



TAREA 1

⌚ Enunciado

Dispones de información de alumnado de **1º DAW** y **2º DAM** en dos ficheros de texto. Crea un proyecto Java llamado **tarea1** con groupId **es.cifpcarlos3** que::

1. **Lee** ambos ficheros, con las clases mínimas necesarias, y **filtre** el alumnado para **incluir solo** quienes sean de la ciudad de **Cartagena**. Se quiere guardar información del alumnado de cada curso. De cada alumno se registrará la siguiente información: nombre, apellidos, edad y fechaRegistro (fecha en que se guardan los datos).

2. **Genera y guarda** la información resultante en **tres formatos**:

- **Binario** (serialización Java, archivo dat).
- **JSON**.
- **XML**.

Todos estos ficheros deben guardarse dentro de una carpeta **salida en la raíz del proyecto**. Si la carpeta no existe, el programa debe **crear** dicha carpeta.

3. **Valida** el proceso **leyendo de nuevo** las salidas generadas (binario/JSON/XML) y mostrando los datos por consola de manera coherente.

No habrá datos añadidos “a mano”: **únicamente** se trabaja con los dos ficheros de entrada.



Datos de entrada

Guarda los siguientes ficheros en la **raíz del proyecto**:

- `lista_alumnado_DAM2.txt`
- `lista_alumnado_DAW1.csv`

Solo se deben considerar las personas cuya ciudad sea Cartagena (ignorando espacios en blanco y líneas vacías).

> Salidas requeridas (carpeta salida)

- **Binario:** p. ej., `cursos.dat` (lista de cursos y su alumnado filtrado).
- **JSON conjunto:** p. ej., `cursos.json`.
- **XML conjunto:** p. ej., `cursos.xml`.
- **(Opcional) JSON por curso:** p. ej., `dam2.json`, `daw1.json`.
- **(Opcional) XML por curso:** p. ej., `dam2.xml`, `daw1.xml`.

Una vez generados los archivos de salida, el programa debe leer de nuevo (al menos) `cursos.dat`, `cursos.json` y `cursos.xml` para validar la persistencia.

Requisitos técnicos obligatorios

- **Java:** Oracle OpenJDK **21 o 25** (≥ 17).
- **Maven:** gestor de dependencias. Usa las **dependencias mínimas** necesarias.
- **Lombok:** para getters/setters/constructores.
- **Jackson 3:** para **JSON/XML**.
 - *Nota:* en Jackson 3, los **mappers** (JsonMapper, XmlMapper) están en `tools.jackson.*`.
 - Las **anotaciones JSON** (@JsonRootName, @JsonFormat) permanecen en `com.fasterxml.jackson.annotation.*`.
 - Las **anotaciones XML** (@JacksonXmlElementWrapper, @JacksonXmlProperty) están en `tools.jackson.dataformat.xml.annotation.*`.



- **I/O:** `java.io` o `java.nio` + `try-with-resources` en todas las aperturas de flujos y canales (incluida lectura de texto y escritura/lectura de binario/JSON/XML).
- **Paquete:** todas las clases dentro de `es.cifpcarlos3`.
- **Carpeta de salida:** `salida` en la carpeta del paquete; se **crea automáticamente** si no existe.
- **Codificación de texto: UTF-8**.
- **Parser de líneas:** uso de separador simple (no usar librerías de “alto nivel” para CSV).

⇒ Consideraciones de diseño

La validación por consola debe permitir comprobar que los datos filtrados coinciden en lo esencial (cursos, alumnos por curso y atributos de cada uno: nombre, apellidos, edad y fechaRegistro).

└─ Entrega

- Proyecto **Maven** completo (`pom.xml` + fuentes en `src/main/java`).
- Ficheros de entrada en la **raíz** del proyecto.
- Carpeta `salida` con todas las salidas generadas (o reproducibles inmediatamente ejecutando el programa).
- El proyecto debe **compilar y ejecutarse** sin errores en el entorno indicado.
- **Opcional:** Subir el proyecto a **GitHub**.