

ACTIVIDAD 1.4

EXPORTACIÓN DE DATOS



Iris Pérez Aparicio
26/10/2025

Campus FP Emprende Humanes
2º DAM
Acceso a datos

ÍNDICE

ESTRUCTURA FINAL	7
ÚTILES	8
NIVEL 1: BÁSICO - Lista de Estudiantes	11
ESTRUCTURA	11
Tareas	13
NIVEL 2: INTERMEDIO - Biblioteca de	28
ESTRUCTURA	28
Tareas	31
NIVEL 3: AVANZADO - Sistema de	52
ESTRUCTURA	52
Tareas	56
EJECUCIÓN	87

Enlace al proyecto en [GITHUB](#)

(https://github.com/IrisCampusFP/ActividadesAccesoADatos/tree/main/UD1-Persistencia_en_Ficheros/Act1.4-Exportacion_de_datos/Act1.4-Exportacion_de_datos)

ESTRUCTURA FINAL

```
✓  Act1.4-Exportacion_de_datos W:\REPOS\acceso_a
    >  .idea
    >  exportaciones
    >  out
    ✓  src
        ✓  nivel1Basico_listaDeEstudiantes
            ✓  exportadores
                ⓒ ExportadorEstudiantesCSV
                ⓒ ExportadorEstudiantesJSON
                ⓒ ExportadorEstudiantesXML
            ✓  modelo
                ⓒ Estudiante
                ⓒ MainEstudiantes
        ✓  nivel2Intermedio_bibliotecaLibros
            ✓  exportadores
                ⓒ ExportadorLibrosCSV
                ⓒ ExportadorLibrosJSON
                ⓒ ExportadorLibrosXML
            ✓  modelo
                ⓒ Libro
                ⓒ MainBiblioteca
        ✓  nivel3Avanzado_sistemaReservasHotel
            ✓  exportadores
                ⓒ ExportadorReservasCSV
                ⓒ ExportadorReservasJSON
                ⓒ ExportadorReservasXML
            ✓  modelo
                ⓒ Cliente
                ⓒ Habitacion
                ⓒ Reserva
                ⓒ MainReservasHotel
            ✓  utils
                ⓒ Utils
                ⓒ Main
        Ⓢ .gitignore
    □ Act1.4-Exportacion_de_datos.iml
```

ÚTILES

utils.Utils.java

```
package utils;

import java.io.*;
import java.util.InputMismatchException;
import java.util.Locale;
import java.util.Scanner;

public class Utils {
    public final static Scanner sc = new Scanner(System.in);

    // CONTROL DE ENTRADA DE USUARIO:

    public static int leerEntero() {
        try {
            int num = sc.nextInt();
            sc.nextLine(); // Limpiar buffer
            return num;
        } catch (InputMismatchException e) {
            sc.nextLine(); // Limpiar buffer
            System.out.print("Por favor, introduce un número entero: ");
            return leerEntero();
        }
    }

    // PARA MENÚS:

    public static int leerOpcion() {
        System.out.print("↳ Seleccione una opción: ");
        return leerEntero();
    }

    // (Pausa el menú hasta que el usuario presione ENTER)
    public static void pausar() {
        System.out.print("\nPresiona ENTER para continuar...");
        sc.nextLine();
    }

    public static void opcionInvalida() {
        System.out.println("Opción inválida. Introduce el número
correspondiente a la opción que quieras seleccionar.");
    }

    // EXPORTACIÓN (FORMATO)

    // Mostrar los valores double con 2 decimales (%.2f) en formato inglés
    // ('.' en vez de ',' para los decimales)
```

```

public static String formatearDouble(Double valorDouble) {
    return String.format(Locale.ENGLISH, "%.2f", valorDouble);
}

// Escapar texto para CSV
public static String escaparCSV(String texto) {
    if (texto == null || texto.isEmpty()) {
        return "";
    }
    // Si contiene el separador, comillas o saltos de línea, debemos
    escapar
    if (texto.contains(";") || texto.contains("\\"") || 
    texto.contains("\n")) {
        // Duplicamos las comillas y encerramos todo entre comillas
        return "\"" + texto.replace("\"\"", "\"\"\\\"\"") + "\"";
    }

    return texto;
}

// Escapar texto para JSON
public static String escaparJSON(String texto) {
    if (texto == null || texto.isEmpty()) {
        return "";
    }
    // IMPORTANTE: El orden importa - escapar \ primero
    return texto.replace("\\\", \"\\\\\\\") // Barra invertida primero
        .replace("\\"", "\\\\"") // Comillas dobles
        .replace("\n", "\\\n") // Nueva línea
        .replace("\r", "\\\r") // Retorno de carro
        .replace("\t", "\\\t"); // Tabulador
}

// Escapar texto para XML
public static String escaparXML(String texto) {
    if (texto == null || texto.isEmpty()) {
        return "";
    }

    // (El orden importa - escapar & primero)
    return texto.replace("&", "&")
        .replace("<", "<")
        .replace(">", ">")
        .replace("\\"", """)
        .replace("\'", "'");
}

// ARCHIVOS

// Crear un directorio si no existe
public static boolean crearDirectorio(String DIRECTORIO) {
    try {

```

```

        File directorio = new File(DIRECTORIO);
        // Si no existe el directorio
        if (!directorio.exists()) {
            return directorio.mkdirs(); // Se crea el directorio y
devuelve true si se crea, false si no
        }
        return true;
    } catch (Exception e) {
        System.out.println("Error al crear el directorio '" + DIRECTORIO
+ "': " + e.getMessage());
        return false;
    }
}

```

NIVEL 1: BÁSICO - Lista de Estudiantes

Descripción

Crea un sistema simple para exportar una lista de estudiantes con sus calificaciones.

ESTRUCTURA



Estructura de datos

```
class Estudiante {  
    - id: int  
    - nombre: String  
    - apellidos: String  
    - edad: int  
    - nota: double  
}
```

Estudiante.java

```
package nivellBasico_listadeEstudiantes.modelo;  
  
public class Estudiante {  
    private int id;  
    private String nombre;  
    private String apellidos;  
    private int edad;  
    private double nota;  
  
    public Estudiante(int id, String nombre, String apellidos, int  
edad, double nota) {  
        this.id = id;  
        this.nombre = nombre;  
        this.apellidos = apellidos;  
        this.edad = edad;  
        this.nota = nota;  
    }  
  
    public int getId() {  
        return id;  
    }  
  
    public String getNombre() {  
        return nombre;  
    }  
  
    public String getApellidos() {  
        return apellidos;  
    }  
}
```

```
public int getEdad() {
    return edad;
}

public double getNota() {
    return nota;
}
}
```

Tareas

1.1. Exportar a CSV

Crear archivo `estudiantes.csv`

Incluir encabezado: `ID;Nombre;Apellidos;Edad;Nota`

Exportar 5 estudiantes de ejemplo

Añadir línea de resumen con la nota media

Ejemplo de salida:

```
CSV
ID;Nombre;Apellidos;Edad;Nota
1;Juan;García López;20;8.5
2;María;Rodríguez;19;9.2
3;Pedro;Martínez;21;7.8
4;Ana;López;20;8.9
5;Carlos;Sánchez;22;6.5

# Nota media;8.18
```

ExportadorEstudiantesCSV.java

```
package nivellBasico_listaDeEstudiantes.exportadores;

import nivellBasico_listaDeEstudiantes.modelo.Estudiante;
import java.io.*;
import java.text.SimpleDateFormat;
import java.util.ArrayList;
```

```

import java.util.Date;

import static utils.Utils.*;

public class ExportadorEstudiantesCSV {
    static final String CARPETA = "exportaciones";
    static String fecha = new
SimpleDateFormat("yyyyMMdd_HHmss").format(new Date());
    private static final String SEPARADOR = ";" // Separador CSV

    public static boolean exportar(ArrayList<Estudiante>
estudiantes, String nombreArchivo) {

        // VALIDACIONES

        if (estudiantes == null || estudiantes.isEmpty()) {
            System.out.println("ERROR: No hay elementos para
exportar.");
            return false;
        }

        String rutaCompleta = CARPETA + File.separator +
nombreArchivo + "_" + fecha + ".csv";

        if (rutaCompleta == null || rutaCompleta.trim().isEmpty())
{
            System.out.println("ERROR: El nombre del archivo no
puede estar vacío.");
            return false;
        }

        // CREAR DIRECTORIO

        if (crearDirectorio(CARPETA)) {
            try (BufferedWriter writer = new BufferedWriter(new
FileWriter(rutaCompleta))) {

                // 1. ESCRIBIR ENCABEZADO
                escribirEncabezado(writer);

                // 2. ESCRIBIR CADA MOVIMIENTO
                for (Estudiante e : estudiantes) {
                    escribirEstudiante(writer, e);
                }

                // 3. ESCRIBIR RESUMEN
                escribirResumen(writer, estudiantes);

            }
            return true;
        } catch (IOException e) {
            System.out.println("Error al escribir el archivo
CSV: " + e.getMessage());
        }
    }
}

```

```

        }
    }
    return false; // (no se ha creado el directorio)
}

// Encabezado (nombres de las columnas)
private static void escribirEncabezado( BufferedWriter writer)
throws IOException {
    writer.write("ID" + SEPARADOR);
    writer.write("Nombre" + SEPARADOR);
    writer.write("Apellidos" + SEPARADOR);
    writer.write("Edad" + SEPARADOR);
    writer.write("Nota"); // El último elemento SIN SEPARADOR
    writer.newLine(); // Salto de línea al final
}

// Escribe todos los campos de cada estudiante
private static void escribirEstudiante( BufferedWriter writer,
Estudiante e) throws IOException {
    // Cada campo separado por el delimitador
    writer.write(e.getId() + SEPARADOR);
    writer.write(escaparCSV(e.getNombre()) + SEPARADOR);
    writer.write(escaparCSV(e.getApellidos()) + SEPARADOR);
    writer.write(e.getEdad() + SEPARADOR);
    writer.write(formatearDouble(e.getNota())); // El último
elemento SIN SEPARADOR
    writer.newLine(); // Salto de línea al final
}

// Escribe el Resumen
private static void escribirResumen( BufferedWriter writer,
ArrayList<Estudiante> estudiantes) throws IOException {
    // Cálculos
    double sumaNotas = 0;
    for (Estudiante e : estudiantes) {
        sumaNotas += e.getNota();
    }

    double notaMedia = sumaNotas / estudiantes.size(); // Suma
de todas las notas entre el número de estudiantes

    writer.newLine(); // Línea en blanco (Para separar el
resumen de los elementos de arriba)

    writer.write("# Nota media" + SEPARADOR +
formatearDouble(notaMedia));
    writer.newLine();
}
}

```

1.2. Exportar a XML

Crear archivo `estudiantes.xml`

Estructura jerárquica clara

Incluir metadata (fecha, total)

Añadir resumen con estadísticas

Ejemplo de salida:

```
<?xml version="1.0" encoding="UTF-8"?>
<clase>
  <metadata>
    <fecha>19/10/2025</fecha>
    <totalEstudiantes>5</totalEstudiantes>
  </metadata>
  <estudiantes>
    <estudiante id="1">
      <nombre>Juan</nombre>
      <apellidos>García López</apellidos>
      <edad>20</edad>
      <nota>8.5</nota>
    </estudiante>
    ...
  </estudiantes>
  <resumen>
    <notaMedia>8.18</notaMedia>
    <notaMaxima>9.2</notaMaxima>
    <notaMinima>6.5</notaMinima>
  </resumen>
</clase>
```

ExportadorEstudiantesXML.java

```
package nivellBasico_listaDeEstudiantes.exportadores;

import nivellBasico_listaDeEstudiantes.modelo.Estudiante;

import java.io.*;
import java.text.SimpleDateFormat;
import java.time.LocalDateTime;
import java.time.format.DateTimeFormatter;
import java.util.ArrayList;
import java.util.Date;
```

```

import static utils.Utils.*;

public class ExportadorEstudiantesXML {
    static final String CARPETA = "exportaciones";
    static String fecha = new
SimpleDateFormat("yyyyMMdd_HH:mm:ss").format(new Date()));

    private static final String INDENTACION = "    ";
    private static final String INDENTACION2 = INDENTACION +
INDENTACION;
    private static final String INDENTACION3 = INDENTACION2 +
INDENTACION;

    private static final String NODOPADRE = "clase";
    private static final String NODOHIJO = "estudiantes";

    public static boolean exportar(ArrayList<Estudiante>
estudiantes, String nombreArchivo) {
        try {

            // VALIDACIONES

            if (estudiantes == null || estudiantes.isEmpty()) {
                System.out.println("ERROR: No hay elementos para
exportar.");
                return false;
            }

            String rutaCompleta = CARPETA + File.separator +
nombreArchivo + "_" + fecha + ".xml";

            if (rutaCompleta == null ||
rutaCompleta.trim().isEmpty()) {
                System.out.println("ERROR: El nombre del archivo
no puede estar vacío.");
                return false;
            }

            // CREAR DIRECTORIO

            if (crearDirectorio(CARPETA)) {
                try (BufferedWriter bw = new BufferedWriter(new
FileWriter(rutaCompleta))) {

                    // 1. Declaración XML y elemento raíz
                    bw.write("<?xml version=\"1.0\""

