ESPE
UNIVERSIDAD DE LAS FUERZAS ARMADAS
INNOVACIÓN PARA LA EXCELENCIA

Algoritmos de Ordenamiento interno, externo y Búsqueda

Por: Diego Montesdeoca y Ángel Rodriguez

Introducción:

- Este trabajo implementa un sistema completo de gestión de catálogo de películas desarrollado en Java, que combina una interfaz gráfica intuitiva con estructuras de datos fundamentales y algoritmos de ordenamiento eficientes. La aplicación permite no solo administrar un inventario de películas con todas las operaciones CRUD.

- La innovación principal reside en la implementación del algoritmo de Mezcla Natural para el ordenamiento externo de grandes volúmenes de datos, optimizado específicamente para trabajar con archivos JSONL que almacenan el catálogo completo.

Objetivos:

Implementación de Mezcla Natural

- Describir el funcionamiento, ventajas y aplicación práctica del algoritmo de mezcla natural.

Ordenamiento con JSON

- Aprender a implementar el algoritmo de ordenamiento con un archivo JSON para datos estructurados.

Fundamentos de Búsqueda

- Conocer los fundamentos teóricos de los algoritmos de ordenamiento internos, externos y de búsqueda.

Algoritmos de ordenamiento:

Clasificación general:

Ordenamiento Interno

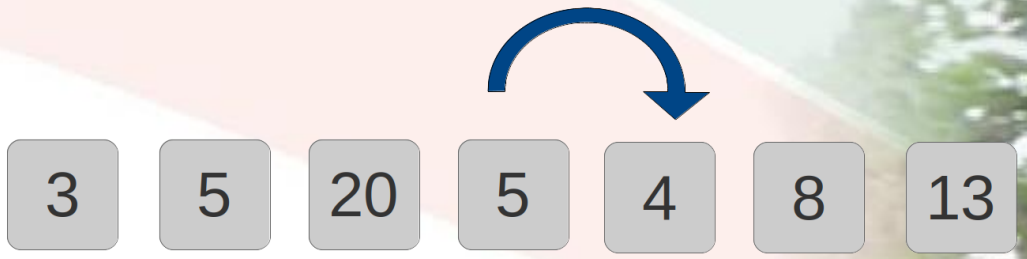
- Se realiza en memoria principal (RAM), ideal para conjuntos de datos pequeños o medianos que caben completamente en la memoria.

Ordenamiento Externo

- Utilizado cuando los datos no caben en memoria, por ejemplo, en discos duros. Implica el uso de técnicas de lectura/escritura optimizadas para manejar grandes volúmenes de información.

Ordenamiento Natural: La Ventaja Adaptativa

- Aprovecha las secuencias ya ordenadas en la entrada, minimizando las operaciones y mejorando la eficiencia en escenarios reales donde los datos suelen tener un orden parcial. Este enfoque adaptativo lo hace excepcionalmente eficaz.



Algoritmos de búsqueda:

Conceptos clave

Búsqueda Secuencial

- Examina elemento por elemento hasta encontrar el valor deseado. Es simple de implementar, pero su eficiencia es baja (O(n)) para grandes conjuntos de datos.

Búsqueda Binaria

- Requiere que los datos estén previamente ordenados. Utiliza la estrategia "divide y vencerás" para encontrar elementos en tiempo logarítmico (O(log n)), siendo mucho más rápida.

Elección del Algoritmo

- La selección adecuada depende críticamente de la estructura y el orden de los datos. Una buena elección puede significar una gran diferencia en el rendimiento.

Los algoritmos de búsqueda (como la lineal y la binaria) localizan un elemento específico dentro de un conjunto de datos. Los algoritmos de ordenación (como el de burbuja o el rápido) reorganizan los elementos de un conjunto para que queden en un orden específico (numérico o alfabético).

El algoritmo de mezcla natural:

Características

El algoritmo de mezcla natural es una variante avanzada del Merge Sort que optimiza el proceso de ordenamiento al identificar y aprovechar las secuencias ya ordenadas, o "runs", presentes en los datos de entrada.

