Universidad Católica del Norte
Facultad de Ingeniería y Ciencias Geológicas
Departamento de Ingeniería de Sistemas y Computación

ESTRUCTURAS DE DATOS

☆ Profesora: Angélica Flores.
☆ Ayudantes: Byron Letelier, Benjamín Miranda.

Taller N°1: C++ (I-2025)
Periodo: 24 de marzo al 27 de abril

I. Objetivo

Implementar arreglos estáticos/dinámicos y listas enlazadas utilizando punteros en el lenguaje de programación C++.

II. Enunciado

Los Bunkers son una banda nacional oriunda de Concepción, activa desde 1994 hasta la fecha, con una pausa en sus actividades entre 2014 y 2022. El año pasado, la banda grabó su cuarto álbum en vivo: *Los Bunkers MTV Unplugged* (ver Figura 1), el cual consiste en un concierto en vivo donde todos los instrumentos tocados son acústicos, y que es producido por el canal MTV.

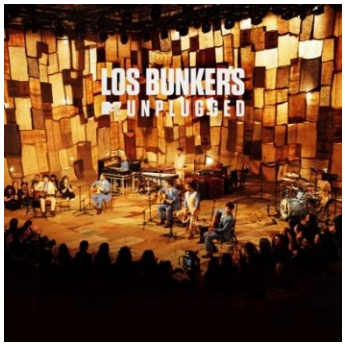

Figura 1. Caratula del álbum "Los Bunkers MTV Unplugged"

Debido a la gran trayectoria de la banda, la tienda de vinilos "Discos G" desea crear una aplicación que permita a los usuarios conocer mejor su música.

2.1. Archivos de Entrada

El equipo administrativo de la tienda ha recopilado la información necesaria en dos archivos en formato .csv con campos separados por ";":

- **Archivo "albumes.csv"**
  Contiene la información de todos los álbumes de Los Bunkers (ver ejemplo en Figura 2). Cada registro tiene el siguiente formato: *id, título, año, es_de_estudio, canciones*

```
1;Los Bunkers;2001;Sí;{1,2,3,4,5,6,7,8,9,10,11}
2;Canción de Lejos;2002;Sí;{12,13,14,15,16,17,18,19,20,21,23}
3;La Culpa;2003;Sí;{24,25,26,27,28,29,30,31,32,33,34,35}
4;Vida De Perros;2005;Sí;{36,37,38,39,40,41,42,43,44,45,46}
```

Figura 2. Ejemplo de archivo albumes.csv

  - Id: identificador único del álbum. Es un valor incremental, siendo 1 el primer álbum publicado por la banda.
  - Título: nombre del álbum.
  - Año: año en que se publicó el álbum.
  - Es_de_estudio: puede tomar los valores "Si" (si fue grabado en un estudio) o "No" (si fue grabado en vivo).

- Canciones: arreglo que contiene los Id de todas las canciones pertenecientes a este álbum. El inicio y el fin del arreglo se delimitan con el uso de llaves { }.

♦ **Archivo "canciones.csv"**

Contiene la información de todas las canciones de la banda (ver ejemplo en Figura 3). Cada registro tiene el siguiente formato:

Id_cancion, album_id, titulo, reproducciones, duración

```
1;1;El detenido;3,724,811;5:11
2;1;Fantasias Animadas de Ayer y Hoy;2,70,733;2:11
3;1;No Se;774,729;5:05
4;1;Buscando Cuadros;616,915;2:17
5;1;Jamas;616,594;3:24
```

Figura 3. Ejemplo de archivo canciones.csv

- Id_canción: identificador único de la canción. Es un valor incremental que comienza en 1 para la primera canción del álbum y aumenta sucesivamente. Al llegar a la última canción del álbum, el contador se reinicia en 1 para el siguiente álbum.
- Álbum_id: identificador del álbum donde está integrada esta canción.
- Título: nombre de la canción.
- Reproducciones: cantidad de veces que fue reproducida la canción en Spotify.
- Duración: tiempo que dura la canción en formato Mi:SS.

2.2. Carga Inicial

Cuando la aplicación inicie, se deben leer los archivos mencionados en la Sección 2.1. y almacenar su información en las estructuras de datos descritas en la Sección III.