```

```

encoding=\\"UTF-8\\\"?>") ;
        bw.newLine();
        bw.write("<" + NODOPADRE + ">") ;
        bw.newLine();

        // 2. Metadata
        fecha =
LocalDateTime.now().format(DateTimeFormatter.ofPattern("dd/MM/yyyy
y HH:mm:ss")) ; // fecha y hora actual

        bw.write(INDENTACION + "<metadata>") ;
        bw.newLine();
        bw.write(INDENTACION2 +
"<version>1.0</version>") ;
        bw.newLine();
        bw.write(INDENTACION2 + "<fecha>" +
escaparXML(fecha) + "</fecha>") ;
        bw.newLine();
        bw.write(INDENTACION2 + "<totalEstudiantes>" +
estudiantes.size() + "</totalEstudiantes>") ;
        bw.newLine();
        bw.write(INDENTACION + "</metadata>") ;
        bw.newLine();

        // 3. Lista de estudiantes
        bw.write(INDENTACION + "<" + NODOHIJO + ">") ;
        bw.newLine();
        for (Estudiante e : estudiantes) {
            bw.write(INDENTACION2 + "<estudiante
id=\"" + e.getId() + "\">>") ;
            bw.newLine();
            bw.write(INDENTACION3 + "<nombre>" +
escaparXML(e.getNombre()) + "</nombre>") ;
            bw.newLine();
            bw.write(INDENTACION3 + "<apellidos>" +
escaparXML(e.getApellidos()) + "</apellidos>") ;
            bw.newLine();
            bw.write(INDENTACION3 + "<edad>" +
e.getEdad() + "</edad>") ;
            bw.newLine();
            bw.write(INDENTACION3 + "<nota>" +
formatearDouble(e.getNota()) + "</nota>") ;
            bw.newLine();
            bw.write(INDENTACION2 + "</estudiante>") ;
            bw.newLine();

```

```

        }
        bw.write(INDENTACION + "</" + NODOHIJO + ">") ;
        bw.newLine();

        // 4. Resumen

        // Cálculos
        double sumaNotas = 0;
        double notaMaxima =
estudiantes.getFirst().getNota();
        double notaMinima =
estudiantes.getFirst().getNota();
        for (Estudiante e : estudiantes) {
            double nota = e.getNota();
            sumaNotas += nota;
            if (nota > notaMaxima) notaMaxima = nota;
            if (nota < notaMinima) notaMinima = nota;
        }

        double notaMedia = sumaNotas /
estudiantes.size();

        bw.write(INDENTACION + "<resumen>");
        bw.newLine();
        bw.write(INDENTACION2 + "<notaMedia>" +
formatearDouble(notaMedia) + "</notaMedia>");
        bw.newLine();
        bw.write(INDENTACION2 + "<notaMaxima>" +
formatearDouble(notaMaxima) + "</notaMaxima>");
        bw.newLine();
        bw.write(INDENTACION2 + "<notaMinima>" +
formatearDouble(notaMinima) + "</notaMinima>");
        bw.newLine();
        bw.write(INDENTACION + "</resumen>");
        bw.newLine();

        bw.write("</" + NODOPADRE + ">"); // Cierro el
nodo padre
        bw.newLine();
        return true;
    }
}
return false; // (no se ha creado el directorio)
} catch (IOException e) {
    System.err.println("Error al escribir el archivo XML:
" + e.getMessage());
}

```

```
        return false;
    }
}
```

1.3. Exportar a JSON

Crear archivo `estudiantes.json`

Formato pretty print

Incluir array de estudiantes

Añadir objeto de estadísticas

Ejemplo de salida:

```
{
  "clase": {
    "metadata": {
      "fecha": "19/10/2025",
      "totalEstudiantes": 5
    },
    "estudiantes": [
      {
        "id": 1,
        "nombre": "Juan",
        "apellidos": "García López", "edad": 20,
        "nota": 8.5
      },
      ...
    ],
    "estadisticas": {
      "notaMedia": 8.18,
      "notaMaxima": 9.2,
      "notaMinima": 6.5,
      "aprobados": 5,
      "suspensos": 0
    }
  }
}
```

ExportadorEstudiantesJSON.java

```
package nivel2Intermedio_bibliotecaLibros.exportadores;

import nivel2Intermedio_bibliotecaLibros.modelo.Libro;

import java.io.*;
import java.nio.charset.StandardCharsets;
import java.text.SimpleDateFormat;
import java.time.LocalDateTime;
import java.time.format.DateTimeFormatter;
import java.util.*;

import static utils.Utils.*;

public class ExportadorLibrosJSON {
    static final String CARPETA = "exportaciones";
    static String fecha = new SimpleDateFormat("yyyyMMdd_HHmmss").format(new Date());

    private static final String INDENTACION = "  ";
    private static final String INDENTACION2 = INDENTACION + INDENTACION;
    private static final String INDENTACION3 = INDENTACION2 + INDENTACION;
    private static final String INDENTACION4 = INDENTACION3 + INDENTACION;

    public static boolean exportar(ArrayList<Libro> libros, String nombreArchivo) {
        // VALIDACIONES

        if (libros == null || libros.isEmpty()) {
            System.out.println("ERROR: No hay elementos para exportar.");
            return false;
        }

        String rutaCompleta = CARPETA + File.separator + nombreArchivo + "_" + fecha +
                ".json";

        if (rutaCompleta == null || rutaCompleta.trim().isEmpty()) {
            System.out.println("ERROR: El nombre del archivo no puede estar vacío.");
            return false;
        }

        // CREAR DIRECTORIO
        if (crearDirectorio(CARPETA)) {
```

```

// Utilizo OutputStreamWriter y FileOutputStream para aplicar UTF_8
try (BufferedWriter bw = new BufferedWriter(new OutputStreamWriter(new
FileOutputStream(rutaCompleta), StandardCharsets.UTF_8))) {

    // 1. APERTURA DEL OBJETO RAÍZ
    bw.write("{}");
    bw.newLine();
    bw.write(INDENTACION + "\"biblioteca\": {}");
    bw.newLine();

    // 2. INFORMACIÓN GENERAL
    String fechaActual =
LocalDateTime.now().format(DateTimeFormatter.ofPattern("dd/MM/yyyy"));
    int totalLibros = libros.size();

    bw.write(INDENTACION2 + "\"informacion\": {}");
    bw.newLine();
    bw.write(INDENTACION3 + "\"nombre\": \"Biblioteca Municipal\",");
    bw.newLine();
    bw.write(INDENTACION3 + "\"fecha\": \"\" + escaparJSON(fechaActual) + "\",");
    bw.newLine();
    bw.write(INDENTACION3 + "\"totalLibros\": " + totalLibros);
    bw.newLine();
    bw.write(INDENTACION2 + "},");
    bw.newLine();

    // 3. AGRUPAR POR CATEGORÍA
    HashMap<String, ArrayList<Libro>> categorias = agruparPorCategoria(libros);

    bw.write(INDENTACION2 + "\"categorias\": {}");
    bw.newLine();

    // --- CÁLCULOS GLOBALES ---
    int totalCategorias = categorias.size();
    int librosDisponibles = 0;
    int librosPrestados = 0;
    int totalPrestamosHistorico = 0;

    // Para controlar la coma entre categorías
    int categoriasEscritas = 0;

    // 4. ESCRIBIR CADA CATEGORÍA
    for (String categoria : categorias.keySet()) {

```

```

ArrayList<Libro> librosCategoria = categorias.get(categoría);

// --- CÁLCULOS POR CATEGORÍA ---
int totalLibrosCategoria = librosCategoria.size();
int totalPrestamosCategoria = 0;
String libroMasPrestado = "";
int maxPrestamos = -1;

for (Libro l : librosCategoria) {
    if (l.getDisponible()) {
        librosDisponibles++;
    } else {
        librosPrestados++;
    }
    totalPrestamosCategoria += l.getPrestamos();
    totalPrestamosHistorico += l.getPrestamos();

    if (l.getPrestamos() > maxPrestamos) {
        maxPrestamos = l.getPrestamos();
        libroMasPrestado = l.getTitulo();
    }
}

double prestamosMedioCategoria = totalLibrosCategoria > 0 ?
    (double) totalPrestamosCategoria / totalLibrosCategoria : 0;

// ESCRIBIR OBJETO CATEGORÍA
bw.write(INDENTACION3 + "\"" + escaparJSON(categoría) + "\": {" );
bw.newLine();
bw.write(INDENTACION4 + "\"totalLibros\": " + totalLibrosCategoria + ",");
bw.newLine();

// LIBROS DE LA CATEGORÍA (ARRAY)
bw.write(INDENTACION4 + "\"libros\": [");
bw.newLine();
for (int i = 0; i < librosCategoria.size(); i++) {
    Libro l = librosCategoria.get(i);
    bw.write(INDENTACION4 + INDENTACION + "{");
    bw.newLine();
    bw.write(INDENTACION4 + INDENTACION2 + "\"isbn\": \"\" +
escaparJSON(l.getIsbn()) + "\",");
    bw.newLine();
}

```

```

        bw.write(INDENTACION4 + INDENTACION2 + "\"" + "titulo" + ":" + "\"" + +
escaparJSON(l.getTitulo() + "\",\"");
        bw.newLine();
        bw.write(INDENTACION4 + INDENTACION2 + "\"" + "autor" + ":" + "\"" + +
escaparJSON(l.getAutor() + "\",\"");
        bw.newLine();
        bw.write(INDENTACION4 + INDENTACION2 + "\"" + "año" + ":" + " " +
l.getAñoPublicacion() + ",");
        bw.newLine();
        bw.write(INDENTACION4 + INDENTACION2 + "\"" + "paginas" + ":" + " " +
l.getNumPaginas() + ",");
        bw.newLine();
        bw.write(INDENTACION4 + INDENTACION2 + "\"" + "disponible" + ":" + " " +
l.getDisponible() + ",");
        bw.newLine();
        bw.write(INDENTACION4 + INDENTACION2 + "\"" + "prestamos" + ":" + " " +
l.getPrestamos());
        bw.newLine();
        bw.write(INDENTACION4 + INDENTACION + "}" + " " + (i <
librosCategoria.size() - 1 ? "," : ""));
        bw.newLine();
    }
    bw.write(INDENTACION4 + "],");
    bw.newLine();
}

// ESTADÍSTICAS DE LA CATEGORÍA
bw.write(INDENTACION4 + "\"" + "estadisticas" + ":" + "{}");
bw.newLine();
bw.write(INDENTACION4 + INDENTACION + "\"" + "totalPrestamos" + ":" + " " +
totalPrestamosCategoria + ",");
bw.newLine();
bw.write(INDENTACION4 + INDENTACION + "\"" + "prestamosMedio" + ":" + " " +
formatearDouble(prestamosMedioCategoria) + ",");
bw.newLine();
bw.write(INDENTACION4 + INDENTACION + "\"" + "libroMasPrestado" + ":" + "\"" + +
escaparJSON(libroMasPrestado) + "\"");
bw.newLine();
bw.write(INDENTACION4 + "}");
bw.newLine();

// CIERRE DE CATEGORÍA
categoriasEscritas++;

```

```

        bw.write(INDENTACION3 + "}" + (categoriasEscritas < categorias.size() ? "," : ""));
    }

    bw.write(INDENTACION2 + "}", ""); // cierre de "categorias"
    bw.newLine();

    // 5. RESUMEN GLOBAL
    bw.write(INDENTACION2 + "\"resumenGlobal\": {" );
    bw.newLine();
    bw.write(INDENTACION3 + "\"totalCategorias\": " + totalCategorias + ",");
    bw.newLine();
    bw.write(INDENTACION3 + "\"totalLibros\": " + totalLibros + ",");
    bw.newLine();
    bw.write(INDENTACION3 + "\"librosDisponibles\": " + librosDisponibles + ",");
    bw.newLine();
    bw.write(INDENTACION3 + "\"librosPrestados\": " + librosPrestados + ",");
    bw.newLine();
    bw.write(INDENTACION3 + "\"totalPrestamosHistorico\": " +
totalPrestamosHistorico);
    bw.newLine();
    bw.write(INDENTACION2 + "}");
    bw.newLine();

    bw.write(INDENTACION + "}"); // CIERRE DEL OBJETO BIBLIOTECA
    bw.newLine();
    bw.write("}"); // CIERRE DEL OBJETO RAÍZ

    return true;
} catch (FileNotFoundException e) {
    throw new RuntimeException(e);
} catch (IOException e) {
    throw new RuntimeException(e);
}
}

return false; // (no se ha creado el directorio)
}

// Agrupa los libros por categoría usando HashMap
private static HashMap<String, ArrayList<Libro>>
agruparPorCategoria(ArrayList<Libro> libros) {
    HashMap<String, ArrayList<Libro>> categorias = new HashMap<>();

```

```

for (Libro l : libros) {
    String cat = l.getCategoria();
    if (!categorias.containsKey(cat)) {
        categorias.put(cat, new ArrayList<>());
    }
    categorias.get(cat).add(l);
}
return categorias;
}
}

```

MainEstudiantes.java (para ejecutar solo este nivel)

```

package nivellBasico_listaDeEstudiantes;

import java.util.ArrayList;
import nivellBasico_listaDeEstudiantes.exportadores.ExportadorEstudiantesCSV;
import nivellBasico_listaDeEstudiantes.exportadores.ExportadorEstudiantesJSON;
import nivellBasico_listaDeEstudiantes.exportadores.ExportadorEstudiantesXML;
import nivellBasico_listaDeEstudiantes.modelo.Estudiante;

import static utils.Utils.*;

public class MainEstudiantes {
    private final static ArrayList<Estudiante> estudiantes = new
ArrayList<>();

    public static void main(String[] args) {

        // Creo 5 estudiantes de ejemplo
        estudiantes.add(new Estudiante(1, "Lucía", "García López",
20, 8.5));
        estudiantes.add(new Estudiante(2, "Carlos", "Martínez
Ruiz", 22, 7.2));
        estudiantes.add(new Estudiante(3, "Ana", "Sánchez
Fernández", 19, 9.0));
        estudiantes.add(new Estudiante(4, "Miguel", "Pérez Gómez",
21, 6.8));
        estudiantes.add(new Estudiante(5, "Sofía", "Hernández

