, *artista*):

*playlist*.append({'cancion': *cancion*, 'artista': *artista*})

Este código define una función llamada agregar\_cancion que toma tres parámetros: playlist, canción y artista.

La función agregar\_cancion agrega una nueva canción a una lista de reproducción (playlist). Esta lista de reproducción es una lista de diccionarios, donde cada diccionario representa una canción y contiene dos pares clave-valor: 'canción' y 'artista'.

La función toma el nombre de la canción (canción) y el nombre del artista (artista) como argumentos y los agrega a la lista de reproducción (playlist) como un nuevo diccionario con las claves 'canción' y 'artista'.

*def* mostrar\_playlist(*playlist*):

    for i, cancion in enumerate(*playlist*, 1):

        print(*f*"{i}. {cancion['cancion']} - {cancion['artista']}")

Esta función, llamada mostrar\_playlist, toma una lista de reproducción (playlist) como argumento y muestra cada canción en la lista junto con su artista.

enumerate(playlist, 1): Este es un bucle for que itera sobre la lista de reproducción (playlist). La función enumerate toma dos argumentos: la lista que quieres iterar (playlist) y un número opcional que especifica el índice inicial de la enumeración (en este caso, 1). Esto significa que el primer elemento de la lista será indexado como 1, el segundo como 2, y así sucesivamente.

for i, cancion in enumerate(playlist, 1): En cada iteración, enumerate devuelve una tupla con dos elementos: el índice actual (i) y el elemento actual de la lista (cancion).

print(f"{i}. {cancion['cancion']} - {cancion['artista']}"): Para cada canción en la lista de reproducción, imprime una cadena formateada que muestra el número de índice (i), el nombre de la canción (cancion['cancion']) y el nombre del artista (cancion['artista']).

*def* guardar\_playlist(*playlist*, *archivo*):

    with open(*archivo*, 'w', *encoding*='utf-8') as file:

        file.write(" CANCIONES DE RAMIRO\n")

        file.write("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n")

        for cancion in *playlist*:

            file.write(*f*"Canción: {cancion['cancion']}\n")

            file.write(*f*"Artista: {cancion['artista']}\n")

            file.write("\n")

Esta función, llamada guardar\_playlist, toma dos parámetros: playlist, que es la lista de canciones que quieres guardar, y archivo, que es el nombre del archivo en el que deseas guardar la lista de reproducción.

**with open(archivo, 'w') as file**: : Abre el archivo especificado en modo de escritura ('w'). El uso de with asegura que el archivo se cierre correctamente después de su uso. El archivo se abre con el nombre especificado en el argumento archivo, que es donde se guardará la lista de reproducción. Se asigna a la variable file para su uso dentro del bloque with.

Se especifica **encoding='utf-8'** al abrir el archivo. Esto asegura que el archivo se abra y se escriba utilizando el conjunto de caracteres UTF-8, que es más versátil y puede manejar una amplia gama de caracteres, incluidos caracteres especiales y acentos.

for cancion in playlist:

file.write(f"Canción: {cancion['cancion']}\n")

file.write(f"Artista: {cancion['artista']}\n")

file.write("\n")

Se escribe "Canción:" y "Artista:" antes de cada uno, respectivamente. Esto ayuda a organizar la información en el archivo de una manera más legible y estructurada. Se añade una línea en blanco después de cada canción y su artista para mejorar la legibilidad del archivo.

*def* editar\_cancion(*playlist*, *index*, *nueva\_cancion*, *nuevo\_artista*):

    if not *playlist*:

        print("La playlist está vacía.")

    elif 0 < *index* <= len(*playlist*):

*playlist*[*index* - 1]['cancion'] = *nueva\_cancion*

*playlist*[*index* - 1]['artista'] = *nuevo\_artista*

        print("Canción editada correctamente.")

    else:

        print("Número de canción no encontrado en la playlist.")