2.3. Menú

Luego de la lectura, el programa debe mostrar un menú que permita interactuar con las siguientes opciones:

A. Mostrar álbum: Dado el Id de un álbum, el sistema debe mostrar su información junto al nombre de las canciones que contiene (ver ejemplo en Figura 4).

```
Busqueda de album
Ingrese el id del album: 1
Album encontrado
Id : 2
Titulo : Segundo Album
Anio: 2015
Es de estudio : Si
Canciones :
- Cancion Dos
- Cancion Tres
- Cancion Cuatro
```

Figura 4. Ejemplo en consola de "Mostrar álbum"

B. Búsqueda avanzada de canciones: El sistema debe mostrar: el título, nombre del álbum al que pertenece y duración, de todas las canciones que resulten tras haber aplicado los siguientes filtros secuencialmente y en orden:

a. **Filtro por rango de años:** si el usuario confirma que desea aplicarlo, entonces debe ingresar dos números enteros positivos para que el sistema solo considere las canciones de los álbumes que se publicaron entre estos años.
- *Nota 1:* La búsqueda debe incluir a los años ingresados.
- *Nota 2:* Sin importar cuál de los valores sea mayor (el primero o el segundo), la búsqueda se debe realizar en un sentido o en el otro (ver Figura 5).

```
Desea filtrar por intervalo de anios (Si/No)? Si
Anio Inicial: 2004
Anio Final: 2010
```

Figura 5. Ejemplo del criterio de búsqueda por rangos de años

b. **Filtro por álbum:** si el usuario confirma que desea aplicarlo, entonces el sistema debe listar todos los Id y títulos de álbum que hayan pasado el filtro anterior (ver Figura 6). El usuario debe luego elegir uno ingresando su correspondiente Id.

```
Desea filtrar por album (Si/No): Si
Albumes:
[1] - Album Uno
[2] - Album Dos
[3] - Album Tres
: 2
```

Figura 6. Ejemplo del criterio de búsqueda por álbum

c. **Filtro por título:** si el usuario confirma que desea aplicarlo, entonces debe ingresar la cadena de texto correspondiente. La búsqueda debe asegurar que la entrada coincida con una parte del título de la canción, respetando mayúsculas y minúsculas según lo esperado. (ver Figura 7).

```
Desea buscar por titulo de la cancion? (Si/No): Si
Titulo de la cancion: Sur
```

Figura 7. Ejemplo del criterio de búsqueda por coincidencia de titulo

d. **Ordenamiento:** si el usuario confirma que desea que las canciones se entreguen ordenadas según el número de reproducciones, entonces el sistema debe preguntar el sentido (ascendente o descendente) y cuál algoritmo de ordenamiento desea aplicar: BubbleSort o QuickSort (ver Figura 8).

```
Desea ordenar por reproducciones (Si/No) ?: Si
De forma ascendente o descendente (Ascendente/Descendente): Ascendente
Algoritmo de ordenamiento (BubbleSort/InsertionSort)?: InsertionSort
```

Figura 8. Ejemplo del criterio de búsqueda ordenamiento

C. Eliminar álbum: El sistema debe permitir eliminar un álbum dado su título. Si se encuentra, entonces se elimina tanto el álbum como todas las canciones asociadas a éste (ver ejemplo en Figura 9).

```
Eliminar un album
Elige un nombre del album a eliminar: La Culpa
Se ha borrado el album
```

Figura 9. Ejemplo en consola de "Eliminar álbum"

Nota: no debe tener sensibilidad para mayúsculas y minúsculas, es decir, que, si un usuario ingresa "la culpa" o "La Culpa", ambos deben ser reconocidos.

D. Trivia: El sistema debe permitir al usuario jugar una pequeña trivia de tres preguntas sobre la información que se maneja de Los Bunkers. Cuando ya ha entregado sus tres respuestas, el sistema indica si fueron correctas o no. Solo si acierta a las tres, gana (ver Figura 10).

```
Pregunta 1: Correcta
Pregunta 2: Incorrecta
Pregunta 3: Incorrecta
Has perdido :(
```

Figura 10. Resultado de jugar a la trivia

Los tres tipos de preguntas son:

a. **Posición en un álbum**: Se elige una canción aleatoria de un álbum aleatorio. La pregunta a realizar es: "Dada la canción '<*titulo canción*>', ¿qué posición tiene en el álbum '<*título álbum*>'?". Considere que "*título canción*" y "*título álbum*" deben ser reemplazados.
Se deben mostrar cuatro alternativas donde solo una sea la correcta y las otras tres sean generadas de forma aleatoria. La Figura 11 muestra un ejemplo.

```
Dada la cancion 'El dia en Que Dejaste de Fingir'
 Que posicion tiene en el album 'La Velocidad de la Luz' ?
A. 1
B. 11
C. 6
D. 4
:
```