```

```

Díaz", 20, 8.0));

    while (true) {
        System.out.println("\n..... MENÚ PRINCIPAL
.....");
        System.out.println("1. Exportar a CSV");
        System.out.println("2. Exportar a XML");
        System.out.println("3. Exportar a JSON");
        System.out.println("4. Exportar a TODOS los
formatos");
        System.out.println("0. Salir");
        int opcion = leerOpcion();

        System.out.println();
        boolean exito = false;
        String nombreArchivo = "estudiantes";

        switch (opcion) {
            case 1:
                exito =
ExportadorEstudiantesCSV.exportar(estudiantes, nombreArchivo);
                break;
            case 2: exito =
ExportadorEstudiantesXML.exportar(estudiantes, nombreArchivo);
                break;
            case 3: exito =
ExportadorEstudiantesJSON.exportar(estudiantes, nombreArchivo);
                break;
            case 4:
                int exitosos = 0;
                // Exportar CSV
                System.out.println("1/3 Exportando a
CSV...");
                if
(ExportadorEstudiantesCSV.exportar(estudiantes, nombreArchivo)) {
                    exitosos++;
                }
                // Exportar XML
                System.out.println("2/3 Exportando a
XML...");
                if
(ExportadorEstudiantesXML.exportar(estudiantes, nombreArchivo)) {
                    exitosos++;
                }
                // Exportar JSON
                System.out.println("3/3 Exportando a
JSON...");
                if

```

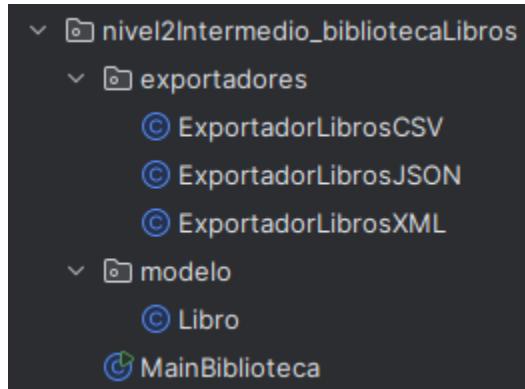
```
(ExportadorEstudiantesJSON.exportar(estudiantes, nombreArchivo))
{
    exitosos++;
}
// Resumen
System.out.println("\nRESUMEN DE
EXPORTACIÓN:");
System.out.println("Formatos exportados: " +
exitosos + "/3\n");
if (exitosos == 3) {
    exito = true;
} else {
    System.out.println("⚠ No se han podido
crear todos los archivos.");
}
break;
case 0: System.out.println("Cerrando
programa...");  

sc.close();
return;
default:
opcionInvalida();
continue;
}
if (exito) {
System.out.println("¡EXPORTACIÓN COMPLETADA
:)!");
System.out.println("(Ubicación:
exportaciones/)");
} else {
System.out.println("No se ha podido completar la
exportación.");
}
pausar();
}
}
```

NIVEL 2: INTERMEDIO - Biblioteca de Libros Descripción

Crea un sistema para exportar el catálogo de una biblioteca con libros organizados por categorías.

ESTRUCTURA



Estructura de datos

```
class Libro {  
    - isbn: String  
    - titulo: String  
    - autor: String  
    - categoria: String  
    - añoPublicacion: int  
    - numPaginas: int  
    - disponible: boolean  
    - prestamos: int  
}
```

Libro.java

```
package nivel2Intermedio_bibliotecaLibros.modelo;  
  
public class Libro {  
    private String isbn;  
    private String titulo;  
    private String autor;  
    private String categoria;  
    private int añoPublicacion;  
    private int numPaginas;  
    private boolean disponible;  
    private int prestamos;
```

```

    // Constructor
    public Libro(String isbn, String titulo, String autor, String
categoria, int añoPublicacion, int numPaginas, boolean
disponible, int prestamos) {
        this.isbn = isbn;
        this.titulo = titulo;
        this.autor = autor;
        this.categoría = categoria;
        this.añoPublicacion = añoPublicacion;
        this.numPaginas = numPaginas;
        this.disponible = disponible;
        this.prestamos = prestamos;
    }

    // Getters
    public String getIsbn() {
        return isbn;
    }

    public String getTitulo() {
        return titulo;
    }

    public String getAutor() {
        return autor;
    }

    public String getCategoría() {
        return categoría;
    }

    public int getAñoPublicacion() {
        return añoPublicacion;
    }

    public int getNumPaginas() {
        return numPaginas;
    }

    public boolean getDisponible() {
        return disponible;
    }

    public int getPrestamos() {
        return prestamos;
    }
}

```

Tareas

2.1. Exportar a CSV con categorías

- Agrupar libros por categoría
- Incluir líneas separadoras entre categorías
- Calcular estadísticas por categoría

Ejemplo de salida:

```
CSV
# BIBLIOTECA MUNICIPAL - CATÁLOGO DE LIBROS

# CATEGORÍA: Ficción
ISBN;Título;Autor;Año;Páginas;Disponible;Préstamos
978-84-123;El Quijote;Miguel de Cervantes;1605;863;true;150
978-84-456;Cien años de soledad;Gabriel García
Márquez;1967;471;false;98

# Subtotal Ficción: 2 libros, 248 préstamos

# CATEGORÍA: Ciencia
...
```

ExportadorLibrosCSV.java

```
package nivel2Intermedio_bibliotecaLibros.exportadores;

import nivel2Intermedio_bibliotecaLibros.modelo.Libro;

import java.io.BufferedReader;
import java.io.File;
import java.io.FileWriter;
import java.io.IOException;
import java.text.SimpleDateFormat;
import java.util.ArrayList;
import java.util.Date;
import java.util.HashMap;
```

```

import static utils.Utils.*;

public class ExportadorLibrosCSV {
    static final String CARPETA = "exportaciones";
    static String fecha = new
SimpleDateFormat("yyyyMMdd_HHmss").format(new Date());
    private static final String SEPARADOR = ";" // Separador CSV

    public static boolean exportar(ArrayList<Libro> libros, String
nombreArchivo) {

        // VALIDACIONES

        if (libros == null || libros.isEmpty()) {
            System.out.println("ERROR: No hay elementos para
exportar.");
            return false;
        }

        String rutaCompleta = CARPETA + File.separator +
nombreArchivo + " " + fecha + ".csv";

        if (rutaCompleta == null || rutaCompleta.trim().isEmpty())
{
            System.out.println("ERROR: El nombre del archivo no
puede estar vacío.");
            return false;
        }

        // CREAR DIRECTORIO

        if (crearDirectorio(CARPETA)) {
            try (BufferedWriter writer = new BufferedWriter(new
FileWriter(rutaCompleta))) {

                // 1. ESCRIBIR ENCABEZADO
                escribirEncabezado(writer);

                // 2. AGRUPAR POR CATEGORÍA
                HashMap<String, ArrayList<Libro>> categorias =
agruparPorCategoria(libros);

                // 3. ESCRIBIR CADA CATEGORÍA
                for (String categoria : categorias.keySet()) {
                    ArrayList<Libro> librosCategoria =

```

```

        categorias.get(categoría);

            escribirSeparadorCategoria(writer, categoría);
            escribirEncabezadoLibros(writer);

            for (Libro libro : librosCategoria) {
                escribirLibro(writer, libro);
            }

            escribirResumenCategoria(writer, categoría,
librosCategoria);

            writer.newLine(); // Línea en blanco entre
categorías
        }

        return true;

    } catch (IOException e) {
        System.out.println("Error al escribir el archivo
CSV: " + e.getMessage());
    }
}

return false;
}

// Encabezado global del catálogo
private static void escribirEncabezado( BufferedWriter writer)
throws IOException {
    writer.write("# BIBLIOTECA MUNICIPAL - CATÁLOGO DE
LIBROS");
    writer.newLine();
    writer.newLine();
}

// Agrupa los libros por categoría usando HashMap
private static HashMap<String, ArrayList<Libro>>
agruparPorCategoria(ArrayList<Libro> libros) {
    HashMap<String, ArrayList<Libro>> categorías = new
HashMap<>();
    for (Libro l : libros) {
        String cat = l.getCategoría();
        if (!categorias.containsKey(cat)) {
            categorías.put(cat, new ArrayList<>());

```

```

        }
        categorias.get(cat).add(l);
    }
    return categorias;
}

// Línea separadora de categoría
private static void escribirSeparadorCategoria(BufferedWriter writer, String categoria) throws IOException {
    writer.write("# CATEGORÍA: " + categoria);
    writer.newLine();
    writer.newLine();
}

// Encabezado (nombres de las columnas de libros)
private static void escribirEncabezadoLibros(BufferedWriter writer) throws IOException {
    writer.write("ISBN" + SEPARADOR);
    writer.write("Título" + SEPARADOR);
    writer.write("Autor" + SEPARADOR);
    writer.write("Año" + SEPARADOR);
    writer.write("Páginas" + SEPARADOR);
    writer.write("Disponible" + SEPARADOR);
    writer.write("Préstamos");
    writer.newLine();
}

// Escribe todos los campos de cada libro
private static void escribirLibro(BufferedWriter writer, Libro l) throws IOException {
    writer.write(escaparCSV(l.getIsbn()) + SEPARADOR);
    writer.write(escaparCSV(l.getTitulo()) + SEPARADOR);
    writer.write(escaparCSV(l.getAutor()) + SEPARADOR);
    writer.write(l.getAñoPublicacion() + SEPARADOR);
    writer.write(l.getNumPaginas() + SEPARADOR);
    writer.write(l.getDisponible() + SEPARADOR);
    writer.write(l.getPrestamos()); // El último elemento SIN SEPARADOR
    writer.newLine();
}

// Escribe el resumen de la categoría
private static void escribirResumenCategoria(BufferedWriter writer, String categoria, ArrayList<Libro> libros) throws

```

```

IOException {
    int totalLibros = libros.size();
    int totalPrestamos = 0;
    for (Libro l : libros) {
        totalPrestamos += l.getPrestamos();
    }

    writer.write("# Subtotal " + categoria + ": " +
totalLibros + " libros, " + totalPrestamos + " préstamos");
    writer.newLine();
}
}

```

2.2. Exportar a XML con estructura compleja

Organizar por categorías (elemento contenedor)

Incluir atributos en elementos

Añadir estadísticas por categoría y globales

Ejemplo de salida:

```

<?xml version="1.0" encoding="UTF-8"?>
<biblioteca>
    <informacion>
        <nombre>Biblioteca Municipal</nombre>
        <fecha>19/10/2025</fecha>
        <totalLibros>50</totalLibros>
    </informacion>

    <categorias>
        <categoria nombre="Ficción" totalLibros="20">
            <libro isbn="978-84-123" disponible="true">
                <titulo>El Quijote</titulo>
                <autor>Miguel de Cervantes</autor>
                <año>1605</año>
                <páginas>863</páginas>
                <prestamos>150</prestamos>
            </libro>
            ...
        <estadísticas>
            <totalPrestamos>1250</totalPrestamos>
            <prestamosMedio>62.5</prestamosMedio>
        </estadísticas>
    </categoria>
</categorias>

```

```

</estadisticas>
</categoria>

<categoria nombre="Ciencia" totalLibros="15"> ...
</categoria>
</categorias>

<resumenGlobal>
<totalCategorias>5</totalCategorias>
<totalLibros>50</totalLibros>
<librosDisponibles>38</librosDisponibles>
<librosPrestados>12</librosPrestados>
</resumenGlobal>
</biblioteca>

```

ExportadorLibrosXML.java

```

package nivel2Intermedio_bibliotecaLibros.exportadores;

import nivel2Intermedio_bibliotecaLibros.modelo.Libro;

import java.io.BufferedReader;
import java.io.File;
import java.io.FileWriter;
import java.io.IOException;
import java.text.SimpleDateFormat;
import java.time.LocalDateTime;
import java.time.format.DateTimeFormatter;
import java.util.ArrayList;
import java.util.Date;
import java.util.HashMap;

import static utils.Utils.*;

public class ExportadorLibrosXML {
    static final String CARPETA = "exportaciones";
    static String fecha = new
SimpleDateFormat("yyyyMMdd_HHmmss").format(new Date());

    private static final String INDENTACION = "    ";
    private static final String INDENTACION2 = INDENTACION +
INDENTACION;
    private static final String INDENTACION3 = INDENTACION2 +
INDENTACION;
    private static final String INDENTACION4 = INDENTACION3 +

```

```


INDENTACION;

    public static boolean exportar(ArrayList<Libro> libros, String
nombreArchivo) {
        try {

            // VALIDACIONES

            if (libros == null || libros.isEmpty()) {
                System.out.println("ERROR: No hay elementos para
exportar.");
                return false;
            }

            String rutaCompleta = CARPETA + File.separator +
nombreArchivo + "_" + fecha + ".xml";

            if (rutaCompleta == null || rutaCompleta.trim().isEmpty())
{
                System.out.println("ERROR: El nombre del archivo no
puede estar vacío.");
                return false;
            }

            // CREAR DIRECTORIO

            if (crearDirectorio(CARPETA)) {
                try (BufferedWriter bw = new BufferedWriter(new
FileWriter(rutaCompleta))) {

                    // 1. Declaración XML y elemento raíz
                    bw.write("<?xml version=\"1.0\""
encoding="UTF-8"?>");
                    bw.newLine();
                    bw.write("<biblioteca>");
                    bw.newLine();

                    // 2. Información general
                    String fechaActual =
LocalDateTime.now().format(DateTimeFormatter.ofPattern("dd/MM/yyyy"));
                    int totalLibros = libros.size();

                    bw.write(INDENTACION + "<informacion>");
                    bw.newLine();
                    bw.write(INDENTACION2 + "<nOMBRE>Biblioteca
Municipal</nOMBRE>");
                }
            }
        }
    }
}


