## **Problema 1**

El conocido reproductor de música **Spotify**, te contrató para hacer un programa para manejar distintos requerimientos de los usuarios a la hora de gestionar sus *playlists*, a continuación se mencionan las funcionalidades deseadas.

## **Requerimientos funcionales**

1. **Cargar Playlist:** Se debe crear una **Lista** de canciones a partir de la información ingresada de un txt, que tendrá la estructura tal como se indica en Estructura de la Playlist. En Ejemplo de Input, se muestra cómo será el txt. Por otra parte, el usuario podrá cargar canciones a su lista, dichas canciones deben tener la misma estructura de la Playlist comentada. Las canciones deben ser ingresadas a la lista en el orden en el que están en el txt.
2. **Buscar Canción:** Se debe poder buscar una canción por su **nombre**, para esto el usuario debe ingresar el nombre de la canción y se mostrará si la canción existe o no dentro de la playlist. En caso de que exista, se mostrarán todos los datos de la canción encontrada.
3. **Filtrar por Artista:** El programa debe ser capaz de mostrar todas las canciones de un **artista**, para esto el usuario especificará el nombre del autor y se deberán mostrar por pantalla todas las canciones que sean de su autoría. Tenga en cuenta que los nombres de los autores son únicos.
4. **Reproducción**: el usuario debe tener una opción que reproduzca la lista de reproducción, el cual haga que aparezca la primera canción de la lista . Cuando la lista llega al final, debe mostrar un mensaje al usuario para que se acabe la reproducción.
5. **Reproducción en Bucle:** El usuario debe poder escuchar su playlist en bucle, para esto debe seleccionar una opción que le permita reproducir de forma aleatoria la playlist sin repetir canciones, una vez reproducida todas las canciones, se debe avisar al usuario con un mensaje y debe empezar a reproducir todo de nuevo.
6. **Eliminar Bucle:** se debe poder quitar el bucle si está activo para pasar las canciones en el orden en el que fueron ingresadas, una vez llegue al final de la lista, se debe indicar al usuario que ya escuchó toda la playlist
7. **Ordenar por Número de Canción:** La playlist se debe poder ordenar por el **número de la canción**. El usuario debe poder elegir ver la lista en este orden de forma creciente o decrecientemente.
8. **Ordenar por Número de Reproducciones:** La playlist se debe poder ordenar por el **número de reproducciones**. El usuario debe poder elegir ver la lista en este orden de forma creciente o decrecientemente.
9. **Ordenar por Nombre de Canción:** La playlist se debe poder **ordenar alfabéticamente** por los nombres de las canciones. El usuario debe poder elegir ver la lista en este orden de forma creciente o decrecientemente.
10. **Mostrar Playlist:** El usuario debe poder ver la información de la playlist, para esto se debe **imprimir** en la interfaz **todas las canciones**. En caso de que la lista sea Circular, se deben imprimir las canciones dos veces.
11. **Historial de Canciones Escuchadas:** Cada vez que se seleccione la opción de reproducir, las canciones que se reproduzcan deben ser ingresadas en un historial.
12. **Contar canciones en el historial**: se debe crear un diccionario que tenga como claves las canciones y como valores la cantidad de veces que aparece en el historial.

**Nota:** Las ordenaciones deben ser hechas por ustedes a través de la implementación de algoritmos, es decir, no se permite utilizar **sort** para ordenar la lista de canciones, ni cualquier otro método preestablecido por python. Ejemplos de algoritmos de ordenación son: metodo burbuja, quick sort, mergesort.

Debe realizar el diagrama de clases, diagrama de flujo, y deben tener en cuenta los Principios Kiss, Dry, YAGNI, AHA, y SOLID.

## **Estructura de la Playlist**

*“Número de Canción”, “Nombre de la Canción”, “Artista”, “Número de Reproducciones”, “Año”, “Idioma”*

○ Los **nombres de las canciones** únicamente contendrán letras, no tendrán números, caracteres especiales ni acentos. Además, los nombres de canciones son únicos

○ El **artista** es un nombre único, no se tomarán en cuenta canciones de varios artistas.

## **Ejemplo de TXT**

## *1,STAR WALKING,Lil Nas X,32341,2022,Ingles*

*2,Not Today,BTS,25824,2017,Coreano*

*3,Vocabulario Basico,Rawayana,29485,2013,Español*

*4,I Took a Pill in Ibiza,Mike Posner,150576,2016,Ingles*

*5,DNA,BTS,123654,2017,Coreano*

*6,Payphone,Maroon V,42345,2012,Ingles*

*7,Uptown Funk,Mark Ronson,254392,2014,Ingles*

*8,Me muero por conocerte,Alex Ubago,83943,2001,Español*

*9,Algo Electrico,Aditus,24454,1985,Español*

*10,Sugar,Maroon v,943454,2015,Ingles*

*11,HIGH,RawayanA,43082,2016,Español*

*12,Running Up That Hill,Kate Bush,64387,1985,Ingles*

*13,Waiting For Love,AvicII,274083,2014,Ingles*

*14,Sigo Aqui,Alex Ubago,173864,2002,Español*

*15,Wake Me Up,Avicii,374927,2013,Ingles*

## **Criterios a Evaluar:**

***Robustez***

El programa maneja valores erróneos introducidos por el usuario, manteniendo su operatividad.

***Usabilidad***

El programa emite mensajes de alerta cuando se introducen valores erróneos. Es fácil de usar.

***Eficiencia***

La estructura de datos adoptada es eficiente, respondiendo adecuadamente a los requerimientos planteados y manteniendo complejidades temporales aceptables.

***Funcionalidad***

Implementación de las funcionalidades indicadas.

**Bonus:**

* Realiza la búsqueda a través del algoritmo de Búsqueda Binaria

**Links de Interés:**

<https://es.khanacademy.org/computing/computer-science/algorithms/binary-search/a/binary-search#:~:text=La%20b%C3%BAsqueda%20binaria%20es%20un,ubicaciones%20posibles%20a%20solo%20una>.

<https://www.cs.buap.mx/~titab/files/Algoritmos_de_ordenacion.pdf>

<https://codeyourapps.com/el-principio-dry-no-te-repitas/#:~:text=El%20principio%20DRY%20(Don't,depurar%20o%20modificar%20tu%20c%C3%B3digo>.

<https://codeyourapps.com/el-principio-kiss-mantenlo-simple-estupido/>

<https://codeyourapps.com/principio-yagni-no-vas-a-necesitarlo/#:~:text=El%20principio%20YAGNI%20o%20You,a%C3%B1adirse%20cuando%20sean%20estrictamente%20necesarios>.

<https://codeyourapps.com/el-principio-aha-evite-las-abstracciones-apresuradas/>

<https://profile.es/blog/principios-solid-desarrollo-software-calidad/>