Proyecto Final de Curso: Gestor de Lista de Reproducción de Canciones haciendo uso de Python

Daniela Choquecondo Aspilcueta, Kristen Fernandez Cardenas, Ronald Garcia Valdivia

***Resumen*— Este documento presenta el proyecto de Trabajo de Investigación Formativa. el cuál se trata sobre un Gestor de Lista de Reproducción de Canciones. El programa utiliza una estructura de Lista, esto tras ver sus ventajas y desventajas, viendo su notación Big-O, además de una implementación de la interfaz gráfica.**

Palabras clave: Playlist, Python

I. Introducción

Este documento trata sobre el desarrollo de un programa para gestionar una lista de reproducción de canciones haciendo uso de una estructura List, además de haber sido programado con el lenguaje Python, el proyecto se llevó a cabo a lo largo de un mes aproximadamente, se basó principalmente en las estructuras ofrecidas por Khan [1]. Se mostrarán diferentes secciones, las cuales son las siguientes: Trabajos relacionados, Herramientas usadas, Metodología, Resultados, Conclusiones y Referencias.

II. Trabajos relacionados

Entre los trabajos relacionados en los que se investigó para encontrar semejantes a este código están:

Primeramente el trabajo de Khan [1] en el que se necesita una aplicación que permita reproducir archivos de audio digital, similar a un reproductor de música físico. Esta aplicación debe permitir reproducir, pausar y detener canciones, así como listar los archivos de música disponibles en el escritorio o portátil. Para ser atractiva, debe tener una interfaz de usuario simple pero bonita. Además, debería mostrar información sobre el archivo que se está reproduciendo y permitir a los usuarios crear listas de reproducción, lo cual requerirá una base de datos para almacenar esta información. Python ofrece bibliotecas como Pygame para trabajar con archivos de audio y sqlite3 para gestionar bases de datos. Y su conclusión del trabajo fue que el reproductor MP3 es un dispositivo para reproducir y escuchar archivos de audio digital, creado en lenguaje Python. La aplicación tiene una interfaz gráfica de usuario simple y fácil de usar, que ofrece cinco opciones: agregar una canción a una lista de reproducción, reproducir la canción, pausar o reanudar la canción, reproducir la canción anterior y reproducir la siguiente canción. El reproductor también puede agregar múltiples pistas a la lista de reproducción al mismo tiempo y muestra la lista en un área de visualización grande. Al seleccionar y reproducir una pista, se pueden escuchar y ver detalles de la canción, como el nombre de la canción, el nombre del cantante, la duración de la canción y el tamaño del archivo.

En el trabajo de McFee [2] se concluye que han demostrado que el rendimiento del modelo de listas de reproducción puede mejorar al tratar categorías específicas de listas de reproducción de manera individual. Aunque los modelos simples propuestos aquí funcionan bien en algunas situaciones, están lejos de ser completos y sugieren muchas direcciones para trabajos futuros. La suposición de un modelo de Markov de primer orden es claramente una simplificación, dado que los usuarios a menudo crean listas de reproducción con interacciones a largo plazo y propiedades temáticas globales. De manera similar, la distribución uniforme de las canciones dentro de un conjunto de aristas permite una implementación eficiente y escalable, pero permitir distribuciones no uniformes también podría ser una vía para mejoras futuras.

III. Herramientas usadas

A continuaciones se mostrarán todas las herramientas usadas durante el proyecto,esto incluye lenguajes de programación, programas y estructuras.

A. Python

Python es un lenguaje de programación ampliamente utilizado en las aplicaciones web, el desarrollo de software, la ciencia de datos y el machine learning (ML). Los desarrolladores utilizan Python porque es eficiente y fácil de aprender, además de que se puede ejecutar en muchas plataformas diferentes. El software Python se puede descargar gratis, se integra bien a todos los tipos de sistemas y aumenta la velocidad del desarrollo.

Entre los principales beneficios están:

Los desarrolladores pueden leer y comprender fácilmente los programas de Python debido a su sintaxis básica similar a la del inglés.

Python permite que los desarrolladores sean más productivos, ya que pueden escribir un programa de Python con menos líneas de código en comparación con muchos otros lenguajes.