```

```

        bw.newLine();
        bw.write(INDENTACION2 + "<fecha>" +
escaparXML(fechaActual) + "</fecha>");
        bw.newLine();
        bw.write(INDENTACION2 + "<totalLibros>" +
totalLibros + "</totalLibros>");
        bw.newLine();
        bw.write(INDENTACION + "</informacion>");
        bw.newLine();

        // 3. Agrupar por categorías
        HashMap<String, ArrayList<Libro>> categorias =
agruparPorCategoria(libros);

        bw.write(INDENTACION + "<categorias>");
        bw.newLine();

        // Para el resumen global
        int totalCategorias = categorias.size();
        int librosDisponibles = 0;
        int librosPrestados = 0;

        // 4. Escribir cada categoría
        for (String categoria : categorias.keySet()) {
            ArrayList<Libro> librosCategoria =
categorias.get(categoria);

            int totalLibrosCategoria =
librosCategoria.size();
            int totalPrestamosCategoria = 0;
            int disponiblesCategoria = 0;

            for (Libro l : librosCategoria) {
                if (l.getDisponible()) {
                    librosDisponibles++;
                    disponiblesCategoria++;
                } else {
                    librosPrestados++;
                }
                totalPrestamosCategoria +=
l.getPrestamos();
            }

            double prestamosMedioCategoria =
totalLibrosCategoria > 0 ? (double) totalPrestamosCategoria /
totalLibrosCategoria : 0;
        }
    }
}

```

```

        // Inicio categoría
        bw.write(INDENTACION2 + "<categoria nombre=\"" +
+ escaparXML(categoría) + "\" totalLibros=\"" + totalLibrosCategoria +
"\">");

        bw.newLine();

        // Libros de la categoría
        for (Libro libro : librosCategoria) {
            bw.write(INDENTACION3 + "<libro isbn=\"" +
escaparXML(libro.getIsbn()) + "\" disponible=\"" +
libro.getDisponible() + "\">");

            bw.newLine();
            bw.write(INDENTACION4 + "<titulo>" +
escaparXML(libro.getTitulo()) + "</titulo>");
            bw.newLine();
            bw.write(INDENTACION4 + "<autor>" +
escaparXML(libro.getAutor()) + "</autor>");
            bw.newLine();
            bw.write(INDENTACION4 + "<año>" +
libro.getAñoPublicacion() + "</año>");
            bw.newLine();
            bw.write(INDENTACION4 + "<paginas>" +
libro.getNumPaginas() + "</paginas>");
            bw.newLine();
            bw.write(INDENTACION4 + "<prestamos>" +
libro.getPrestamos() + "</prestamos>");
            bw.newLine();
            bw.write(INDENTACION3 + "</libro>");
            bw.newLine();
        }

        // Estadísticas de la categoría
        bw.write(INDENTACION3 + "<estadisticas>");
        bw.newLine();
        bw.write(INDENTACION4 + "<totalPrestamos>" +
totalPrestamosCategoria + "</totalPrestamos>");
        bw.newLine();
        bw.write(INDENTACION4 + "<prestamosMedio>" +
formatearDouble(prestamosMedioCategoria) + "</prestamosMedio>");
        bw.newLine();
        bw.write(INDENTACION3 + "</estadisticas>");
        bw.newLine();

        // Fin categoría
        bw.write(INDENTACION2 + "</categoria>");
        bw.newLine();

```

```

        }

        bw.write(INDENTACION + "</categorias>");

        bw.newLine();

        // 5. Resumen global
        bw.write(INDENTACION + "<resumenGlobal>");
        bw.newLine();
        bw.write(INDENTACION2 + "<totalCategorias>" +
totalCategorias + "</totalCategorias>");
        bw.newLine();
        bw.write(INDENTACION2 + "<totalLibros>" +
totalLibros + "</totalLibros>");
        bw.newLine();
        bw.write(INDENTACION2 + "<librosDisponibles>" +
librosDisponibles + "</librosDisponibles>");
        bw.newLine();
        bw.write(INDENTACION2 + "<librosPrestados>" +
librosPrestados + "</librosPrestados>");
        bw.newLine();
        bw.write(INDENTACION + "</resumenGlobal>");

        bw.newLine();

        // 6. Cerrar raíz
        bw.write("</biblioteca>");
        bw.newLine();
        return true;
    }
}

return false; // (no se ha creado el directorio)
} catch (IOException e) {
    System.err.println("Error al escribir el archivo XML: " +
e.getMessage());
    return false;
}
}

// Agrupa los libros por categoría usando HashMap
private static HashMap<String, ArrayList<Libro>>
agruparPorCategoria(ArrayList<Libro> libros) {
    HashMap<String, ArrayList<Libro>> categorias = new HashMap<>();
    for (Libro l : libros) {
        String cat = l.getCategoría();
        if (!categorias.containsKey(cat)) {
            categorias.put(cat, new ArrayList<>());
        }
    }
}

```

```
        categorias.get(cat).add(l);
    }
    return categorias;
}
}
```

2.3. Exportar a JSON con datos anidados

Usar objetos anidados para categorías
Incluir arrays dentro de objetos
Estadísticas detalladas por categoría

Ejemplo de salida:

```
{  
  "biblioteca": {  
    "informacion": {  
      "nombre": "Biblioteca Municipal",  
      "fecha": "19/10/2025",  
      "totalLibros": 50  
    },  
    "categorias": {  
      "Ficción": {  
        "totalLibros": 20,  
        "libros": [  
          {  
            "isbn": "978-84-123",  
            "titulo": "El Quijote",  
            "autor": "Miguel de Cervantes",  
            "año": 1605,  
            "paginas": 863,  
            "disponible": true,  
            "prestamos": 150  
          },  
          ...  
        ],  
        "estadisticas": {  
          "totalPrestamos": 1250,  
          "prestamosMedio": 62.5,  
          "libroMasPrestado": "El Quijote"  
        }  
      }  
    }  
  }  
}
```

```

        }
    },
    "Ciencia": {
        ...
    },
    "resumenGlobal": {
        "totalCategorias": 5,
        "totalLibros": 50,
        "librosDisponibles": 38,
        "librosPrestados": 12,
        "totalPrestamosHistorico": 3450
    }
}
}
}

```

ExportadorLibrosJSON.java

```

package nivel2Intermedio_bibliotecaLibros.exportadores;

import nivel2Intermedio_bibliotecaLibros.modelo.Libro;

import java.io.*;
import java.nio.charset.StandardCharsets;
import java.text.SimpleDateFormat;
import java.time.LocalDateTime;
import java.time.format.DateTimeFormatter;
import java.util.*;

import static utils.Utils.*;

public class ExportadorLibrosJSON {
    static final String CARPETA = "exportaciones";
    static String fecha = new
SimpleDateFormat("yyyyMMdd_HHmss").format(new Date());

    private static final String INDENTACION = "    ";
    private static final String INDENTACION2 = INDENTACION +
INDENTACION;
    private static final String INDENTACION3 = INDENTACION2 +
INDENTACION;
    private static final String INDENTACION4 = INDENTACION3 +
INDENTACION;

```

```

public static boolean exportar(ArrayList<Libro> libros, String
nombreArchivo) {
    // VALIDACIONES

    if (libros == null || libros.isEmpty()) {
        System.out.println("ERROR: No hay elementos para
exportar.");
        return false;
    }

    String rutaCompleta = CARPETA + File.separator +
nombreArchivo + "_" + fecha + ".json";

    if (rutaCompleta == null || rutaCompleta.trim().isEmpty())
{
        System.out.println("ERROR: El nombre del archivo no
puede estar vacío.");
        return false;
    }

    // CREAR DIRECTORIO
    if (crearDirectorio(CARPETA)) {
        // Utilizo OutputStreamWriter y FileOutputStream para
aplicar UTF_8
        try (BufferedWriter bw = new BufferedWriter(new
OutputStreamWriter(new FileOutputStream(rutaCompleta),
StandardCharsets.UTF_8))) {

            // 1. APERTURA DEL OBJETO RAÍZ
            bw.write("{");
            bw.newLine();
            bw.write(INDENTACION + "\"biblioteca\": {");
            bw.newLine();

            // 2. INFORMACIÓN GENERAL
            String fechaActual =
LocalDateTime.now().format(DateTimeFormatter.ofPattern("dd/MM/yyyy"));
            int totalLibros = libros.size();

            bw.write(INDENTACION2 + "\"informacion\": {");
            bw.newLine();

```

```

        bw.write(INDENTACION3 + "\"nombre\": \"Biblioteca
Municipal\",");
        bw.newLine();
        bw.write(INDENTACION3 + "\"fecha\": \""
+ escaparJSON(fechaActual) + "\",");
        bw.newLine();
        bw.write(INDENTACION3 + "\"totalLibros\": " +
totalLibros);
        bw.newLine();
        bw.write(INDENTACION2 + "},");
        bw.newLine();

        // 3. AGRUPAR POR CATEGORÍA
        HashMap<String, ArrayList<Libro>> categorias =
agruparPorCategoria(libros);

        bw.write(INDENTACION2 + "\"categorias\": {");
        bw.newLine();

        // --- CÁLCULOS GLOBALES ---
        int totalCategorias = categorias.size();
        int librosDisponibles = 0;
        int librosPrestados = 0;
        int totalPrestamosHistorico = 0;

        // Para controlar la coma entre categorías
        int categoriasEscritas = 0;

        // 4. ESCRIBIR CADA CATEGORÍA
        for (String categoria : categorias.keySet()) {
            ArrayList<Libro> librosCategoria =
categorias.get(categoria);

            // --- CÁLCULOS POR CATEGORÍA ---
            int totalLibrosCategoria =
librosCategoria.size();
            int totalPrestamosCategoria = 0;
            String libroMasPrestado = "";
            int maxPrestamos = -1;

            for (Libro l : librosCategoria) {
                if (l.getDisponible()) {

```

```

                librosDisponibles++;
            } else {
                librosPrestados++;
            }
            totalPrestamosCategoria += l.getPrestamos();
            totalPrestamosHistorico += l.getPrestamos();

            if (l.getPrestamos() > maxPrestamos) {
                maxPrestamos = l.getPrestamos();
                libroMasPrestado = l.getTitulo();
            }
        }

        double prestamosMedioCategoria =
totalLibrosCategoria > 0 ?
                    (double) totalPrestamosCategoria /
totalLibrosCategoria : 0;

        // ESCRIBIR OBJETO CATEGORÍA
        bw.write(INDENTACION3 + "\"\n" +
escaparJSON(categoría) + "\": {" );
        bw.newLine();
        bw.write(INDENTACION4 + "\"totalLibros\": " +
totalLibrosCategoria + ",");
        bw.newLine();

        // LIBROS DE LA CATEGORÍA (ARRAY)
        bw.write(INDENTACION4 + "\"libros\": [");
        bw.newLine();
        for (int i = 0; i < librosCategoria.size();
i++) {
            Libro l = librosCategoria.get(i);
            bw.write(INDENTACION4 + INDENTACION +
"{" );
            bw.newLine();
            bw.write(INDENTACION4 + INDENTACION2 +
"\\"isbn\": \"\n" + escaparJSON(l.getIsbn()) + "\",\"");
            bw.newLine();
            bw.write(INDENTACION4 + INDENTACION2 +
"\\"titulo\": \"\n" + escaparJSON(l.getTitulo()) + "\",\"");
            bw.newLine();

```

```

                bw.write(INDENTACION4 + INDENTACION2 +
"\\"autor\": \" + escaparJSON(l.getAutor()) + "\",");
                bw.newLine();
                bw.write(INDENTACION4 + INDENTACION2 +
"\\"año\": " + l.getAñoPublicacion() + ",");
                bw.newLine();
                bw.write(INDENTACION4 + INDENTACION2 +
"\\"paginas\": " + l.getNumPaginas() + ",");
                bw.newLine();
                bw.write(INDENTACION4 + INDENTACION2 +
"\\"disponible\": " + l.getDisponible() + ",");
                bw.newLine();
                bw.write(INDENTACION4 + INDENTACION2 +
"\\"prestamos\": " + l.getPrestamos());
                bw.newLine();
                bw.write(INDENTACION4 + INDENTACION + "}" +
+ (i < librosCategoria.size() - 1 ? "," : ""));
                bw.newLine();
            }
            bw.write(INDENTACION4 + "],");
            bw.newLine();

            // ESTADÍSTICAS DE LA CATEGORÍA
            bw.write(INDENTACION4 + "\"estadisticas\":");
            {
                bw.newLine();
                bw.write(INDENTACION4 + INDENTACION +
"\\"totalPrestamos\": " + totalPrestamosCategoria + ",");
                bw.newLine();
                bw.write(INDENTACION4 + INDENTACION +
"\\"prestamosMedio\": " + formatearDouble(prestamosMedioCategoria) +
",");
                bw.newLine();
                bw.write(INDENTACION4 + INDENTACION +
"\\"libroMasPrestado\": \" + escaparJSON(libroMasPrestado) +
"\",");
                bw.newLine();
                bw.write(INDENTACION4 + "}");
                bw.newLine();

                // CIERRE DE CATEGORÍA
                categoriasEscritas++;
                bw.write(INDENTACION3 + "}" + +