Figura 11. Ejemplo del tipo de pregunta 1 de la trivia

b. **Canción con más reproducciones**: Se elige un álbum al azar y se escoge la canción con más reproducciones. La pregunta a realizar es: "Dadas las siguientes canciones del álbum '<*título álbum*>', ¿cuál es la más escuchada?". Considere que "*título álbum*" debe ser reemplazado.
Se deben mostrar cuatro alternativas donde solo una sea la correcta y las tres restantes correspondan a otras canciones del mismo álbum. La Figura 12 muestra un ejemplo.

```
Dadas las siguientes canciones del album Barrio Estacion,
cual es la mas escuchada?
A. Una Nube Cuelga Sobre Mi
B. Me muelen a palos
C. Nada Nuevo Bajo El Sol
D. El Tiempo Que Se Va
:
```

Figura 12. Ejemplo del tipo de pregunta 2 de la trivia

c. **Canción con menor duración**: Se elige un álbum al azar y se selecciona su canción más corta. La pregunta a realizar es: "Dadas las siguientes canciones del álbum '<*título álbum*>', ¿cuál es la de menor duración?". Considere que "*título álbum*" debe ser reemplazado.
Se deben mostrar cuatro alternativas donde solo una sea la correcta y las tres restantes correspondan a otras canciones del mismo álbum. La Figura 13 muestra un ejemplo.

```
Dadas las siguientes canciones del album Musica Libre,
cual es la de menor duracion?
A. Y Nada Mas
B. Pequenia Serenata Diurna
C. Leyenda
D. Santiago de Chile
:
```

Figura 13. Ejemplo del tipo de pregunta 3 de la trivia

E. Visualizar Álbumes **(OPCIONAL):** Al interactuar con esta opción, el sistema debe mostrar la lista enlazada de álbumes de forma gráfica, parecido a como se visualiza en la Figura 14.

```
(1) -> (2) -> (3) -> (4) -> (5) -> (6) -> (7) -> (8)
```

Figura 14. Ejemplo gráfico de la lista enlazada

La implementación de este punto no es obligatoria. Quienes lo hagan pueden optar a ganar hasta 2 puntos extra.

F. Salir: Al interactuar con esta opción el programa debe realizar lo siguiente:
a. Borrar todos los punteros asociados a las estructuras de datos utilizadas.
b. Guardar en el archivo canciones.csv y albumes.csv las canciones y álbumes

III. Estructuras de Datos

La implementación de las siguientes estructuras de datos es **obligatoria**. En caso de que alguna no sea implementada, se utilice otra estructura de datos en su reemplazo, o se recurra a una librería que contenga la implementación interna, el taller completo será evaluado con la nota mínima.

3.1. Arreglo Dinámico

La ventaja de un arreglo dinámico es que puede aumentar o disminuir su tamaño al variar la cantidad de datos que contiene. Por ejemplo, si el arreglo tiene una capacidad inicial de 5 elementos, al

insertar el número 6 deberá expandirse. El sistema debe permitirlo mediante la reasignación de memoria utilizando la función nativa “realloc”.

Todas las canciones deben estar almacenadas en un arreglo dinámico. (ver Figura 15).

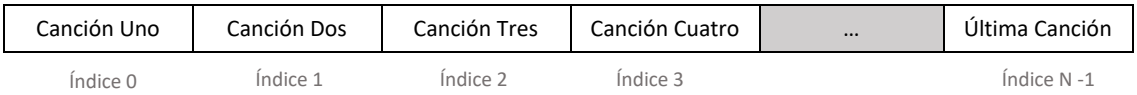

Figura 15. Arreglo dinámico que almacena todas las canciones del sistema

3.2. Lista Enlazada Simple

Los álbumes deben estar guardados en una lista enlazada simple con un índice asociado a cada nodo (partiendo en cero). Debe estar ordenada de menor a mayor según el Id de álbum (ver Figura 16).

Figura 16. Estructura de lista enlazada que almacena álbumes

3.3. Relación Álbum - Canción

El campo “canciones” dentro la clase Álbum debe tener un listado con los **índices** de todas las canciones asociadas a él separadas por coma. Estos ***índices*** corresponden a la posición donde están almacenadas las canciones en el arreglo dinámico de la Sección 3.1. La Figura 17 ilustra un ejemplo.