Su funcionamiento consiste en dividir la lista en estas sublistas naturalmente ordenadas y luego fusionarlas eficientemente hasta obtener una lista completamente ordenada.

Su complejidad promedio es O(n log n), lo que lo convierte en un método estable y altamente eficiente para procesar datos parcialmente ordenados, superando al Merge Sort clásico en estos escenarios.

La principal ventaja de este algoritmo es su capacidad para reducir significativamente el número de comparaciones y movimientos de datos, especialmente cuando los datos de entrada tienen un orden parcial preexistente.

Comparativa con otros algoritmos:

ESPE
UNIVERSIDAD DE LAS FUERZAS ARMADAS
INNOVACIÓN PARA LA EXCELENCIA

Bubble Sort y Selección

Estos algoritmos son simples de entender e implementar, pero su complejidad $O(n^2)$ los hace ineficientes para grandes volúmenes de datos.

Merge Sort Clásico

Ofrece una complejidad garantizada de O(n log n) en todos los casos, pero no aprovecha el orden parcial de los datos, realizando siempre el mismo número de operaciones.

Mezcla Natural

Mantiene una complejidad de O(n log n), pero mejora significativamente el rendimiento en casos reales con datos parcialmente ordenados al reducir las comparaciones y movimientos.

Ordenamiento Externo

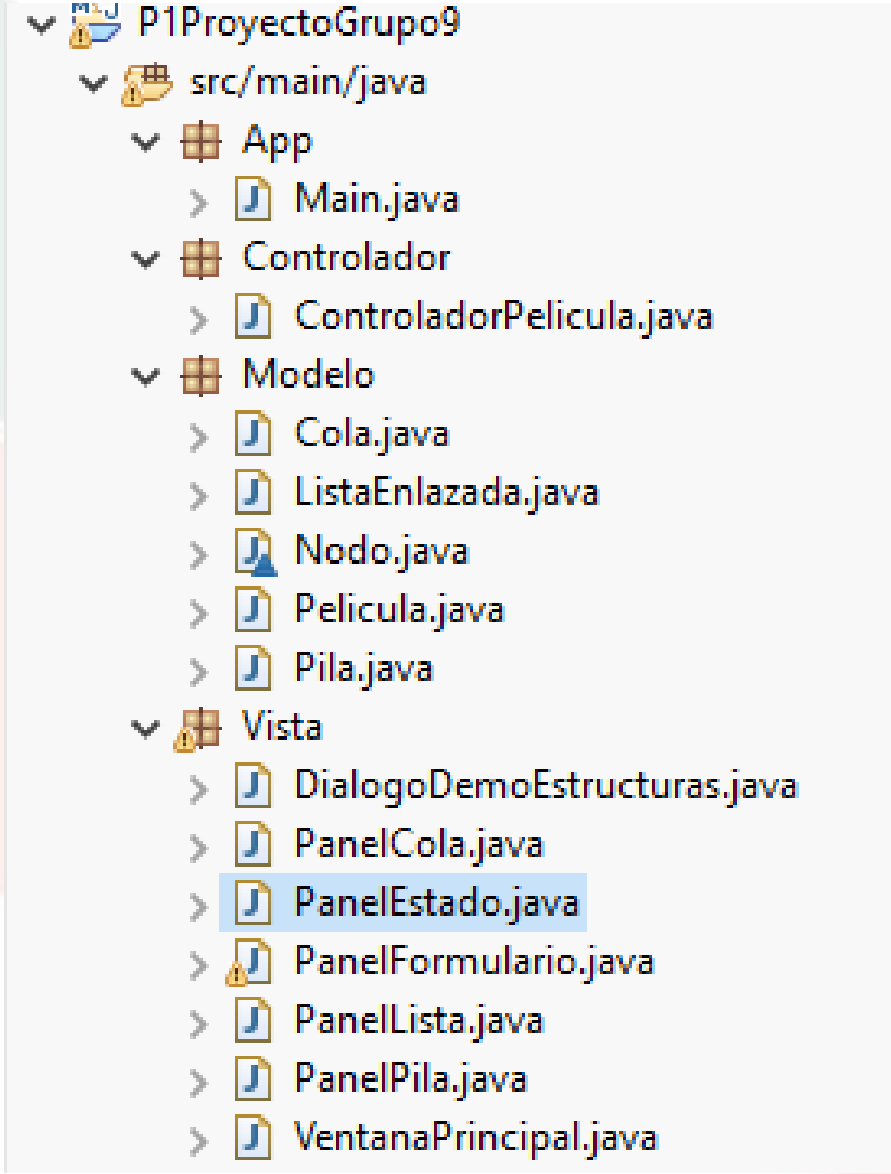
Es indispensable para datos muy grandes que no caben en la memoria. Aunque es más complejo de implementar, su uso es crítico en escenarios de big data.