```

```

(categoriasEscritas < categorias.size() ? "," : ""));
        bw.newLine();
    }

    bw.write(INDENTACION2 + "}", ""); // cierre de
"categorias"
    bw.newLine();

    // 5. RESUMEN GLOBAL
    bw.write(INDENTACION2 + "\"resumenGlobal\": {" );
    bw.newLine();
    bw.write(INDENTACION3 + "\"totalCategorias\": " +
totalCategorias + ",");
    bw.newLine();
    bw.write(INDENTACION3 + "\"totalLibros\": " +
totalLibros + ",");
    bw.newLine();
    bw.write(INDENTACION3 + "\"librosDisponibles\": " +
librosDisponibles + ",");
    bw.newLine();
    bw.write(INDENTACION3 + "\"librosPrestados\": " +
librosPrestados + ",");
    bw.newLine();
    bw.write(INDENTACION3 +
"\n\"totalPrestamosHistorico\": " + totalPrestamosHistorico);
    bw.newLine();
    bw.write(INDENTACION2 + "}" );
    bw.newLine();

    bw.write(INDENTACION + "}" ); // CIERRE DEL OBJETO
BIBLIOTECA
    bw.newLine();
    bw.write("}" ); // CIERRE DEL OBJETO RAÍZ

    return true;
} catch (FileNotFoundException e) {
    throw new RuntimeException(e);
} catch (IOException e) {
    throw new RuntimeException(e);
}
}

return false; // (no se ha creado el directorio)
}

```

```

// Agrupa los libros por categoría usando HashMap
private static HashMap<String, ArrayList<Libro>>
agruparPorCategoria(ArrayList<Libro> libros) {
    HashMap<String, ArrayList<Libro>> categorias = new
HashMap<>();
    for (Libro l : libros) {
        String cat = l.getCategoría();
        if (!categorias.containsKey(cat)) {
            categorias.put(cat, new ArrayList<>());
        }
        categorias.get(cat).add(l);
    }
    return categorias;
}
}

```

MainBiblioteca.java (para ejecutar solo este nivel)

```

package nivel2Intermedio_bibliotecaLibros;

import
nivel2Intermedio_bibliotecaLibros.exportadores.ExportadorLibrosC
SV;
import
nivel2Intermedio_bibliotecaLibros.exportadores.ExportadorLibrosX
ML;
import
nivel2Intermedio_bibliotecaLibros.exportadores.ExportadorLibrosJ
SON;
import nivel2Intermedio_bibliotecaLibros.modelo.Libro;

import java.util.ArrayList;

import static utils.Utils.*;
public class MainBiblioteca {
    private final static ArrayList<Libro> libros = new

```

```

ArrayList<>();

public static void main(String[] args) {

    // Creación de libros de ejemplo
    libros.add(new Libro("978-84-123", "El Quijote", "Miguel de Cervantes", "Ficción", 1605, 863, true, 150));
    libros.add(new Libro("978-84-456", "Cien años de soledad", "Gabriel García Márquez", "Ficción", 1967, 471, false, 98));
    libros.add(new Libro("978-84-789", "Breve historia del tiempo", "Stephen Hawking", "Ciencia", 1988, 256, true, 73));
    libros.add(new Libro("978-84-101", "Introducción a la programación", "Ana Torres", "Ciencia", 2010, 350, true, 45));
    libros.add(new Libro("978-84-202", "Historia universal", "VV.AA.", "Historia", 2005, 590, false, 60));
    libros.add(new Libro("978-84-123", "El Quijote", "Miguel de Cervantes", "Ficción", 1605, 863, true, 150));
    libros.add(new Libro("978-84-456", "Cien años de soledad", "Gabriel García Márquez", "Ficción", 1967, 471, false, 98));
    libros.add(new Libro("978-84-789", "Breve historia del tiempo", "Stephen Hawking", "Ciencia", 1988, 256, true, 73));
    libros.add(new Libro("978-84-101", "Introducción a la programación", "Ana Torres", "Ciencia", 2010, 350, true, 45));
    libros.add(new Libro("978-84-202", "Historia universal", "VV.AA.", "Historia", 2005, 590, false, 60));
    libros.add(new Libro("978-84-303", "Los pilares de la Tierra", "Ken Follett", "Ficción", 1989, 1076, true, 110));
    libros.add(new Libro("978-84-404", "Sapiens: De animales a dioses", "Yuval Noah Harari", "Historia", 2011, 496, true, 85));
    libros.add(new Libro("978-84-505", "La teoría del todo", "Stephen Hawking", "Ciencia", 2002, 224, false, 40));
    libros.add(new Libro("978-84-606", "Fundación", "Isaac Asimov", "Ficción", 1951, 320, true, 135));
}

```

```

        libros.add(new Libro("978-84-707", "Un mundo feliz",
"Aldous Huxley", "Ficción", 1932, 288, false, 90));
        libros.add(new Libro("978-84-808", "El origen de las
especies", "Charles Darwin", "Ciencia", 1859, 502, true, 51));
        libros.add(new Libro("978-84-909", "La Segunda Guerra
Mundial", "Antony Beevor", "Historia", 2012, 1152, false, 58));
        libros.add(new Libro("978-84-010", "El nombre de la
rosa", "Umberto Eco", "Ficción", 1980, 672, true, 120));
        libros.add(new Libro("978-84-111", "Crónica de una muerte
anunciada", "Gabriel García Márquez", "Ficción", 1981, 128,
true, 80));
        libros.add(new Libro("978-84-212", "Cosmos", "Carl
Sagan", "Ciencia", 1980, 384, false, 62));
        libros.add(new Libro("978-84-313", "Guns, Germs, and
Steel", "Jared Diamond", "Historia", 1997, 528, true, 70));

    while (true) {
        System.out.println("\n***** MENÚ PRINCIPAL
*****");
        System.out.println("1. Exportar a CSV");
        System.out.println("2. Exportar a XML");
        System.out.println("3. Exportar a JSON");
        System.out.println("4. Exportar a TODOS los
formatos");
        System.out.println("0. Salir");
        int opcion = leerOpcion();

        System.out.println();
        boolean exito = false;
        String nombreArchivo = "libros";

        switch (opcion) {
            case 1:
                exito = ExportadorLibrosCSV.exportar(libros,
nombreArchivo);
                break;
        }
    }
}

```

```

        case 2:
            exito = ExportadorLibrosXML.exportar(libros,
nombreArchivo);
            break;
        case 3:
            exito = ExportadorLibrosJSON.exportar(libros,
nombreArchivo);
            break;
        case 4:
            int exitosos = 0;
            // Exportar CSV
            System.out.println("1/3 Exportando a
CSV...");
            if (ExportadorLibrosCSV.exportar(libros,
nombreArchivo)) {
                exitosos++;
            }
            // Exportar XML
            System.out.println("2/3 Exportando a
XML...");
            if (ExportadorLibrosXML.exportar(libros,
nombreArchivo)) {
                exitosos++;
            }
            // Exportar JSON
            System.out.println("3/3 Exportando a
JSON...");
            if (ExportadorLibrosJSON.exportar(libros,
nombreArchivo)) {
                exitosos++;
            }
            // Resumen
            System.out.println("\nRESUMEN DE
EXPORTACIÓN:");
            System.out.println("Formatos exportados: " +
exitosos + "/3\n");

```

```

        if (exitosos == 3) {
            exito = true;
        } else {
            System.out.println("⚠ No se han podido
crear todos los archivos.");
        }
        break;
    case 0:
        System.out.println("Cerrando programa...");
        sc.close();
        return;
    default:
        opcionInvalida();
        continue;
    }
    if (exito) {
        System.out.println("¡EXPORTACIÓN COMPLETADA
:) !");
        System.out.println("(Ubicación:
exportaciones)");
    } else {
        System.out.println("No se ha podido completar la
exportación.");
    }
    pausar();
}
}
}

```

NIVEL 3: AVANZADO - Sistema de Reservas de Hotel

ESTRUCTURA



Descripción

Crea un sistema completo de exportación para un hotel con habitaciones, reservas y clientes.

Estructura de datos

```
class Cliente {  
    - id: int  
    - nombre: String  
    - email: String  
    - telefono: String  
}  
  
class Habitacion {  
    - numero: int  
    - tipo: String (Individual, Doble, Suite)  
    - precioPorNoche: double  
    - disponible: boolean  
}  
  
class Reserva {  
    - id: int  
    - cliente: Cliente  
    - habitacion: Habitacion  
    - fechaEntrada: LocalDate  
    - fechaSalida: LocalDate  
    - noches: int
```

```

    - precioTotal: double
    - estado: String (Confirmada, Cancelada, Completada)
}

```

Cliente.java

```

package nivel3Avanzado_sistemaReservasHotel.modelo;

public class Cliente {
    private int id;
    private String nombre;
    private String email;
    private String telefono;

    public Cliente(int id, String nombre, String email, String telefono) {
        this.id = id;
        this.nombre = nombre;
        this.email = email;
        this.telefono = telefono;
    }

    // Getters
    public int getId() {
        return id;
    }
    public String getNombre() {
        return nombre;
    }
    public String getEmail() {
        return email;
    }
    public String getTelefono() {
        return telefono;
    }
}

```

Habitacion.java

```

package nivel3Avanzado_sistemaReservasHotel.modelo;

public class Habitacion {
    private int numero;
    private String tipo; // Individual, Doble, Suite
    private double precioPorNoche;
}

```

```

private boolean disponible;

public Habitacion(int numero, String tipo, double precioPorNoche, boolean disponible) {
    this.numero = numero;
    this.tipo = tipo;
    this.precioPorNoche = precioPorNoche;
    this.disponible = disponible;
}

// Getters
public int getNumero() {
    return numero;
}
public String getTipo() {
    return tipo;
}
public double getPrecioPorNoche() {
    return precioPorNoche;
}
public boolean getDisponible() {
    return disponible;
}
}

```

Reserva.java

```

package nivel3Avanzado_sistemaReservasHotel.modelo;

import java.time.LocalDate;

public class Reserva {
    private int id;
    private Cliente cliente;
    private Habitacion habitacion;
    private LocalDate fechaEntrada;
    private LocalDate fechaSalida;
    private int noches;
    private double precioTotal;
    private String estado; // Confirmada, Cancelada, Completada

    public Reserva(int id, Cliente cliente, Habitacion habitacion,
LocalDate fechaEntrada, LocalDate fechaSalida, int noches, double
precioTotal, String estado) {
        this.id = id;
    }
}

```

```

        this.cliente = cliente;
        this.habitacion = habitacion;
        this.fechaEntrada = fechaEntrada;
        this.fechaSalida = fechaSalida;
        this.noches = noches;
        this.precioTotal = precioTotal;
        this.estado = estado;
    }

    // Getters
    public int getId() {
        return id;
    }
    public Cliente getCliente() {
        return cliente;
    }
    public Habitacion getHabitacion() {
        return habitacion;
    }
    public LocalDate getFechaEntrada() {
        return fechaEntrada;
    }
    public LocalDate getFechaSalida() {
        return fechaSalida;
    }
    public int getNoches() {
        return noches;
    }
    public double getPrecioTotal() {
        return precioTotal;
    }
    public String getEstado() {
        return estado;
    }
}

```

Tareas

3.1. Exportar a CSV con relaciones

Exportar reservas con información completa (cliente + habitación)

Aplanar las relaciones en columnas

Incluir campos calculados

Ejemplo de salida:

```
CSV
ID;ClienteNombre;ClienteEmail;HabitacionNum;TipoHabitacion;FechaEntrada
1;Juan
García;juan@email.com;101;Doble;20/10/2025;23/10/2025;3;270.00;Confirmado
2;María
López;maria@email.com;205;Suite;21/10/2025;25/10/2025;4;800.00;Confirmado
...
```

ExportadorReservasCSV.java

```
package nivel3Avanzado_sistemaReservasHotel.exportadores;

import nivel3Avanzado_sistemaReservasHotel.modelo.Reserva;
import java.io.BufferedReader;
import java.io.File;
import java.io.FileWriter;
import java.io.IOException;
import java.text.SimpleDateFormat;
import java.time.format.DateTimeFormatter;
import java.util.ArrayList;
import java.util.Date;

import static utils.Utils.*;

public class ExportadorReservasCSV {
    static final String CARPETA = "exportaciones";
    static String fecha = new SimpleDateFormat("yyyyMMdd_HHmmss").format(new Date());
    private static final String SEPARADOR = ";" // Separador CSV

    public static boolean exportar(ArrayList<Reserva> reservas,
        String nombreArchivo) {

        // VALIDACIONES

        if (reservas == null || reservas.isEmpty()) {
            System.out.println("ERROR: No hay elementos para exportar.");
            return false;
        }

        File archivo = new File(CARPETA + "/" + nombreArchivo);
        try (BufferedWriter bw = new BufferedWriter(new FileWriter(archivo))) {
            bw.write("ID;ClienteNombre;ClienteEmail;HabitacionNum;TipoHabitacion;FechaEntrada" + System.lineSeparator());
            for (Reserva reserva : reservas) {
                bw.write(reserva.getId() + ";" + reserva.getNombre() + ";" + reserva.getEmail() + ";" + reserva.getNumHabitacion() + ";" + reserva.getTipoHabitacion() + ";" + reserva.getFechaEntrada() + ";" + reserva.getFechaSalida() + ";" + reserva.getPrecio() + ";" + reserva.getEstado());
                bw.newLine();
            }
        } catch (IOException e) {
            System.out.println("Error al escribir en el archivo: " + e.getMessage());
        }
    }
}
```

```

    }

    String rutaCompleta = CARPETA + File.separator +
nombreArchivo + "_" + fecha + ".csv";

    if (rutaCompleta == null || rutaCompleta.trim().isEmpty())
{
    System.out.println("ERROR: El nombre del archivo no
puede estar vacío.");
    return false;
}

// CREAR DIRECTORIO

if (crearDirectorio(CARPETA) {
    try (BufferedWriter writer = new BufferedWriter(new
FileWriter(rutaCompleta))) {

        // 1. ESCRIBIR ENCABEZADO
        escribirEncabezado(writer);

        // 2. ESCRIBIR CADA RESERVA
        for (Reserva r : reservas) {
            escribirReserva(writer, r);
        }

        // 3. ESCRIBIR RESUMEN
        escribirResumen(writer, reservas);

        return true;

    } catch (IOException e) {
        System.out.println("Error al escribir el archivo
CSV: " + e.getMessage());
    }
}
return false; // (no se ha creado el directorio)
}

// Encabezado (nombres de las columnas)
private static void escribirEncabezado(BufferedWriter writer)
throws IOException {
    writer.write("ID" + SEPARADOR);
    writer.write("ClienteNombre" + SEPARADOR);
    writer.write("ClienteEmail" + SEPARADOR);
}

