**Trabajo Especial Programación Orientada a Objetos**

**Ayudante:** Juan Manuel

**Integrantes:**

Francisco Gonzalez

Maximiliano Alvarez

Nicolas Attilio

**Introducción**

Un sistema de administración de música permite organizar nuestra colección musical en base a cierta información relevante que contienen los archivos o pistas de audio.

**Desarrollo**

A la hora de plantear el problema, decidimos comenzar con la Clase Abstracta ElementoMusical, debido a que era nuestro centro y punto de partida. Elegimos hacerla abstracta ya que no íbamos a tener un ElementoMusical instanciado, además, desde aquí pudimos partir hacia la Pista y la Playlist. Continuamos con lo concreto, creamos la Clase Pista, concreta, que era contenedora de todos los datos de la canción (id, año, título, etc.), con sus getters, setters y luego con la redefinición de todos los métodos abstractos de ElementoMusical, por su herencia.

La segunda decisión que tomamos fue crear una Clase llamada Playlist para lograr abstraer el comportamiento tanto de la Playlist Simple como de la Playlist Automática. Al heredar de ElementoMusical, volvimos a definir todos los métodos abstractos. Como se menciona anteriormente, tomamos este camino por facilidad de implementación a la hora de crear las playlists automáticas y las playlists simples.

Dentro de ElementoMusical decidimos colocar el método “buscar” y “copiar”, ya que luego serían redefinidos por cada una de las clases para sus propios usos basándose en los criterios. Para dichos Criterios, creamos la Interfaz Criterio con el método abstracto cumple, el cual devuelve un boolean que define si lo cumplía o no. En esta instancia se nos presentó la duda de crear entre la Interfaz “Criterio” y los filtros de búsqueda pertinentes, dos clases llamadas “FiltroSimple” y “FiltroCompuesto” para poder abstraer los comportamientos, pero llegamos a la conclusión de que no valía la pena porque era simplemente abstraer atributos y no métodos, por lo que optamos por realizar todos los criterios definidos (10 en total), siguiendo solamente de la Interfaz ya mencionada.

Las estructuras más utilizadas fueron: Vectores para las Playlists y ArrayLists para las búsquedas.

A la hora de abstraer comportamientos tuvimos varios dilemas cuando nos encontrábamos con que no siempre era necesario definir un método abstracto en todas las clases hijas ( en algunas solo se retornaba por ejemplo un 1, ya que no podíamos dejarla sin implementar). Esto nos hizo replantear la ubicación de varios métodos en el diagrama y por ende en el código, pero llegamos a la conclusión (luego de un largo debate) de que era necesaria esa abstracción por más que, como ya se dijo, nos causara dudas ese código de una línea con solo un return.

Para implementar el main creamos una clase contenedora llamada Biblioteca Musical, donde se realizó el testing propuesto con sus debidas instanciaciones. Se realizó un recorrido en una lista sumando las duraciones de una lista de pistas debido a que uno de los puntos lo solicitaba y no teníamos un método para hacerlo.