Desarrollo:

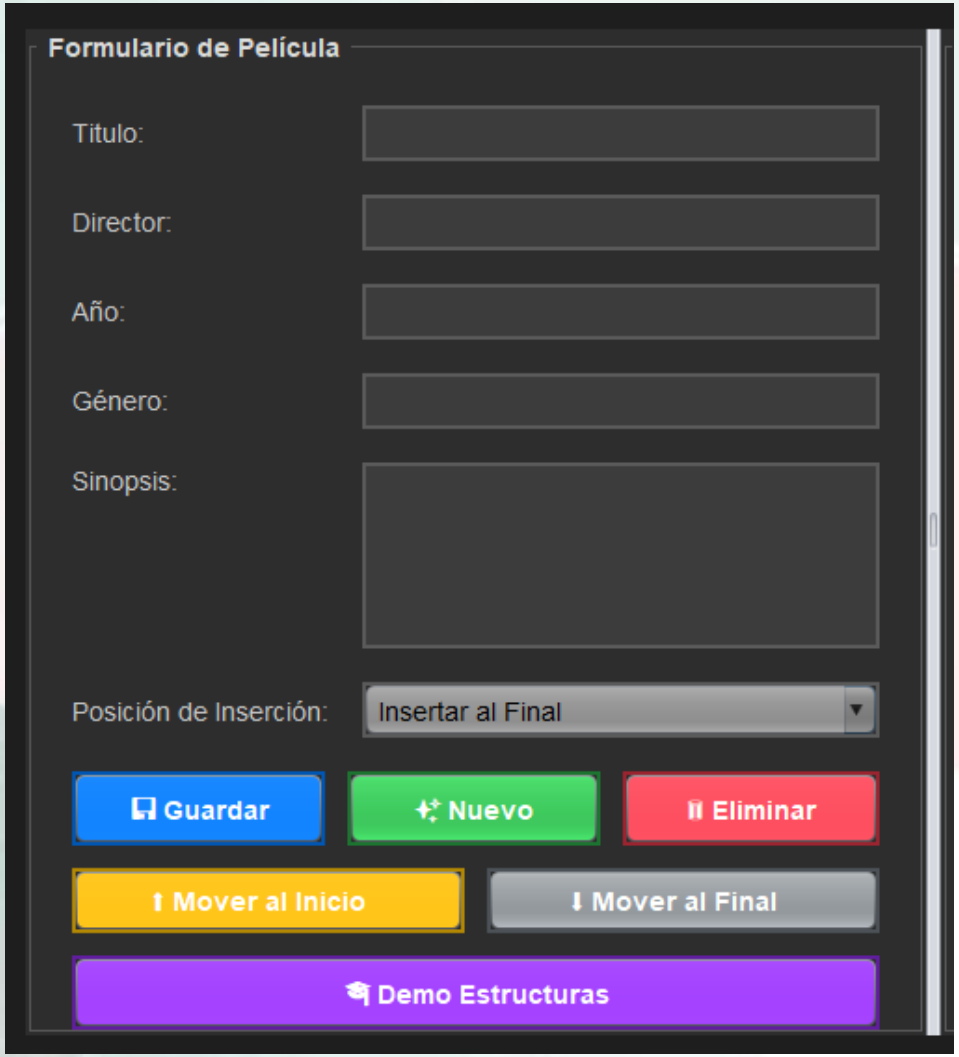
Arquitectura General del Sistema
El sistema sigue el patrón Modelo-Vista-Controlador (MVC) con una clara separación de responsabilidades.