```

```

        writer.write("ClienteTelefono" + SEPARADOR);
        writer.write("HabitacionNum" + SEPARADOR);
        writer.write("TipoHabitacion" + SEPARADOR);
        writer.write("PrecioNoche" + SEPARADOR);
        writer.write("FechaEntrada" + SEPARADOR);
        writer.write("FechaSalida" + SEPARADOR);
        writer.write("Noches" + SEPARADOR);
        writer.write("PrecioTotal" + SEPARADOR);
        writer.write("Estado"); // El último elemento SIN
SEPARADOR
        writer.newLine();
    }

    // Escribe todos los campos de cada reserva (aplanando
relaciones)
    private static void escribirReserva(BufferedWriter writer,
Reserva r) throws IOException {
    DateTimeFormatter formatoFecha =
DateTimeFormatter.ofPattern("dd/MM/yyyy");

    writer.write(r.getId() + SEPARADOR);
    writer.write(esparCSV(r.getCliente().getNombre()) +
SEPARADOR);
    writer.write(esparCSV(r.getCliente().getEmail()) +
SEPARADOR);
    writer.write(esparCSV(r.getCliente().getTelefono()) +
SEPARADOR);
    writer.write(r.getHabitacion().getNumero() + SEPARADOR);
    writer.write(esparCSV(r.getHabitacion().getTipo()) +
SEPARADOR);

writer.write(formatearDouble(r.getHabitacion().getPrecioPorNoche(
)) + SEPARADOR);
    writer.write(r.getFechaEntrada().format(formatoFecha) +
SEPARADOR);
    writer.write(r.getFechaSalida().format(formatoFecha) +
SEPARADOR);
    writer.write(r.getNoches() + SEPARADOR);
    writer.write(formatearDouble(r.getPrecioTotal()) +
SEPARADOR);
    writer.write(esparCSV(r.getEstado()));
    writer.newLine();
}

// Escribe el resumen

```

```

private static void escribirResumen(BufferedWriter writer,
ArrayList<Reserva> reservas) throws IOException {
    // CÁLCULOS
    int totalReservas = reservas.size();
    double sumaTotal = 0;
    int confirmadas = 0, canceladas = 0, completadas = 0;

    for (Reserva r : reservas) {
        sumaTotal += r.getPrecioTotal();
        switch (r.getEstado().toLowerCase()) {
            case "confirmada": confirmadas++; break;
            case "cancelada": canceladas++; break;
            case "completada": completadas++; break;
        }
    }

    writer.newLine();
    writer.write("# Total reservas: " + totalReservas);
    writer.newLine();
    writer.write("# Total ingresos (reservas): " +
    formatearDouble(sumaTotal));
    writer.newLine();
    writer.write("# Confirmadas: " + confirmadas + " | "
    Canceladas: " + canceladas + " | Completadas: " + completadas);
    writer.newLine();
}
}

```

3.2. Exportar a XML con relaciones complejas

Mantener la estructura de objetos anidados

Incluir información completa de cliente y habitación en cada reserva

Estadísticas por tipo de habitación y estado

Ejemplo de salida:

```

<?xml version="1.0" encoding="UTF-8"?>
<hotel>
<informacion>
<nombree>Hotel Paradise</nombree>

```

```

<fecha>19/10/2025</fecha>
</informacion>

<reservas totalReservas="10">
<reserva id="1" estado="Confirmada">
<cliente>
<id>1</id>
<nombre>Juan García</nombre>
<email>juan@email.com</email>
<telefono>666111222</telefono>
</cliente>
<habitacion numero="101" tipo="Doble">
<precioPorNoche>90.00</precioPorNoche>
<disponible>false</disponible>
</habitacion>
<fechas>
<entrada>20/10/2025</entrada>
<salida>23/10/2025</salida>
<noches>3</noches>
</fechas>
<precio>
<total>270.00</total>
<porNoche>90.00</porNoche>
</precio>
</reserva>
...
</reservas>

<estadisticas>
<porTipoHabitacion>
<Individual>
<totalReservas>3</totalReservas>
<ingresos>450.00</ingresos>
</Individual>
<Doble>
<totalReservas>5</totalReservas>
<ingresos>1350.00</ingresos>
</Doble>
<Suite>
<totalReservas>2</totalReservas>
<ingresos>1600.00</ingresos>
</Suite>
</porTipoHabitacion>
<porEstado>

```

```

<Confirmada>7</Confirmada>
<Cancelada>1</Cancelada>
<Completada>2</Completada>
</porEstado>
<resumen>
<totalReservas>10</totalReservas>
<ingresosTotal>3400.00</ingresosTotal>
<nochesReservadas>38</nochesReservadas>
</resumen>
</estadisticas>
</hotel>
```

ExportadorReservasXML.java

```

package nivel3Avanzado_sistemaReservasHotel.exportadores;

import nivel3Avanzado_sistemaReservasHotel.modelo.Reserva;
import nivel3Avanzado_sistemaReservasHotel.modelo.Cliente;
import nivel3Avanzado_sistemaReservasHotel.modelo.Habitacion;

import java.io.BufferedReader;
import java.io.File;
import java.io.FileWriter;
import java.io.IOException;
import java.text.SimpleDateFormat;
import java.time.format.DateTimeFormatter;
import java.time.LocalDateTime;
import java.util.ArrayList;
import java.util.Date;
import java.util.HashMap;

import static utils.Utils.*;

public class ExportadorReservasXML {
    static final String CARPETA = "exportaciones";
    static String fecha = new
SimpleDateFormat("yyyyMMdd_HHmmss").format(new Date());

    private static final String INDENTACION = "    ";
    private static final String INDENTACION2 = INDENTACION +
INDENTACION;
    private static final String INDENTACION3 = INDENTACION2 +
INDENTACION;
```

```

    private static final String INDENTACION4 = INDENTACION3 +
INDENTACION;

    public static boolean exportar(ArrayList<Reserva> reservas,
String nombreArchivo) {
    try {

        // VALIDACIONES

        if (reservas == null || reservas.isEmpty()) {
            System.out.println("ERROR: No hay elementos para
exportar.");
            return false;
        }

        String rutaCompleta = CARPETA + File.separator +
nombreArchivo + "_" + fecha + ".xml";

        if (rutaCompleta == null ||
rutaCompleta.trim().isEmpty()) {
            System.out.println("ERROR: El nombre del archivo
no puede estar vacío.");
            return false;
        }

        // CREAR DIRECTORIO

        if (crearDirectorio(CARPETA)) {
            try (BufferedWriter bw = new BufferedWriter(new
FileWriter(rutaCompleta))) {

                // 1. Declaración XML y elemento raíz
                bw.write("<?xml version=\"1.0\""
encoding="UTF-8"?>");
                bw.newLine();
                bw.write("<hotel>");
                bw.newLine();

                // 2. Información general
                String fechaActual =
LocalDateTime.now().format(DateTimeFormatter.ofPattern("dd/MM/yyyy"));
                bw.write(INDENTACION + "<informacion>");
                bw.newLine();

```

```

        bw.write(INDENTACION2 + "<nombre>Hotel
Paradise Iris Pérez</nombre>");
        bw.newLine();
        bw.write(INDENTACION2 + "<fecha>" +
escaparXML(fechaActual) + "</fecha>");
        bw.newLine();
        bw.write(INDENTACION + "</informacion>"); 
        bw.newLine();

        // 3. Reservas con información anidada
        bw.write(INDENTACION + "<reservas
totalReservas=\\" + reservas.size() + "\\\">>"); 
        bw.newLine();

        DateTimeFormatter formatoFecha =
DateTimeFormatter.ofPattern("dd/MM/yyyy");

        // Para estadísticas
        HashMap<String, Integer> reservasPorTipo = new
HashMap<>();
        HashMap<String, Double> ingresosPorTipo = new
HashMap<>();
        HashMap<String, Integer> reservasPorEstado =
new HashMap<>();
        int nochesReservadas = 0;
        double ingresosTotal = 0;

        for (Reserva r : reservas) {
            // Actualizar estadísticas
            String tipo = r.getHabitacion().getTipo();
            String estado = r.getEstado();
            reservasPorTipo.put(tipo,
reservasPorTipo.getOrDefault(tipo, 0) + 1);
            ingresosPorTipo.put(tipo,
ingresosPorTipo.getOrDefault(tipo, 0.0) + r.getPrecioTotal());
            reservasPorEstado.put(estado,
reservasPorEstado.getOrDefault(estado, 0) + 1);
            nochesReservadas += r.getNoches();
            ingresosTotal += r.getPrecioTotal();

            // Inicio reserva
            bw.write(INDENTACION2 + "<reserva id=\\" + 
r.getId() + "\\\" estado=\\" + escaparXML(estado) + "\\\">>"); 
            bw.newLine();

```

```

        // Cliente
        Cliente c = r.getCliente();
        bw.write(INDENTACION3 + "<cliente>");
        bw.newLine();
        bw.write(INDENTACION4 + "<id>" + c.getId()
+ "</id>");

        bw.newLine();
        bw.write(INDENTACION4 + "<nombre>" +
escaparXML(c.getNombre()) + "</nombre>");
        bw.newLine();
        bw.write(INDENTACION4 + "<email>" +
escaparXML(c.getEmail()) + "</email>");
        bw.newLine();
        bw.write(INDENTACION4 + "<telefono>" +
escaparXML(c.getTelefono()) + "</telefono>");

        bw.newLine();
        bw.write(INDENTACION3 + "</cliente>");
        bw.newLine();

        // Habitacion
        Habitacion h = r.getHabitacion();
        bw.write(INDENTACION3 + "<habitacion
numero=\\" + h.getNumero() + "\" tipo=\\" +
escaparXML(h.getTipo()) + "\">\n");
        bw.newLine();
        bw.write(INDENTACION4 + "<precioPorNoche>" +
formatearDouble(h.getPrecioPorNoche()) + "</precioPorNoche>");

        bw.newLine();
        bw.write(INDENTACION4 + "<disponible>" +
h.getDisponible() + "</disponible>");

        bw.newLine();
        bw.write(INDENTACION3 + "</habitacion>");

        bw.newLine();

        // Fechas
        bw.write(INDENTACION3 + "<fechas>");
        bw.newLine();
        bw.write(INDENTACION4 + "<entrada>" +
r.getFechaEntrada().format(formatoFecha) + "</entrada>");

        bw.newLine();
        bw.write(INDENTACION4 + "<salida>" +
r.getFechaSalida().format(formatoFecha) + "</salida>");

        bw.newLine();
        bw.write(INDENTACION4 + "<noches>" +
r.getNoches() + "</noches>");
```

```

        bw.newLine();
        bw.write(INDENTACION3 + "</fechas>"); 
        bw.newLine();

        // Precio
        bw.write(INDENTACION3 + "<precio>"); 
        bw.newLine();
        bw.write(INDENTACION4 + "<total>" + 
formatarDouble(r.getPrecioTotal()) + "</total>"); 
        bw.newLine();
        bw.write(INDENTACION4 + "<porNoche>" + 
formatarDouble(h.getPrecioPorNoche()) + "</porNoche>"); 
        bw.newLine();
        bw.write(INDENTACION3 + "</precio>"); 
        bw.newLine();

        // Fin reserva
        bw.write(INDENTACION2 + "</reserva>"); 
        bw.newLine();
    }

    bw.write(INDENTACION + "</reservas>"); 
    bw.newLine();

    // 4. Estadísticas
    bw.write(INDENTACION + "<estadisticas>"); 
    bw.newLine();

    // Por tipo de habitación
    bw.write(INDENTACION2 + 
"<porTipoHabitacion>"); 
    bw.newLine();
    for (String tipo : reservasPorTipo.keySet()) { 
        bw.write(INDENTACION3 + "<" + 
escaparXML(tipo) + ">"); 
        bw.newLine();
        bw.write(INDENTACION4 + "<totalReservas>" + 
reservasPorTipo.get(tipo) + "</totalReservas>"); 
        bw.newLine();
        bw.write(INDENTACION4 + "<ingresos>" + 
formatarDouble(ingresosPorTipo.get(tipo)) + "</ingresos>"); 
        bw.newLine();
        bw.write(INDENTACION3 + "<" + 
escaparXML(tipo) + ">"); 
        bw.newLine();
    }
}

```

```

        }
        bw.write(INDENTACION2 +
"</porTipoHabitacion>");

        bw.newLine();

        // Por estado
        bw.write(INDENTACION2 + "<porEstado>");
        bw.newLine();
        for (String estado :
reservasPorEstado.keySet()) {
            bw.write(INDENTACION3 + "<" +
escaparXML(estado) + ">" + reservasPorEstado.get(estado) + "</" +
escaparXML(estado) + ">");
            bw.newLine();
        }
        bw.write(INDENTACION2 + "</porEstado>");
        bw.newLine();

        // Resumen global
        bw.write(INDENTACION2 + "<resumen>");
        bw.newLine();
        bw.write(INDENTACION3 + "<totalReservas>" +
reservas.size() + "</totalReservas>");
        bw.newLine();
        bw.write(INDENTACION3 + "<ingresosTotal>" +
formatearDouble(ingresosTotal) + "</ingresosTotal>");
        bw.newLine();
        bw.write(INDENTACION3 + "<nochesReservadas>" +
nochesReservadas + "</nochesReservadas>");
        bw.newLine();
        bw.write(INDENTACION2 + "</resumen>");
        bw.newLine();

        bw.write(INDENTACION + "</estadisticas>");
        bw.newLine();

        bw.write("</hotel>");
        bw.newLine();

        return true;
    }
}

return false; // (no se ha creado el directorio)
} catch (IOException e) {

