

Propuesta Didáctica

Módulo: Programación (1.º DAM)

Actividad: Gestor de música con persistencia en ficheros

2.1 ¿Qué supuesto queremos trabajar?

Diseñar e implementar una aplicación que:

- Gestione una colección de canciones
- Permita añadir, buscar, eliminar y listar
- Organice los datos en estructuras internas
- Almacene y recupere los datos usando **ficheros de texto o binarios**

 Objetivo: aplicar estructuras de datos, ficheros y lógica de control

2.2 Contextualización del alumnado

- Alumnado de 1.º de **Desarrollo de Aplicaciones Multiplataforma**
- Perfil heterogéneo: distintos niveles de experiencia previa
- En fase de consolidación de programación estructurada
- Familiarización progresiva con persistencia y flujo de datos






2.3 Conocimientos previos requeridos

- Declaración de variables y funciones
- Uso de estructuras condicionales y bucles
- Entrada/salida básica por consola
- Manejo de arrays o listas
- Conocimiento inicial del uso de **ficheros en Java o Python**

2.4 Objetivos de aprendizaje

- Manipular estructuras de datos en memoria
- Trabajar con lectura y escritura en ficheros
- Aplicar buenas prácticas en diseño y organización del código
- Comprender la persistencia de la información
- Desarrollar autonomía en resolución de problemas

2.5 Metodología

-  Enfoque práctico: aprender haciendo
-  Trabajo incremental: pequeños hitos funcionales
-  Trabajo individual o por parejas
-  Experimentación con entradas reales
-  Puesta en común de resultados y dificultades

2.6 Material didáctico (DUA)

- Código base comentado y escalonado
- Diagramas de flujo y esquemas de clases
- Ejemplos de ficheros de entrada/salida
- Alternativa textual o visual según necesidad
- Actividades adaptadas con niveles de complejidad diferenciada



2.7 Secuencia de acciones formativas

1. Introducción: ¿qué es la persistencia?
2. Ejemplo guiado: cargar y guardar objetos en fichero
3. Diseño del gestor de canciones (título, artista, duración, género)
4. Implementación paso a paso (añadir, listar, eliminar...)
5. Persistencia mediante ficheros (guardar/cargar colección)
6. Pruebas, validación y presentación del trabajo

2.8 Actividad principal

“Gestor de música con persistencia en ficheros”

El alumnado debe:

- Crear una estructura de almacenamiento para canciones
- Permitir al usuario añadir, buscar, eliminar y listar canciones
- Agrupar por género o duración si se desea
- Guardar la colección en un fichero de texto o binario
- Recuperar la colección al iniciar el programa
- Mostrar mensajes claros y estructurar bien el menú de opciones

 Opcional: permitir edición de canciones existentes o exportación a CSV



2.9 Evaluación: Instrumentos y criterios



Criterios de evaluación:

- Correcto uso de estructuras de datos
- Gestión funcional del almacenamiento en ficheros
- Limpieza y claridad del código
- Interfaz sencilla y efectiva
- Persistencia funcional entre ejecuciones



Instrumentos:

- Rúbrica funcional
- Pruebas con ficheros reales
- Observación del proceso
- Defensa o explicación breve del código

2.10 Inclusión y atención a la diversidad

- Proporcionar plantilla inicial comentada
- Opción de trabajar con datos precargados
- Alternancia entre programación en consola o pseudocódigo
- Acompañamiento individual para reforzar lógica o sintaxis
- Evaluación flexible adaptada al ritmo del estudiante

2.11 Actividades de ampliación


 Para quienes terminen antes:

- Exportar los datos en diferentes formatos (`.csv` , `.json`)
- Implementar ordenación por artista o duración
- Crear una interfaz gráfica simple (Swing, JavaFX, Tkinter...)
- Añadir soporte para múltiples listas o usuarios
- Incluir estadísticas (canción más larga, número de canciones por género)

Cierre

✓ Proyecto completo que combina:

- Lógica de control
- Persistencia
- Estructuración del código
- Interacción con el usuario

 ¡Tu aplicación será una base real para gestionar información de forma estructurada y persistente!