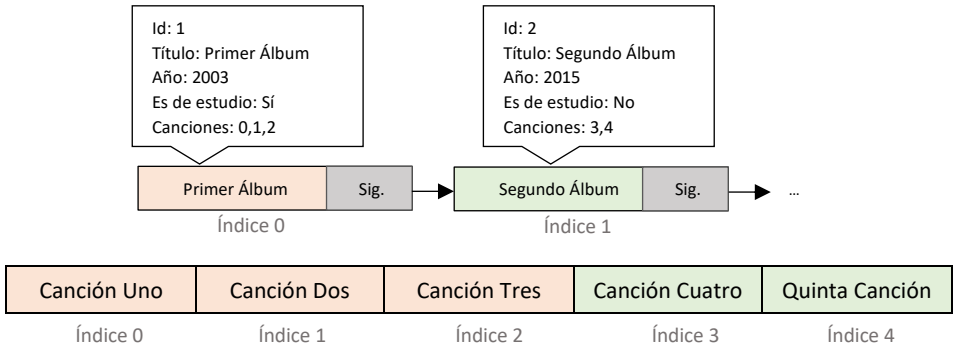

Figura 17. Ejemplo de relación entre la lista enlazada y el arreglo dinámico

IV. Informe

Adicional al programa escrito en C++, se debe entregar un informe que explique el proceso de elaboración del taller.

La estructura del informe debe considerar:

- **Portada:** Página inicial del informe que debe contener:
  - Nombre y apellidos de la docente
  - Nombres y apellidos de los ayudantes

  - Nombres, apellidos y correos electrónicos de los estudiantes
  - Título o nombre del taller (por ejemplo: Taller 1).
- **Introducción:** Breve explicación sobre el problema a abordar junto a cómo se presentará la solución.
- **Cuerpo:** Debe explicar:
  - Metodología para resolver el problema. Se debe explicar como se abordó el problema. Por ejemplo, describir si se dividió en pasos para llegar al resultado o si se aplicó algún otro enfoque sistemático.
  - Diagrama de Clases que represente los elementos esenciales para su funcionamiento, como clases principales, atributos, métodos y relaciones entre ellas.
  - Explicar cómo se implementaron las estructuras de datos utilizadas y su propósito en el código.
  - Estrategias para abordar el taller. Se deben explicar las tácticas utilizadas, por ejemplo: planificación previa, identificación de problemas clave, o la división del trabajo en el equipo.
- **Conclusión:** análisis final sobre lo aprendido, las dificultades enfrentadas y qué se debería mejorar para el siguiente taller.

Se descuentan 0,2 décimas por cada falta de ortografía, puntuación y/o redacción.

V. Entrega

5.1. Fecha

La fecha límite para la entrega del taller es el 27 de abril hasta las 23:59 hrs. Por cada hora de atraso en la entrega, se descontará un punto a la nota final del taller, es decir: si la entrega llega entre 1 y 60 minutos tarde, implica un descuento de 10 décimas; si llega entre 61 y 120 minutos tarde, implica un descuento de 20 décimas, y así sucesivamente.

5.2. Entregables

Se debe subir:

1. Programa en C++. Proyecto completo en la IDE Clion o Visual Studio.
2. Informe escrito en formato .docx.

5.3. Integrantes

El taller se puede realizar de forma individual o en parejas.

5.4. Formato de Entrega

El taller se debe subir a GitHub utilizando el controlador de versiones Git. El repositorio debe estar en privado y contener contribuciones de ambos integrantes, si el taller se resuelve solo, solo debe haber contribuciones de ese estudiante. A la hora de enviar el proyecto, se debe agregar al repositorio en rol de colaborador a los ayudantes:

- byron.letelier@alumnos.ucn.cl
- benjamin.miranda02@alumnos.ucn.cl

El taller se revisará hasta el commit más reciente antes el último plazo de entrega del taller, 5:00 AM (con descuento). Además, como respaldo, se debe subir **obligatoriamente** a Campus Virtual el enlace del repositorio de Github en un archivo de texto. De no respetarse esto último, se aplicará un descuento de 10 décimas.

La no entrega del proyecto implicará la no revisión del taller y, por lo tanto, su calificación con la nota mínima (1.0).