```

```
        System.err.println("Error al escribir el archivo XML:  
" + e.getMessage());  
        return false;  
    }  
}  
}
```

3.3. Exportar a JSON con estructura óptima

- Diseñar estructura JSON eficiente
- Evitar duplicación de información
- Incluir múltiples niveles de anidación

Ejemplo de salida:

```
{  
  "hotel": {  
    "informacion": {  
      "nombre": "Hotel Paradise",  
      "fecha": "19/10/2025"  
    },  
    "clientes": {  
      "1": {  
        "nombre": "Juan García",  
        "email": "juan@email.com",  
        "telefono": "666111222"  
      },  
      ...  
    },  
    "habitaciones": {  
      "101": {  
        "tipo": "Doble",  
        "precioPorNoche": 90.00,  
        "disponible": false  
      },  
      ...  
    },  
    "reservas": [  
      {  
        "id": 1,  
        "clienteId": 1,  
        "habitacionId": 101,  
        "fecha": "19/10/2025",  
        "precio": 90.00  
      }  
    ]  
  }  
}
```

```

"habitacionNumero": 101,
"fechaEntrada": "20/10/2025",
"fechaSalida": "23/10/2025", "noches":
3,
"precioTotal": 270.00,
"estado": "Confirmada"
},
...
],
"estadisticas": {
"porTipoHabitacion": {
"Individual": {
"totalReservas": 3,
"ingresos": 450.00,
"porcentaje": 30.0
},
"Doble": {
"totalReservas": 5,
"ingresos": 1350.00,
"porcentaje": 50.0
},
"Suite": {
"totalReservas": 2,
"ingresos": 1600.00,
"porcentaje": 20.0
}
},
"porEstado": {
"Confirmada": 7,
"Cancelada": 1,
"Completada": 2
},
"resumen": {
"totalReservas": 10,
"ingresosTotal": 3400.00,
"nochesReservadas": 38,
"ocupacionMedia": 65.5
}
}
}
}
}

```

ExportadorReservasJSON.java

```
package nivel3Avanzado_sistemaReservasHotel.exportadores;
```

```

import nivel3Avanzado_sistemaReservasHotel.modelo.Reserva;
import nivel3Avanzado_sistemaReservasHotel.modelo.Cliente;
import nivel3Avanzado_sistemaReservasHotel.modelo.Habitacion;

import java.io.*;
import java.nio.charset.StandardCharsets;
import java.text.SimpleDateFormat;
import java.time.LocalDateTime;
import java.time.format.DateTimeFormatter;
import java.util.*;

import static utils.Utils.*;

public class ExportadorReservasJSON {
    static final String CARPETA = "exportaciones";
    static String fecha = new
SimpleDateFormat("yyyyMMdd_HH:mm:ss").format(new Date()));

    private static final String INDENTACION = "    ";
    private static final String INDENTACION2 = INDENTACION +
INDENTACION;
    private static final String INDENTACION3 = INDENTACION2 +
INDENTACION;
    private static final String INDENTACION4 = INDENTACION3 +
INDENTACION;

    public static boolean exportar(ArrayList<Reserva> reservas,
String nombreArchivo) {
        // VALIDACIONES

        if (reservas == null || reservas.isEmpty()) {
            System.out.println("ERROR: No hay elementos para
exportar.");
            return false;
        }

        String rutaCompleta = CARPETA + File.separator +
nombreArchivo + "_" + fecha + ".json";

        if (rutaCompleta == null || rutaCompleta.trim().isEmpty())
{
            System.out.println("ERROR: El nombre del archivo no
puede estar vacío.");
            return false;
        }

        // CREAR DIRECTORIO
        if (crearDirectorio(CARPETA)) {

```

```

        // Utilizo OutputStreamWriter y FileOutputStream para
aplicar UTF_8
try (BufferedWriter bw = new BufferedWriter(new
OutputStreamWriter(new FileOutputStream(rutaCompleta),
StandardCharsets.UTF_8))) {

    // 1. APERTURA DEL OBJETO RAÍZ
    bw.write("{");
    bw.newLine();
    bw.write(INDENTACION + "\"hotel\": {");
    bw.newLine();

    // 2. INFORMACIÓN GENERAL
    String fechaActual =
LocalDateTime.now().format(DateTimeFormatter.ofPattern("dd/MM/yyyy
"));

    bw.write(INDENTACION2 + "\"informacion\": {");
    bw.newLine();
    bw.write(INDENTACION3 + "\"nombre\": \"Hotel
Paradise Iris Pérez\",");
    bw.newLine();
    bw.write(INDENTACION3 + "\"fecha\": \""
+ escaparJSON(fechaActual) + "\"");
    bw.newLine();
    bw.write(INDENTACION2 + "},\"");
    bw.newLine();

    // 3. GENERAR MAPAS ÚNICOS DE CLIENTES Y
HABITACIONES
    HashMap<Integer, Cliente> clientesUnicos = new
HashMap<>();
    HashMap<Integer, Habitacion> habitacionesUnicas =
new HashMap<>();
    for (Reserva r : reservas) {
        clientesUnicos.put(r.getCliente().getId(),
r.getCliente());
        habitacionesUnicas.put(r.getHabitacion().getNumero(),
r.getHabitacion());
    }

    // 4. CLIENTES
    bw.write(INDENTACION2 + "\"clientes\": {");
    bw.newLine();
    int contClientes = 0;
    for (Integer id : clientesUnicos.keySet()) {
        Cliente c = clientesUnicos.get(id);

```

```

        bw.write(INDENTACION3 + "\\" + id + "\\": {" );
        bw.newLine();
        bw.write(INDENTACION4 + "\\\"nombre\\": \" +
escaparJSON(c.getNombre()) + "\", ");
        bw.newLine();
        bw.write(INDENTACION4 + "\\\"email\\": \" +
escaparJSON(c.getEmail()) + "\", ");
        bw.newLine();
        bw.write(INDENTACION4 + "\\\"telefono\\": \" +
escaparJSON(c.getTelefono()) + "\");
        bw.newLine();
        bw.write(INDENTACION3 + "}" + (++contClientes
< clientesUnicos.size() ? "," : ""));
        bw.newLine();
    }
    bw.write(INDENTACION2 + "},");
    bw.newLine();

    // 5. HABITACIONES
    bw.write(INDENTACION2 + "\\\"habitaciones\\": {" );
    bw.newLine();
    int contHabitaciones = 0;
    for (Integer num : habitacionesUnicas.keySet()) {
        Habitacion h = habitacionesUnicas.get(num);
        bw.write(INDENTACION3 + "\\" + num + "\\": {" );
        bw.newLine();
        bw.write(INDENTACION4 + "\\\"tipo\\": \" +
escaparJSON(h.getTipo()) + "\", ");
        bw.newLine();
        bw.write(INDENTACION4 + "\\\"precioPorNoche\\": "
+ formatearDouble(h.getPrecioPorNoche()) + ",");
        bw.newLine();
        bw.write(INDENTACION4 + "\\\"disponible\\": " +
h.getDisponible());
        bw.newLine();
        bw.write(INDENTACION3 + "}" + ++
contHabitaciones < habitacionesUnicas.size() ? "," : ""));
        bw.newLine();
    }
    bw.write(INDENTACION2 + "},");
    bw.newLine();

    // 6. RESERVAS
    bw.write(INDENTACION2 + "\\\"reservas\\": []);
    bw.newLine();
    DateTimeFormatter formatoFecha =
DateTimeFormatter.ofPattern("dd/MM/yyyy");
    for (int i = 0; i < reservas.size(); i++) {

```

```

        Reserva r = reservas.get(i);
        bw.write(INDENTACION3 + "{");
        bw.newLine();
        bw.write(INDENTACION4 + "\"id\": " + r.getId()
+ ", ");
        bw.newLine();
        bw.write(INDENTACION4 + "\"clienteId\": " +
r.getCliente().getId() + ",");
        bw.newLine();
        bw.write(INDENTACION4 + "\"habitacionNumero\": "
+ r.getHabitacion().getNumero() + ",");
        bw.newLine();
        bw.write(INDENTACION4 + "\"fechaEntrada\": \""
+ r.getFechaEntrada().format(formatoFecha) + "\",");
        bw.newLine();
        bw.write(INDENTACION4 + "\"fechaSalida\": \""
+ r.getFechaSalida().format(formatoFecha) + "\",");
        bw.newLine();
        bw.write(INDENTACION4 + "\"noches\": " +
r.getNoches() + ",");
        bw.newLine();
        bw.write(INDENTACION4 + "\"precioTotal\": " +
formatearDouble(r.getPrecioTotal()) + ",");
        bw.newLine();
        bw.write(INDENTACION4 + "\"estado\": \""
+ escaparJSON(r.getEstado()) + "\");
        bw.newLine();
        bw.write(INDENTACION3 + "}" + (i <
reservas.size() - 1 ? "," : ""));
        bw.newLine();
    }
    bw.write(INDENTACION2 + "],");
    bw.newLine();

    // 7. ESTADÍSTICAS
    // --- CÁLCULOS ---
    HashMap<String, Integer> reservasPorTipo = new
    HashMap<String, Double> ingresosPorTipo = new
    HashMap<String, Integer> reservasPorEstado = new
    int nochesReservadas = 0;
    double ingresosTotal = 0;

    for (Reserva r : reservas) {
        String tipo = r.getHabitacion().getTipo();
        String estado = r.getEstado();

```

```

        reservasPorTipo.put(tipo,
reservasPorTipo.getOrDefault(tipo, 0) + 1);
        ingresosPorTipo.put(tipo,
ingresosPorTipo.getOrDefault(tipo, 0.0) + r.getPrecioTotal());
        reservasPorEstado.put(estado,
reservasPorEstado.getOrDefault(estado, 0) + 1);
        nochesReservadas += r.getNoches();
        ingresosTotal += r.getPrecioTotal();
    }

    // Porcentaje por tipo de habitación
    HashMap<String, Double> porcentajePorTipo = new
HashMap<>();
    for (String tipo : reservasPorTipo.keySet()) {
        double porcentaje = reservas.size() > 0 ?
(reservasPorTipo.get(tipo) * 100.0) / reservas.size() : 0;
        porcentajePorTipo.put(tipo, porcentaje);
    }
    // Ocupación media
    double ocupacionMedia = reservas.size() > 0 ?
(nochesReservadas * 100.0) / reservas.size() : 0;

    bw.write(INDENTACION2 + "\"estadisticas\": { ");
    bw.newLine();

    // Por tipo de habitación
    bw.write(INDENTACION3 + "\"porTipoHabitacion\": ");
    bw.newLine();
    int tiposEscritos = 0;
    for (String tipo : reservasPorTipo.keySet()) {
        bw.write(INDENTACION4 + "\"" +
escaparJSON(tipo) + "\": {" );
        bw.newLine();
        bw.write(INDENTACION4 + INDENTACION +
"\\"totalReservas\": " + reservasPorTipo.get(tipo) + ", ");
        bw.newLine();
        bw.write(INDENTACION4 + INDENTACION +
"\\"ingresos\": " + formatearDouble(reservasPorTipo.get(tipo)) + ,
");
        bw.newLine();
        bw.write(INDENTACION4 + INDENTACION +
"\\"porcentaje\": " +
formatearDouble(porcentajePorTipo.get(tipo)));
        bw.newLine();
        bw.write(INDENTACION4 + "}" + (++tiposEscritos
< reservasPorTipo.size() ? "," : ""));
        bw.newLine();
    }
}

```

```

        }

        bw.write(INDENTACION3 + "}" ,") ;
        bw.newLine();

        // Por estado
        bw.write(INDENTACION3 + "\\"porEstado\": {" );
        bw.newLine();
        int estadosEscritos = 0;
        for (String estado : reservasPorEstado.keySet()) {
            bw.write(INDENTACION4 + "\\" +
escaparJSON(estado) + ":" + reservasPorEstado.get(estado) + (
++estadosEscritos < reservasPorEstado.size() ? "," : ""));
            bw.newLine();
        }
        bw.write(INDENTACION3 + "}" ,") ;
        bw.newLine();

        // Resumen global
        bw.write(INDENTACION3 + "\\"resumen\": {" );
        bw.newLine();
        bw.write(INDENTACION4 + "\\"totalReservas\": " +
reservas.size() + ",");
        bw.newLine();
        bw.write(INDENTACION4 + "\\"ingresosTotal\": " +
formatearDouble(ingresosTotal) + ",");
        bw.newLine();
        bw.write(INDENTACION4 + "\\"nochesReservadas\": " +
nochesReservadas + ",");
        bw.newLine();
        bw.write(INDENTACION4 + "\\"ocupacionMedia\": " +
formatearDouble(ocupacionMedia));
        bw.newLine();
        bw.write(INDENTACION3 + "}" );
        bw.newLine();

        bw.write(INDENTACION2 + "}" ); // cierre
estadisticas
        bw.newLine();

        bw.write(INDENTACION + "}" ); // cierre hotel
        bw.newLine();
        bw.write("}" ); // cierre raíz

        return true;
    } catch (FileNotFoundException e) {
        throw new RuntimeException(e);
    } catch (IOException e) {
        throw new RuntimeException(e);
    }
}

```

```

        }
    }
    return false; // (no se ha creado el directorio)
}
}