Modelo (Package Modelo)
Pelicula.java: Clase entidad con validaciones integradas para cada atributo

- ListaEnlazada.java: Implementación de lista enlazada simple con operaciones de inserción, eliminación y búsqueda.
- Pila.java & Cola.java: Estructuras LIFO y FIFO para gestionar películas vistas recientemente y por ver.
- PersistenciaPeliculasJSON.java: Módulo central que maneja la serialización JSON y el algoritmo de Mezcla Natural.



Desarrollo:



Vista (Package Vista)
VentanaPrincipal.java: Contenedor principal que organiza los paneles.

- PanelFormulario.java: Formulario para entrada y edición de datos.
- PanelLista.java: Tabla con funcionalidades de búsqueda y ordenamiento.
- DialogoDemoEstructuras.java: Ventana modal para demostración interactiva.

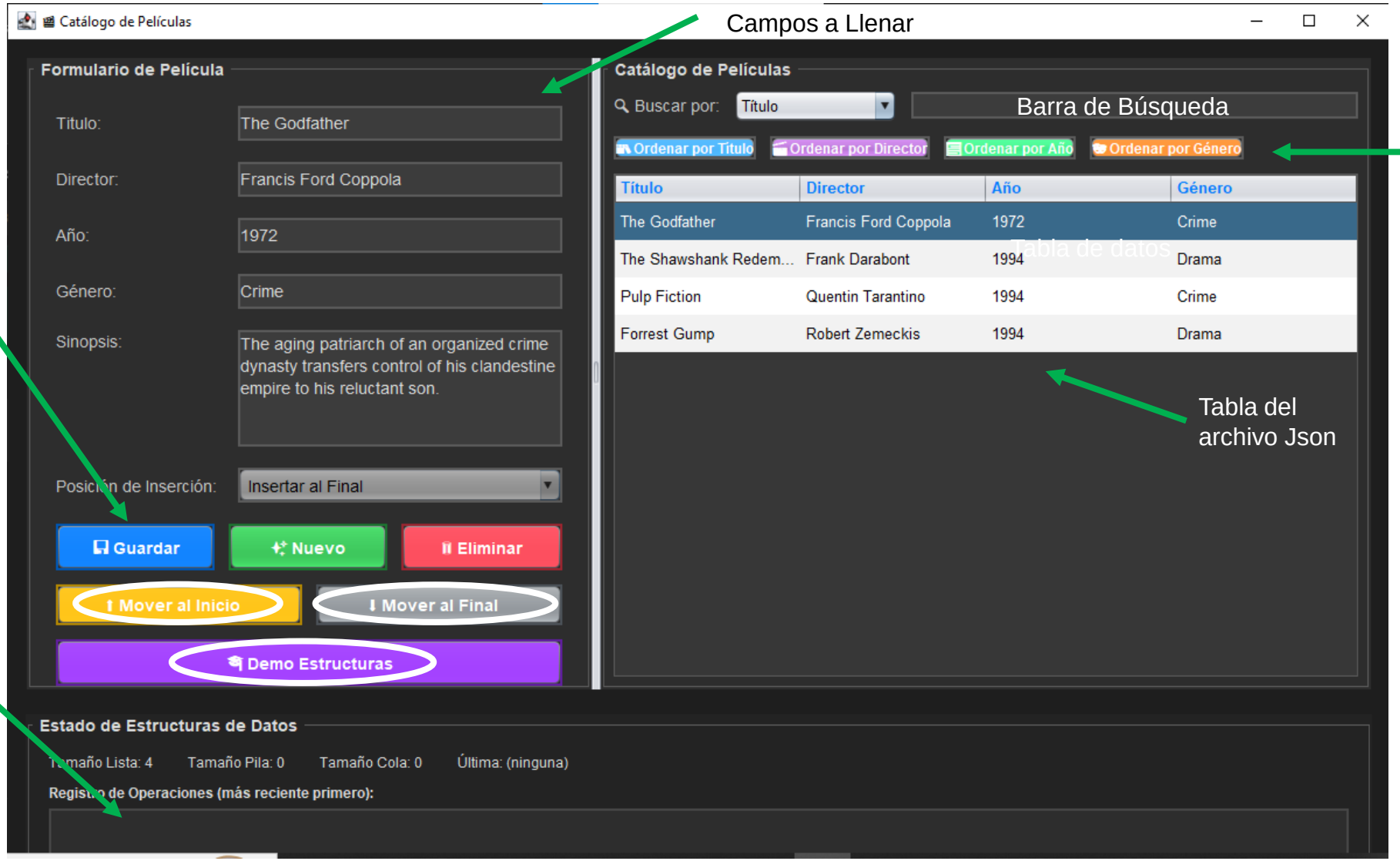
Controlador (Package Controlador)
ControladorPelicula.java: Coordina todas las interacciones entre modelo y vista, gestionando eventos y flujo de datos.