Python cuenta con una gran biblioteca estándar que contiene códigos reutilizables para casi cualquier tarea. De esta manera, los desarrolladores no tienen que escribir el código desde cero.

Los desarrolladores pueden utilizar Python fácilmente con otros lenguajes de programación conocidos, como Java, C y C + +.

Se usó como una guía el trabajo de González [3].

B. Visual Studio Code

Visual Studio Code (VS Code) es un editor de código fuente desarrollado por Microsoft. Es software libre y multiplataforma, está disponible para Windows, GNU/Linux y macOS. VS Code tiene una buena integración con Git, cuenta con soporte para depuración de código, y dispone de un sinnúmero de extensiones, que básicamente te da la posibilidad de escribir y ejecutar código en cualquier lenguaje de programación.

Como un trabajo relacionado al uso de este programa, se tiene el trabajo de Amann [4].

C. Github

GitHub es un servicio basado en la nube que aloja un sistema de control de versiones (VCS) llamado Git. Éste permite a los desarrolladores colaborar y realizar cambios en proyectos compartidos, a la vez que mantienen un seguimiento detallado de su progreso. Se usó como guía el trabajo de Cosentino [5].

D. TDA Lista

La Lista es una estructura de datos muy importante en los lenguajes de programación donde:

- Representa una colección de elementos ordenados.
- Puede contener elementos repetidos.
- Cada elemento de la lista tiene un índice que lo ubica dentro de la misma.

Operaciones

En general las listas proveen las siguientes operaciones:

- Construir una lista
- Obtener el tamaño de la lista
- Verificar si está vacía
- Obtener el primer elemento de la lista, usualmente llamado head o cabeza
- Agregar un nuevo elemento a la lista.
- Obtener el elemento para un índice dado.
- Etc.

Según Méndez [6]: Si los programadores se limitaran a utilizar únicamente los tipos de datos que los lenguajes de programación proveen en forma primitiva, estos lenguajes no serían muy útiles. Por ello mediante la ayuda de un concepto central, llamado abstracción que permite a partir de la combinación de ciertas herramientas primitivas que ofrecen los lenguajes de programación tradicionales, crear nuevos tipos de datos.

IV. Metodología

Tras una búsqueda se encontraron las principales características de una estructura TDA Lista, todas estas fueron evaluadas para conocer sus beneficios y desventajas para ser usada como la estructura del proyecto:

TABLA 1

Características	TDA Lista
Definición	Colección ordenada de elementos donde cada elemento tiene un sucesor (excepto el último).
Estructura	Elementos enlazados secuencialmente, típicamente con punteros al siguiente elemento.
Operaciones básicas	• Inserción de elemento • Eliminación de elementos • Acceso a elementos • Recorrido secuencial
Complejidad de inserción	O(1) para insertar al inicio o al final (listas enlazadas); O(n) para insertar en una posición específica.
Complejidad de búsqueda	O(n) en promedio y en el peor caso (listas enlazadas); O(1) para acceso por índice (listas array).
Uso de memoria	Eficiente en listas array; más memoria en listas enlazadas debido a punteros.
Ventajas	• Sencillez de implementación. • Inserciones y eliminaciones eficientes en listas enlazadas. • Acceso rápido en listas array.
Desventajas	Búsqueda lenta en listas enlazadas. Inserción y eliminación lentas en listas array.

Tras esta rápida revisión se decidió trabajar con esta estructura, dado que también

V. Propuesta

El trabajo se desarrolló de la siguiente forma, se usó la aplicación Visual Studio Code para desarrollar el código, el lenguaje escogido fue Python por la familiaridad, La estructura es un TDALista, además se hizo uso de Github para subir los avances del trabajo.

Para el Trabajo se creó la clase"ReproductorMusica", entre los principales métodos de esta estructura se encuentran:

- __init__: Inicializa la lista de reproducción y las canciones disponibles.
- agregar_cancion: Añade una canción a la lista de reproducción.
- eliminar_cancion: Elimina una canción de la lista de reproducción usando su track_id.
- cambiar_orden: Cambia la posición de una canción en la lista de reproducción.
- reproduccion_aleatoria: Mezcla las canciones en la lista de reproducción de manera aleatoria.
- ordenar_playlist: Ordena la lista de reproducción según un atributo específico.
- obtener_canciones_disponibles: Devuelve la lista de

canciones disponibles.