Esta función, llamada editar\_cancion, permite editar una canción en una lista de reproducción. Toma cuatro parámetros: playlist, que es la lista de reproducción que quieres editar, index, que es el índice de la canción que deseas editar (el índice comienza desde 1), nueva\_cancion, que es el nuevo nombre de la canción, y nuevo\_artista, que es el nuevo nombre del artista.

if not playlist:: Comprueba si la lista de reproducción está vacía. Si lo está, imprime un mensaje indicando que la playlist está vacía y la función no hace nada más.

elif 0 < index <= len(playlist):: Comprueba si el índice proporcionado (index) está dentro del rango válido para la lista de reproducción. Si es así, procede a editar la canción en el índice especificado.

playlist[index - 1]['cancion'] = nueva\_cancion

playlist[index - 1]['artista'] = nuevo\_artista: Si el índice es válido, actualiza el nombre de la canción y el nombre del artista en la lista de reproducción utilizando el índice proporcionado (index). Se resta 1 al índice porque en Python los índices de las listas comienzan en 0, pero aquí se espera que el usuario proporcione un índice que comience en 1.

print("Canción editada correctamente."): Si la canción se edita correctamente, imprime un mensaje de confirmación.

else:: Si el índice no es válido (es decir, está fuera del rango de la lista de reproducción), imprime un mensaje indicando que el número de canción no se encontró en la playlist.

*def* eliminar\_cancion(*playlist*, *index*):

    if not *playlist*:

        print("La playlist está vacía.")

    elif 0 < *index* <= len(*playlist*):

        deleted\_song = *playlist*.pop(*index* - 1)

        print(*f*"La canción '{deleted\_song['cancion']}' ha sido eliminada de la playlist.")

    else:

        print("Número de canción no encontrado en la playlist.")

Esta función, eliminar\_cancion, se utiliza para eliminar una canción de una lista de reproducción en función de su índice en la lista.

**if not playlist:**: Verifica si la lista de reproducción está vacía. Si lo está, imprime un mensaje indicando que la lista de reproducción está vacía y no realiza ninguna acción adicional.

**elif 0 < index <= len(playlist):**: Verifica si el índice proporcionado (index) está dentro del rango válido para la lista de reproducción. Si el índice es válido, procede a eliminar la canción en ese índice.

**deleted\_song = playlist.pop(index - 1)**: Si el índice es válido, utiliza el método pop() para eliminar la canción en el índice especificado (index - 1). La canción eliminada se guarda en la variable deleted\_song.

**print(f"La canción '{deleted\_song['cancion']}' ha sido eliminada de la playlist."):** Después de eliminar la canción, se imprime un mensaje que indica cuál fue la canción eliminada.

**else::** Si el índice no es válido (es decir, está fuera del rango de la lista de reproducción), imprime un mensaje indicando que el número de canción no se encontró en la lista de reproducción.

*def* vaciar\_playlist(*playlist*):

*playlist*.clear()

    print("Playlist vaciada correctamente.")

La función vaciar\_playlist elimina todas las canciones de una lista de reproducción dada.

playlist.clear(): Este método clear() elimina todos los elementos de la lista de reproducción, dejándola vacía.

print("Playlist vaciada correctamente."): Después de vaciar la lista de reproducción, imprime un mensaje indicando que la operación se ha realizado correctamente.

*ef* buscar\_en\_playlist(*playlist*, *keyword*):

    resultados = []

    for cancion in *playlist*:

        if *keyword*.lower() in cancion['cancion'].lower() or *keyword*.lower() in cancion['artista'].lower():

            resultados.append(cancion)

    return resultados

Esta función, **buscar\_en\_playlist**, busca canciones en una lista de reproducción que coincidan con una palabra clave especificada.

**resultados = []**: Se crea una lista vacía llamada resultados para almacenar las canciones que coinciden con la palabra clave.

Se itera sobre cada canción en la lista de reproducción (playlist).