Desarrollo (Código del ordenamiento):

```java
private static void mezclaNatural(Path origen, Comparator<Pelicula> comparador) throws IOException {
    Path archivoA = origen.resolveSibling("tmp_a.jsonl");
    Path archivoB = origen.resolveSibling("tmp_b.jsonl");
    Path archivoSalida = origen.resolveSibling("tmp_out.jsonl");

    while (true) {
        int corridas = dividirEnCorridas(origen, archivoA, archivoB, comparador);
        if (corridas <= 1) {
            // Ya está ordenado en origen
            Files.deleteIfExists(archivoA);
            Files.deleteIfExists(archivoB);
            Files.deleteIfExists(archivoSalida);
            return;
        }
        fusionarCorridas(archivoA, archivoB, archivoSalida, comparador);
        Files.deleteIfExists(origen);
        Files.move(archivoSalida, origen, StandardCopyOption.REPLACE_EXISTING, StandardCopyOption.ATOMIC_MOVE);
    }
}
```

Ejecución:



Operaciones de lista

Registro de acciones

Ordenamiento (Mezcla Natural)

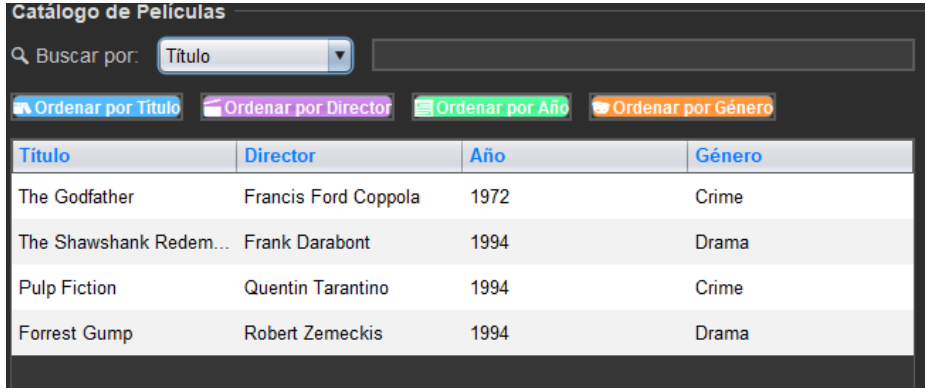
Tabla del archivo Json

Ejecución:


Demo de Estructuras de Datos - Trabajando con Datos Reales
Catálogo (ListaEnlazada)
The Godfather (1972)
The Shawshank Redemption (1994)
Pulp Fiction (1994)
Forrest Gump (1994)
Vistas Recientemente (Pila LIFO)
The Godfather (1972)
The Shawshank Redemption (1994)
The Shawshank Redemption (1994)
Cola Por Ver (Cola FIFO)
The Shawshank Redemption (1994)
Marcar como Vista
Agregar a Por Ver
Eliminar del Catálogo
Ver Más Reciente Otra Vez
Limpiar Historial
Ver Siguiente Película
Remover Siguiente de Cola
Limpiar Toda la Cola
Lista: 4
Pila: 3
Cola: 1
[20:28:11] CATALOGO → Cola: 'The Shawshank Redemption' agregada a por ver (se verá después)
[20:28:14] Cola → Pila: Vista 'The Shawshank Redemption' (FIFO desencolar → agregada al historial)
[20:28:19] CATÁLOGO → Pila: 'The Godfather' marcada como vista (agregada a vistas recientemente)

Ejecución:

Tabla Normal



Título	Director	Año	Género
The Godfather	Francis Ford Coppola	1972	Crime
The Shawshank Redem...	Frank Darabont	1994	Drama
Pulp Fiction	Quentin Tarantino	1994	Crime
Forrest Gump	Robert Zemeckis	1994	Drama

Tabla Ordenada por título

Título	Director	Año	Género
Forrest Gump	Robert Zemeckis	1994	Drama
Pulp Fiction	Quentin Tarantino	1994	Crime
The Godfather	Francis Ford Coppola	1972	Crime
The Shawshank Redem...	Frank Darabont	1994	Drama

Tabla Ordenada por director

Título	Director	Año	Género
The Godfather	Francis Ford Coppola	1972	Crime
The Shawshank Redem...	Frank Darabont	1994	Drama
Pulp Fiction	Quentin Tarantino	1994	Crime
Forrest Gump	Robert Zemeckis	1994	Drama

Por año Por Género

Título	Director	Año	Género
The Godfather	Francis Ford Coppola	1972	Crime
Pulp Fiction	Quentin Tarantino	1994	Crime
Forrest Gump	Robert Zemeckis	1994	Drama
The Shawshank Redem...	Frank Darabont	1994	Drama

Tabla de Complejidad Algorítmica

Tamaño Catálogo	Mezcla Natural (seg)	Timsort Memoria (seg)	Memoria Usada	Archivos Temp
10	0.02	0.001	<1 MB	3 × 0.5 KB
100	0.15	0.02	2 MB	3 × 5 KB
1	0.45	0.08	15 MB	3 × 50 KB
10	2.80	0.35	120 MB	3 × 500 KB
100	32.50	4.20	1.1 GB	3 × 5 MB
500	180.00	28.50	5.5 GB	3 × 25 MB
1,000,000	425.00	OutOfMemoryError	N/A	3 × 50 MB

Conclusiones:

La implementación del algoritmo de Mezcla Natural sobre archivos JSON provee una solución escalable para el ordenamiento de grandes catálogos que no cabrían completamente en memoria RAM, manteniendo un balance óptimo entre eficiencia temporal y uso de recursos.

La elección de JSON como formato de almacenamiento demostró ser acertada, combinando la legibilidad de JSON con la eficiencia del procesamiento secuencial. Los datos se guardan automáticamente y no se pierden al cerrar el programa.

La integración de múltiples estructuras de datos (Lista, Pila, Cola) en una aplicación cohesiva proporciona un ejemplo práctico de cómo estas estructuras fundamentales pueden cooperar para resolver problemas del mundo real.

Recomendaciones:

Manejar mejor los errores cuando el archivo de datos se daña o no se puede leer

Validar que no se repitan películas con el mismo título al editarlas

Guardar solo lo que cambió en lugar de todo el archivo cada vez

Sistema de categorías para clasificar por series, sagas o colecciones

Separar el código de ordenamiento en su propia clase

ESPE
UNIVERSIDAD DE LAS FUERZAS ARMADAS
INNOVACIÓN PARA LA EXCELENCIA

GRACIAS POR

SU ATENCION