- obtener_playlist: Devuelve la lista de reproducción actual.

Todos estos métodos están en función a el atributo Track_Name este fué decidido como el indicador o llave de las canciones por así decirlo, ya que un usuario final busca las canciones según el nombre de las canciones. A continuación se presentan algunas partes del código, el de las funciones principales:

```
    def agregar_cancion(self):
        seleccion = self.lista_canciones.curselection()
        if seleccion:
            cancion = self.reproductor.obtener_canciones_disponibles()[seleccion[0]]
            self.reproductor.agregar_cancion(cancion)
            self.actualizar_playlist()
            messagebox.showinfo("Éxito", f"Canción '{cancion.track_name}' añadida a la playlist.")
        else:
            messagebox.showerror("Error", "Seleccione una canción para agregar.")

    def eliminar_cancion(self):
        seleccion = self.lista_playlist.curselection()
        if seleccion:
            track_id = self.reproductor.obtener_playlist()[seleccion[0]].track_id
            self.reproductor.eliminar_cancion(track_id)
            self.actualizar_playlist()
            messagebox.showinfo("Éxito", "Canción eliminada de la playlist.")
        else:
            messagebox.showerror("Error", "Seleccione una canción para eliminar.")

    def cambiar_orden(self):
        posicion_actual = simpledialog.askinteger("Cambiar Orden", "Ingrese la posición actual de la c
        nueva_posicion = simpledialog.askinteger("Cambiar Orden", "Ingrese la nueva posición de la ca
        if posicion_actual is not None and nueva_posicion is not None:
            self.reproductor.cambiar_orden(posicion_actual-1, nueva_posicion-1)
            self.actualizar_playlist()
```

Se puede decir que el trabajo fue todo un éxito, debido a que las funciones están presentes en el proyecto, la interfaz es agradable e intuitiva para el usuario, la búsqueda cumple con los requisitos y se tomó en cuenta el archivo completo para la prueba de

VI. Conclusiones

Queda como conclusión que usar Python facilita el desarrollo debido a su sintaxis simple y la disponibilidad de bibliotecas que simplifican tareas comunes, como la manipulación de archivos de audio y la creación de interfaces gráficas. La estructura TDA Lista es sencilla y eficiente para operaciones básicas como agregar, eliminar y reordenar elementos.

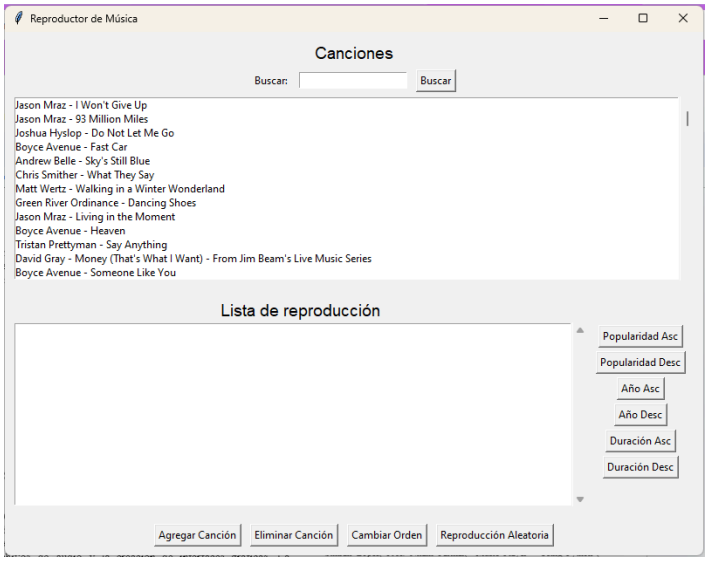
VII. Trabajos futuros

Se tiene planeado aplicar la estructura desarrollada con una mejor interfaz gráfica, para que sea más agradable con el usuario. Por otro lado, se pensaba añadir un enlace directo para que se pueda consultar a YouTube y se reproduzca la canción seleccionada o dependiendo de la reproducción aleatoria. Asimismo también se puede implementar el mismo proyecto, pero utilizando diferentes estructuras y lenguajes de programación.