**if keyword.lower() in cancion['cancion'].lower() or keyword.lower() in cancion['artista'].lower()**: Comprueba si la palabra clave (convertida a minúsculas) está contenida en el nombre de la canción o en el nombre del artista de la canción (ambos convertidos a minúsculas). Esto se hace para que la búsqueda sea insensible a mayúsculas y minúsculas, es decir, no importa si la palabra clave tiene mayúsculas o minúsculas.

Si la palabra clave coincide con el nombre de la canción o el nombre del artista de la canción, se agrega esa canción a la lista resultados.

Al final, se devuelve la lista resultados, que contiene todas las canciones que coinciden con la palabra clave.

*def* main():

    playlist = []

    archivo = 'Archivos\Playlist de Ramiro.txt'

    while True:

        print("\n1. AGREGAR CANCION")

        print("2. MOSTRAR PLAYLIST")

        print("3. EDITAR CANCION")

        print("4. ELIMINAR CANCION")

        print("5. VACIAR PLAYLIST")

        print("6. GUARDAR PLAYLIST")

        print("7. BUSCAR REGISTRO")

        print("8. SALIR")

        opcion = input("Seleccione una opción: ")

        if opcion == '1':

            cancion = input("Ingrese el nombre de la canción: ")

            artista = input("Ingrese el nombre del artista: ")

            agregar\_cancion(playlist, cancion, artista)

        elif opcion == '2':

            print("\nPlaylist:")

            mostrar\_playlist(playlist)

        elif opcion == '3':

            index = int(input("Ingrese el número de la canción que desea editar: "))

            nueva\_cancion = input("Ingrese el nuevo nombre de la canción: ")

            nuevo\_artista = input("Ingrese el nuevo nombre del artista: ")

            editar\_cancion(playlist, index, nueva\_cancion, nuevo\_artista)

        elif opcion == '4':

            index = int(input("Ingrese el número de la canción que desea eliminar: "))

            eliminar\_cancion(playlist, index)

        elif opcion == '5':

            vaciar\_playlist(playlist)

        elif opcion == '6':

            guardar\_playlist(playlist, archivo)

            print(*f*"Playlist guardada en {archivo}")

        elif opcion == '7':

            keyword = input("Ingrese la palabra clave a buscar: ")

            resultados = buscar\_en\_playlist(playlist, keyword)

            if resultados:

                print("Resultados de la búsqueda:")

                mostrar\_playlist(resultados)

            else:

                print("No se encontraron resultados para la palabra clave ingresada.")

        elif opcion == '8':

            print("UD SALIO DEL PROGRAMA.")

            break

        else:

            print("Opción inválida. Por favor, seleccione una opción válida.")

if \_\_name\_\_ == "\_\_main\_\_":

    main()

Inicializa una lista vacía llamada playlist. Esta lista se utilizará para almacenar las canciones de la lista de reproducción.

Establece el nombre de un archivo como 'Archivos\Playlist de Ramiro.txt' y lo asigna a la variable archivo. Esto es el nombre y la ubicación del archivo donde se almacenará la lista de reproducción.

Bucle de menú: Dentro de un bucle while True, muestra un menú de opciones para que el usuario elija qué acción realizar.

Manejo de opciones: Dependiendo de la opción seleccionada por el usuario, se realizan diferentes acciones:

1 a 5: Agregar canción, mostrar playlist, editar canción, eliminar canción y vaciar playlist respectivamente. Estas opciones llaman a las funciones definidas anteriormente para realizar las operaciones correspondientes.

6: Guardar la playlist en el archivo especificado.

7: Buscar registros en la playlist basados en una palabra clave ingresada por el usuario.

8: Salir del programa.

Validación de entrada: Si el usuario ingresa una opción inválida, se muestra un mensaje de error y se le pide que elija una opción válida.

Condición de salida: Si el usuario elige la opción '8' para salir del programa, se rompe el bucle y el programa finaliza.

Invocación del programa principal: Finalmente, la condición if \_\_name\_\_ == "\_\_main\_\_": garantiza que la función main() se ejecute solo cuando se ejecute este script como el programa principal.