

## Actividad 2.8

Simula una estruc---

marp: true

title: Actividad Principal – Gestor musical

theme: default

\_class: lead

paginate: true

backgroundColor: #ffffff

## Actividad Principal

Gestor musical

## Objetivo

Desarrollar un programa modular en Java que:

- Gestione canciones mediante **nodos enlazados**
- Agrupe canciones por género en **listas dinámicas**
- Permita visualizar los cambios en la estructura tras cada operación

## Funcionalidades obligatorias

Tu aplicación debe permitir:

- Añadir canciones con:
  - Título
  - Artista
  - Duración
  - Género
- Almacenar canciones en **listas enlazadas por género**
- Buscar canciones por género y/o título
- Eliminar canciones de una lista
- Listar canciones agrupadas por género

## Visualización estructural

El programa debe:

- Mostrar la estructura de listas tras cada operación
- Permitir comprender cómo se modifican los enlaces
- Usar consola (texto) para representar los nodos enlazados

 Opcional: incluir diagramas ASCII o mensajes descriptivos

## Opcionales para ampliar (extra)

Si terminas la parte básica, puedes:

- Ordenar canciones por duración o título
- Calcular estadísticas por género (cantidad, tiempo total)
- Implementar una interfaz gráfica simple (Swing/JavaFX)
- Añadir menú interactivo por consola
- Exportar/importar desde archivo


## Condiciones

- Lenguaje obligatorio: **Java**
- Uso de **clases, métodos y referencias**
- Prohibido usar arrays para gestionar las listas
- El código debe ser modular, claro y documentado

 Evalúa tu estructura, no solo el resultado final

## Entregables

- Código fuente Java completo
- Evidencia de pruebas (salidas por consola)
- Documentación breve del diseño de clases
- (Opcional) Justificación de ampliaciones añadidas

 ¡Construye tu propio gestor musical y entiende cómo funcionan las estructuras dinámicas en la práctica!

tura viva: programa con nodos

📌 Diseña una aplicación en **Java** que utilice **listas enlazadas** para gestionar canciones agrupadas por género.



## **Objetivo**

El objetivo de esta actividad es:

- Comprender y aplicar el concepto de **estructura dinámica**
- Representar canciones como **nodos enlazados**
- Manipular listas enlazadas con operaciones reales: añadir, buscar, eliminar, listar

 Simular una **estructura viva**, que se modifica a medida que el usuario interactúa

## ¿Qué debe hacer tu aplicación?

Tu programa en Java debe permitir:

- Añadir canciones con:
  - Título
  - Artista
  - Duración
  - Género
- Almacenar las canciones en **listas enlazadas separadas por género**
- Realizar operaciones sobre cada lista:
  - Buscar canciones
  - Eliminar canciones

## Comportamiento estructural

El sistema debe mostrar, en consola:

- Cambios producidos en la estructura tras cada operación
- Representación clara de la lista por género

 Puedes usar salidas de texto o diagramas ASCII simples

## Opcionales para subir nota

Si terminas la funcionalidad básica, puedes añadir:

- Ordenar canciones por título o duración dentro de cada lista
- Mostrar estadísticas:
  - Número de canciones por género
  - Tiempo total por género
- Crear un **menú interactivo** en consola
- Desarrollar una **interfaz gráfica simple** (Swing o JavaFX)

## Organización del código

El código debe seguir principios de **modularidad** y **claridad**:

- Clases separadas: `Cancion` , `Nodo` , `ListaGenero` , `Main`
- Métodos reutilizables: `agregar()` , `buscar()` , `eliminar()` , `mostrar()`
- Separación entre lógica de datos y lógica de interfaz

## Requisitos técnicos


- Lenguaje: Java
- Uso de **referencias entre objetos** (listas enlazadas)
- Control de errores: listas vacías, género no existente, etc.
- Uso de **clases propias** (POO)
- Mostrar estructura antes y después de cada operación

## Entregables

- Archivo `.java` funcional y bien documentado
- Muestra de pruebas en consola (capturas o ejemplos)
- Diagrama opcional del diseño de clases
- Descripción breve de ampliaciones realizadas (si hay)

## Resultado esperado

Tu aplicación debe:

- ✓ Gestionar listas enlazadas de canciones por género
  - ✓ Permitir operaciones de búsqueda, eliminación, listado
  - ✓ Representar visualmente (por texto) los cambios en la estructura
  - ✓ Ser funcional, modular y mantenible
-  ¡Construye tu propia biblioteca musical y entiende cómo viven las estructuras dinámicas!