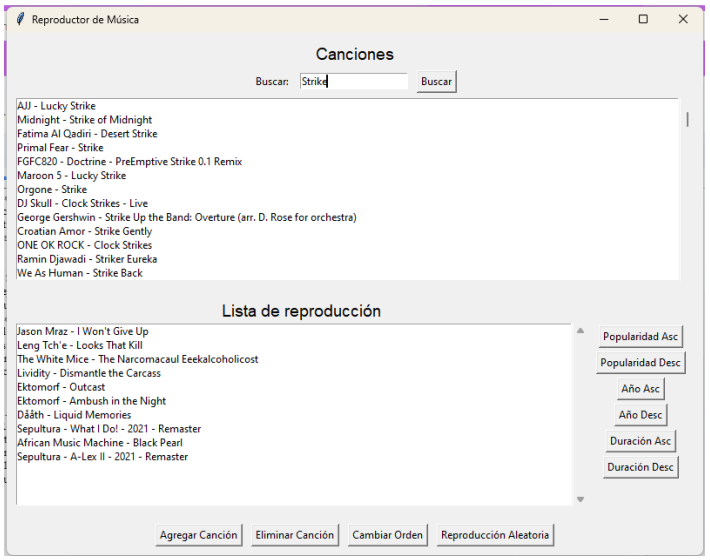
VIII. Resultados

A continuación se mostrará el resultado del proyecto "Reproductor de Música", se creó una interfaz funcional y entendible para el usuario. Los botones creados están correctamente etiquetados y funcionales, se quedó en utilizar el color de la interfaz debido a que es más agradable a la vista del usuario.

La imagen mostrada a continuación es la única ventana visible para el usuario, mostrará solo las canciones disponibles y el artista ordenadas de acuerdo al archivo de prueba.



Como se observa la Lista de Reproducción se encuentra vacía debido a que todavía no se utilizaron los botones de "Añadir Canción" o "Eliminar Canción".



En la última imagen ya se tiene una PlayList con canciones escogidas por el usuario y ya teniendo esta parte lista se puede poner a prueba las siguientes funciones como ordenar las canciones en base a: popularidad, año y duración. Por otro lado, en la parte inferior están las demás funciones para el orden ya sea Aleatorio o escogido por el usuario.

Referencias

[1] Mr. Mehraan Khan, Ms. Leelasa Thakur, Ms. Manika Arunkumar, Ms. Simran Lopes, Prof. Nilam Parmar, "Music Player – Using Python", "International Journal of Research in Engineering and Science (IJRES)", Jun. 2021.

[2] Brian MacFee, Gert Lanckriet, "*HYPERGRAPH MODELS OF PLAYLIST DIALECT", 2012.*

[3] Raúl González Duque, "Python para todos", Nov. 2018.

[4] Sven Amann; Sebastian Proksch; Sarah Nadi; Mira Mezini, "A Study of Visual Studio Usage in Practice", "2016 IEEE 23rd International Conference on Software Analysis, Evolution, and Reengineering (SANER)", March 2016.

[5] Valerio Cosentino, Javier Luis, and Jordi Cabot, "Findings from

GitHub: methods, datasets and limitations", " MSR '16: Proceedings of the 13th International Conference on Mining Software Repositories", May 2016.

[6] Dr. Mariano Méndez, "Algoritmos y Programación II Tda Lista y sus Derivados", Marzo 2020.

[7] Matthew E. P. Davies, Philippe Hamel, Kazuyoshi Yoshii, and Masataka, "Automatic Creation of Multi-Song Music Mashups" , IEEE/ACM transactions on audio, speech, and language processing, vol. 22, no. 12, December 2014

[8] Sushmita G. Kamble ; A. H. Kulkarni, "IEEE/ACM transactions on audio, speech, and language processing", vol. 22, no. 12, December 2014 2016 International Conference on Advances in Computing, Communications and Informatics (ICACCI)

[9] Nirmal R Bhalani ; Jaikaran Singh ; Mukesh Tiwari, "Karaoke Machine implementation and validation using Out of Phase Stereo method" in 2012 International Conference on Communication, Information & Computing Technology (ICCICT)