Nota: En el caso de no haber contribuciones de un integrante (commits), este recibirá la nota mínima.

5.5. Interrogación

Todos los estudiantes que cursen la asignatura Estructura de Datos serán interrogados al menos una vez durante el semestre en relación con sus entregas de talleres. El propósito de esta interrogación es asegurarse de que los participantes puedan demostrar el conocimiento y la participación en el trabajo entregado.

En términos de programación, resulta vital señalar que la contribución de cada miembro del equipo al taller debe ser equitativa y esencial para la construcción de las estructuras de datos. En otras palabras, no se permitirá que un estudiante se encargue únicamente de implementar las estructuras de datos mientras que otro se dedica exclusivamente a crear métodos “get” y “set”, realizar despliegues por consola, o desarrollar la documentación. Es fundamental que ambos demuestren su dominio de las estructuras de datos.

En los días posteriores a la entrega del taller, el ayudante enviará un correo electrónico a todos los seleccionados para la interrogación con el fin de coordinar el lugar y el horario en el que se llevará a cabo. Es importante destacar que esta reunión debe ser obligatoriamente presencial y, en caso de que sea en parejas, ambos integrantes deben asistir. Si alguno de ellos no asiste, ambos recibirán la calificación mínima de 1.0.

La interrogación de los estudiantes seleccionados implica mantener la nota obtenida o reducirla según lo estime el ayudante, en la medida que sea necesario. Por ejemplo, si se determina que el estudiante no participó efectivamente en el taller, la calificación podría reducirse de 7.0 a 1.0.

5.6. Ejecución

En el caso de que el proyecto no compile, el taller se evaluará con nota máxima 4.0.

Incluso si el estudiante tiene alguna bonificación, por ejemplo, décimas de ayudantía, no se puede superar esta nota. Sobre ella se deben aplicar también todos los descuentos en los que pueda haber incurrido el estudiante.

5.7. Usabilidad

La ejecución del programa debe ser amigable con el usuario, esto significa que debe cumplir lo siguiente:

- Durante la ejecución del programa, siempre se muestran todas las opciones disponibles, asegurando que el usuario pueda interactuar con el sistema en todo momento.
- Todas las entradas de datos deben ser validadas. Si algo falla, se debe indicar el porqué, y mantener la información ingresada previamente para que el usuario solo corrija el dato erróneo sin tener que volver al menú anterior.

El no cumplimiento de algún inciso provocará descuento en los apartados que asocien alguna de estas actividades.

VI. Rúbrica

Programa en C++ (85%)		
Categoría	**Apartado**	**Puntaje**
Diseño	Se respeta el encapsulamiento de las clases	5
	El programa principal es sencillo (método main)	5
	Posee documentación en el proyecto. Doxygen documentation para CLion y XML documentation para Visual Studio	10
	Utiliza nombres nemotécnicos, tanto para variables, constantes, funciones, métodos, clases y paquetes	5
	El programa se diseña utilizando archivos .h y .cpp	2
Ejecución	Lectura de archivos. Se leen correctamente los archivos “canciones.csv” y “albumes.csv”	10
	Almacenamiento de elementos en las estructuras de datos	5
	Menú	2
	Opción A: Mostrar álbum	5
	Opción B: Búsqueda avanzada de canciones	10
	Opción C: Eliminar Álbum	9
	Opción D: Trivia	12
	Opción F: Salir	5
Total (60% de exigencia)		**85**

Sea P_p el puntaje obtenido, la nota final del programa en C++ se calcula mediante la siguiente expresión:

$$N_p = \begin{cases} \frac{3P_p}{51} + 1, & si\ P_p < 51 \\ \frac{3P_p - 153}{34} + 4, & si\ P_p \geq 51 \end{cases}$$

Informe (15%)		
Categoría	**Apartado**	**Puntaje**
Informe	Portada	5
	Introducción	5
	Cuerpo	15
	Conclusión	10
Total (60% de exigencia)		**35**

Sea P_i el puntaje obtenido, la nota final de informe se calcula mediante la siguiente expresión:

$$N_i = \begin{cases} \frac{3P_i}{21} + 1, & si\ P_i < 21 \\ \frac{3P_i - 63}{14} + 4, & si\ P_i \geq 21 \end{cases}$$

Donde N_p y N_i, se redondean a un solo decimal.

$$Nota\ Final = N_p * 0{,}85 + N_i * 0{,}15$$

¡ÉXITO! ☺