

Materia:	Programación II		
Nivel:	2º Cuatrimestre		
Tipo de Examen:	Recuperatorio de Segundo Parcial		
Apellido ⁽¹⁾ :		Fecha:	5 dic 2025
Nombre/s ⁽¹⁾ :		Docente a cargo ⁽²⁾ :	Baus / Rocha
División ⁽¹⁾ :	122-2	Nota ⁽²⁾ :	
DNI ⁽¹⁾ :		Firma ⁽²⁾ :	

(1) Campos a completar solo por el estudiante en caso de imprimir este enunciado en papel.

(2) Campos a completar solo por el docente en caso de imprimir este enunciado en papel.

Sistema de Gestión de Criaturas del Parque Jurásico

El Parque Jurásico administra diversas **criaturas prehistóricas** que habitan en zonas controladas del parque. Se necesita un sistema informático que permita **registrar, organizar, buscar, ordenar, transformar y exportar datos** sobre estas criaturas.

El objetivo es implementar un subsistema de gestión utilizando POO, genéricos, interfaces funcionales, ordenamientos y persistencia en archivos, sin usar Streams.

1. Interfaz Almacenable<T>

Crear una interfaz genérica `Almacenable<T>` que represente la capacidad de almacenar y administrar elementos.

Debe declarar los siguientes métodos:

Operaciones básicas

- `void agregar(T elemento);`
- `void eliminarSegun(Predicate<T> criterio);`
- `List<T> obtenerTodos();`
- `T buscar(Predicate<T> criterio);`

Ordenamientos

- `void ordenar();`
- `void ordenar(Comparator<T> comparador);`

Procesamiento

- `List<T> filtrar(Predicate<T> criterio);`
- `List<T> transformar(Function<T,T> operador);`

- `int contar(Predicate<T> criterio);`

Persistencia

- `void guardarEnBinario(String ruta) throws Exception;`
- `void cargarDesdeBinario(String ruta) throws Exception;`
- `void guardarEnCSV(String ruta) throws Exception;`
- `void cargarDesdeCSV(String ruta, Function<String,T> fromCSV) throws Exception;`
- `void guardarEnJSON(String ruta) throws Exception;`

2. Clase genérica Inventario<T>

Implementar una clase `Inventario<T>` que:

- Implemente `Almacenable<T>`
- Utilice internamente una `List<T>`
- Implemente **todos los métodos** de la interfaz
- Utilice solamente bucles clásicos (`for`, `while`)
- Maneje archivos:
 - Binario (`ObjectOutputStream` / `ObjectInputStream`)
 - CSV (con separador **coma**)
 - JSON con formato legible (**pretty printing**, usando `Gson`)

3. Enum Especie

Crear el enum:

TREX, VELOCIRAPTOR, TRICERATOPS, BRONTOSAURIO, PTERODACTILO

4. Clase CriaturaJurásica

Debe incluir:

Atributos

- `int id`
- `String nombre`
- `Especie especie`
- `int nivelPeligrosidad (1 a 10)`
- `int añoDescubrimiento`

Comportamiento requerido

- Constructor completo
- Getters
- `toString()` adecuado

- Implementar CSVConvertible
- String toCSV() separando por coma
- static String toHeaderCSV()
- id,nombre,especie,nivelPeligrosidad,anioDescubrimiento
- static CriaturaJurasica fromCSV(String linea)
- Implementar Comparable<CriaturaJurasica> con el siguiente orden natural:

Orden natural solicitado

1. Mayor nivel de peligrosidad primero
2. A igual peligrosidad, especie con **año de descubrimiento más antiguo** primero (menor año)

5. Programa Principal (Tester)

El método `main` debe cumplir estos pasos, en este orden exacto:

1. Crear un inventario de `CriaturaJurasica`.
2. Agregar **al menos seis** criaturas de distintas especies y niveles de peligrosidad.
3. Mostrar todas las criaturas.
4. Ordenarlas usando el **orden natural**.
5. Ordenarlas por **nombre alfabético** → completar `Comparator`.
6. Filtrar criaturas con **nivelPeligrosidad** ≥ 7 → completar `Predicate`.
7. Transformar criaturas de especie **VELOCIRAPTOR**:
 - duplicar su peligrosidad +1
 - máximo permitido = 10
 - completar `Function`.
8. Contar cuántas criaturas fueron descubiertas **antes de 1900** → completar `Predicate`.
9. Guardar el inventario en:
 - archivo binario
 - CSV
 - JSON
10. Crear un segundo inventario y cargar datos desde CSV usando `fromCSV`.
11. Mostrar el inventario cargado desde CSV.

```
public static void main(String[] args) {
    try {
        Almacenable<CriaturaJurasica> inv = new Inventario<>();

        inv.agregar(new CriaturaJurasica(1, "Rexy", Especie.TREX, 10, 1905));
        inv.agregar(new CriaturaJurasica(2, "Blue", Especie.VELOCIRAPTOR, 7, 1924));
        inv.agregar(new CriaturaJurasica(3, "Spike", Especie.TRICERATOPS, 5, 1889));
        inv.agregar(new CriaturaJurasica(4, "Bronte", Especie.BRONTOSAURIO, 3, 1878));
        inv.agregar(new CriaturaJurasica(5, "SkyWing", Especie.PTERODACTILO, 6, 1911));
        inv.agregar(new CriaturaJurasica(6, "SwiftClaw", Especie.VELOCIRAPTOR, 8,
1930));

        System.out.println("=== Inventario original ===");
        for (CriaturaJurasica c : inv.obtenerTodos()) {
            System.out.println(c);
        }
    }
}
```

```
// 1) Orden natural (peligrosidad descendente)
inv.ordenar();
System.out.println("\n=== Orden natural ===");
for (CriaturaJurasica c : inv.obtenerTodos()) {
    System.out.println(c);
}

// 2) Ordenar por nombre → completar Comparator
inv.ordenar(/* completar Comparator por nombre */);
System.out.println("\n=== Ordenadas por nombre ===");
for (CriaturaJurasica c : inv.obtenerTodos()) {
    System.out.println(c);
}

// 3) Filtrar peligrosidad >= 7 → completar Predicate
System.out.println("\n=== Peligrosidad >= 7 ===");
List<CriaturaJurasica> filtradas =
    inv.filtrar(/* completar Predicate */);
for (CriaturaJurasica c : filtradas) {
    System.out.println(c);
}

// 4) Transformación: VELOCIRAPTOR con peligrosidad duplicada (+1, máx 10)
System.out.println("\n=== Transformación de Velociraptors ===");
List<CriaturaJurasica> transformadas =
    inv.transformar(/* completar Function */);
for (CriaturaJurasica c : transformadas) {
    System.out.println(c);
}

// 5) Contar criaturas descubiertas antes de 1900 → completar Predicate
int antiguas = inv.contar(/* completar Predicate */);
System.out.println("\nDescubiertas antes de 1900: " + antiguas);

// 6) Persistencia
inv.guardarEnBinario("data/jurasico.bin");
inv.guardarEnCSV("data/jurasico.csv");
inv.guardarEnJSON("data/jurasico.json");

// 7) Cargar desde CSV → completar función fromCSV
Almacenable<CriaturaJurasica> invCSV = new Inventario<>();
invCSV.cargarDesdeCSV("data/jurasico.csv",
    /* completar función fromCSV */);

System.out.println("\n=== Inventario cargado desde CSV ===");
for (CriaturaJurasica c : invCSV.obtenerTodos()) {
    System.out.println(c);
}

} catch (Exception e) {
    System.err.println("Error: " + e.getMessage());
}
}
```