```

MainReservasHotel.java (para ejecutar solo este nivel)

```

package nivel3Avanzado_sistemaReservasHotel;

import nivel3Avanzado_sistemaReservasHotel.modelo.Cliente;
import nivel3Avanzado_sistemaReservasHotel.modelo.Habitacion;
import nivel3Avanzado_sistemaReservasHotel.modelo.Reserva;

import
nivel3Avanzado_sistemaReservasHotel.exportadores.ExportadorReserv
asCSV;
import
nivel3Avanzado_sistemaReservasHotel.exportadores.ExportadorReserv
asXML;
import
nivel3Avanzado_sistemaReservasHotel.exportadores.ExportadorReserv
asJSON;

import java.util.ArrayList;
import java.time.LocalDate;

import static utils.Utils.*;

public class MainReservasHotel {
    private final static ArrayList<Reserva> reservas = new
ArrayList<>();

    public static void main(String[] args) {

        // Creo clientes de ejemplo
        Cliente cliente1 = new Cliente(1, "Marta García",
"juan@email.com", "666111222");
        Cliente cliente2 = new Cliente(2, "María López",
"maria@email.com", "666222333");
        Cliente cliente3 = new Cliente(3, "Miguel Sánchez",
"pedro@email.com", "666333444");

        // Creo habitaciones de ejemplo
    }
}

```

```

        Habitacion hab1 = new Habitacion(101, "Doble", 90.00,
false);
        Habitacion hab2 = new Habitacion(205, "Suite", 200.00,
true);
        Habitacion hab3 = new Habitacion(303, "Individual", 50.00,
true);
        Habitacion hab4 = new Habitacion(404, "Doble", 80.00,
true);
        Habitacion hab5 = new Habitacion(505, "Suite", 200.00,
false);

        // Creo reservas de ejemplo
        reservas.add(new Reserva(1, cliente1, hab1,
LocalDate.parse("2025-10-20"), LocalDate.parse("2025-10-23"), 3,
270.00, "Confirmada"));
        reservas.add(new Reserva(2, cliente2, hab2,
LocalDate.parse("2025-10-21"), LocalDate.parse("2025-10-25"), 4,
800.00, "Confirmada"));
        reservas.add(new Reserva(3, cliente3, hab3,
LocalDate.parse("2025-10-18"), LocalDate.parse("2025-10-20"), 2,
100.00, "Completada"));
        reservas.add(new Reserva(4, cliente1, hab4,
LocalDate.parse("2025-10-22"), LocalDate.parse("2025-10-24"), 2,
160.00, "Cancelada"));
        reservas.add(new Reserva(5, cliente2, hab3,
LocalDate.parse("2025-10-25"), LocalDate.parse("2025-10-28"), 3,
150.00, "Confirmada"));
        reservas.add(new Reserva(6, cliente3, hab5,
LocalDate.parse("2025-10-26"), LocalDate.parse("2025-10-29"), 3,
600.00, "Confirmada"));
        reservas.add(new Reserva(7, cliente1, hab1,
LocalDate.parse("2025-10-28"), LocalDate.parse("2025-10-31"), 3,
270.00, "Completada"));
        reservas.add(new Reserva(8, cliente2, hab4,
LocalDate.parse("2025-10-27"), LocalDate.parse("2025-10-30"), 3,
240.00, "Confirmada"));
        reservas.add(new Reserva(9, cliente3, hab2,
LocalDate.parse("2025-10-29"), LocalDate.parse("2025-11-01"), 3,
600.00, "Confirmada"));
        reservas.add(new Reserva(10, cliente1, hab3,
LocalDate.parse("2025-10-30"), LocalDate.parse("2025-11-01"), 2,
100.00, "Confirmada"));

        while (true) {
            System.out.println("\n..... MENÚ PRINCIPAL .....");
            System.out.println("1. Exportar a CSV");

```

```

        System.out.println("2. Exportar a XML");
        System.out.println("3. Exportar a JSON");
        System.out.println("4. Exportar a TODOS los
formatos");
        System.out.println("0. Salir");
        int opcion = leerOpcion();

        System.out.println();
        boolean exito = false;
        String nombreArchivo = "reservas_hotel";

        switch (opcion) {
            case 1:
                exito =
ExportadorReservasCSV.exportar(reservas, nombreArchivo);
                break;
            case 2:
                exito =
ExportadorReservasXML.exportar(reservas, nombreArchivo);
                break;
            case 3:
                exito =
ExportadorReservasJSON.exportar(reservas, nombreArchivo);
                break;
            case 4:
                int exitosos = 0;
                // Exportar CSV
                System.out.println("1/3 Exportando a CSV...");
                if (ExportadorReservasCSV.exportar(reservas,
nombreArchivo)) {
                    exitosos++;
                }
                // Exportar XML
                System.out.println("2/3 Exportando a XML...");
                if (ExportadorReservasXML.exportar(reservas,
nombreArchivo)) {
                    exitosos++;
                }
                // Exportar JSON
                System.out.println("3/3 Exportando a
JSON...");
                if (ExportadorReservasJSON.exportar(reservas,
nombreArchivo)) {
                    exitosos++;
                }
        }
    }
}

```

```

        // Resumen
        System.out.println("\nRESUMEN DE
EXPORTACIÓN:");
        System.out.println("Formatos exportados: " +
exitosos + "/3\n");
        if (exitosos == 3) {
            exito = true;
        } else {
            System.out.println("Δ No se han podido
crear todos los archivos.");
        }
        break;
    case 0:
        System.out.println("Cerrando programa...");
        sc.close();
        return;
    default:
        opcionInvalida();
        continue;
    }
    if (exito) {
        System.out.println("¡EXPORTACIÓN COMPLETADA :) !");
        System.out.println("(Ubicación: exportaciones/)");
    } else {
        System.out.println("No se ha podido completar la
exportación.");
    }
    pausar();
}
}
}

```

Main.java (para ejecutar el proyecto completo)

```

import
nivellBasico_listadeEstudiantes.exportadores.ExportadorEstudiante
sCSV;
import
nivellBasico_listadeEstudiantes.exportadores.ExportadorEstudiante
sXML;

```

```

import
nivellBasico_listaDeEstudiantes.exportadores.ExportadorEstudiante
sJSON;
import nivellBasico_listaDeEstudiantes.modelo.Estudiante;

import
nivel2Intermedio_bibliotecaLibros.exportadores.ExportadorLibrosCS
V;
import
nivel2Intermedio_bibliotecaLibros.exportadores.ExportadorLibrosXM
L;
import
nivel2Intermedio_bibliotecaLibros.exportadores.ExportadorLibrosJS
ON;
import nivel2Intermedio_bibliotecaLibros.modelo.Libro;

import
nivel3Avanzado_sistemaReservasHotel.exportadores.ExportadorReserv
asCSV;
import
nivel3Avanzado_sistemaReservasHotel.exportadores.ExportadorReserv
asXML;
import
nivel3Avanzado_sistemaReservasHotel.exportadores.ExportadorReserv
asJSON;
import nivel3Avanzado_sistemaReservasHotel.modelo.Cliente;
import nivel3Avanzado_sistemaReservasHotel.modelo.Habitacion;
import nivel3Avanzado_sistemaReservasHotel.modelo.Reserva;

import java.time.LocalDate;
import java.util.ArrayList;
import java.util.Scanner;

import static utils.Utils.*;

public class Main {
    // Nivel 1: Estudiantes
    private final static ArrayList<Estudiante> estudiantes = new
ArrayList<>();

    // Nivel 2: Libros
    private final static ArrayList<Libro> libros = new
ArrayList<>();

    // Nivel 3: Reservas hotel
    private final static ArrayList<Reserva> reservas = new
ArrayList<>();
}

```

```

public static void main(String[] args) {

    // Agrego 5 estudiantes de ejemplo
    estudiantes.add(new Estudiante(1, "Lucía", "García López",
20, 8.5));
        estudiantes.add(new Estudiante(2, "Carlos", "Martínez
Ruiz", 22, 7.2));
        estudiantes.add(new Estudiante(3, "Ana", "Sánchez
Fernández", 19, 9.0));
        estudiantes.add(new Estudiante(4, "Miguel", "Pérez Gómez",
21, 6.8));
        estudiantes.add(new Estudiante(5, "Sofía", "Hernández
Díaz", 20, 8.0));

    // Agrego libros de ejemplo
    libros.add(new Libro("978-84-123", "El Quijote", "Miguel
de Cervantes", "Ficción", 1605, 863, true, 150));
        libros.add(new Libro("978-84-456", "Cien años de soledad",
"Gabriel García Márquez", "Ficción", 1967, 471, false, 98));
        libros.add(new Libro("978-84-789", "Breve historia del
tiempo", "Stephen Hawking", "Ciencia", 1988, 256, true, 73));
        libros.add(new Libro("978-84-101", "Introducción a la
programación", "Ana Torres", "Ciencia", 2010, 350, true, 45));
        libros.add(new Libro("978-84-202", "Historia universal",
"VV.AA.", "Historia", 2005, 590, false, 60));
        libros.add(new Libro("978-84-303", "Los pilares de la
Tierra", "Ken Follett", "Ficción", 1989, 1076, true, 110));
        libros.add(new Libro("978-84-404", "Sapiens: De animales a
dioses", "Yuval Noah Harari", "Historia", 2011, 496, true, 85));
        libros.add(new Libro("978-84-505", "La teoría del todo",
"Stephen Hawking", "Ciencia", 2002, 224, false, 40));
        libros.add(new Libro("978-84-606", "Fundación", "Isaac
Asimov", "Ficción", 1951, 320, true, 135));
        libros.add(new Libro("978-84-707", "Un mundo feliz",
"Aldous Huxley", "Ficción", 1932, 288, false, 90));
        libros.add(new Libro("978-84-808", "El origen de las
especies", "Charles Darwin", "Ciencia", 1859, 502, true, 51));
        libros.add(new Libro("978-84-909", "La Segunda Guerra
Mundial", "Antony Beevor", "Historia", 2012, 1152, false, 58));
        libros.add(new Libro("978-84-010", "El nombre de la rosa",
"Umberto Eco", "Ficción", 1980, 672, true, 120));
        libros.add(new Libro("978-84-111", "Crónica de una muerte
anunciada", "Gabriel García Márquez", "Ficción", 1981, 128, true,
80));
        libros.add(new Libro("978-84-212", "Cosmos", "Carl Sagan",
"Ciencia", 1980, 384, false, 62));
}

```

```

        libros.add(new Libro("978-84-313", "Guns, Germs, and
Steel", "Jared Diamond", "Historia", 1997, 528, true, 70));

        // Creo clientes de ejemplo
        Cliente cliente1 = new Cliente(1, "Juan García",
"juan@email.com", "666111222");
        Cliente cliente2 = new Cliente(2, "María López",
"maria@email.com", "666222333");
        Cliente cliente3 = new Cliente(3, "Pedro Sánchez",
"pedro@email.com", "666333444");

        // Creo habitaciones de ejemplo
        Habitacion hab1 = new Habitacion(101, "Doble", 90.00,
false);
        Habitacion hab2 = new Habitacion(205, "Suite", 200.00,
true);
        Habitacion hab3 = new Habitacion(303, "Individual", 50.00,
true);
        Habitacion hab4 = new Habitacion(404, "Doble", 80.00,
true);
        Habitacion hab5 = new Habitacion(505, "Suite", 200.00,
false);

        // Agrego reservas de ejemplo
        reservas.add(new Reserva(1, cliente1, hab1,
LocalDate.parse("2025-10-20"), LocalDate.parse("2025-10-23"), 3,
270.00, "Confirmada"));
        reservas.add(new Reserva(2, cliente2, hab2,
LocalDate.parse("2025-10-21"), LocalDate.parse("2025-10-25"), 4,
800.00, "Confirmada"));
        reservas.add(new Reserva(3, cliente3, hab3,
LocalDate.parse("2025-10-18"), LocalDate.parse("2025-10-20"), 2,
100.00, "Completada"));
        reservas.add(new Reserva(4, cliente1, hab4,
LocalDate.parse("2025-10-22"), LocalDate.parse("2025-10-24"), 2,
160.00, "Cancelada"));
        reservas.add(new Reserva(5, cliente2, hab3,
LocalDate.parse("2025-10-25"), LocalDate.parse("2025-10-28"), 3,
150.00, "Confirmada"));
        reservas.add(new Reserva(6, cliente3, hab5,
LocalDate.parse("2025-10-26"), LocalDate.parse("2025-10-29"), 3,
600.00, "Confirmada"));
        reservas.add(new Reserva(7, cliente1, hab1,
LocalDate.parse("2025-10-28"), LocalDate.parse("2025-10-31"), 3,
270.00, "Completada"));

```

```

        reservas.add(new Reserva(8, cliente2, hab4,
LocalDate.parse("2025-10-27"), LocalDate.parse("2025-10-30"), 3,
240.00, "Confirmada"));
        reservas.add(new Reserva(9, cliente3, hab2,
LocalDate.parse("2025-10-29"), LocalDate.parse("2025-11-01"), 3,
600.00, "Confirmada"));
        reservas.add(new Reserva(10, cliente1, hab3,
LocalDate.parse("2025-10-30"), LocalDate.parse("2025-11-01"), 2,
100.00, "Confirmada"));

    while (true) {
        System.out.println("\n..... MENÚ PRINCIPAL .....");
        System.out.println("1. Exportar estudiantes");
        System.out.println("2. Exportar libros");
        System.out.println("3. Exportar reservas de hotel");
        System.out.println("0. Salir");
        int opcion = leerOpcion();

        System.out.println();
        switch (opcion) {
            case 1:
                menuExportarEstudiantes();
                break;
            case 2:
                menuExportarLibros();
                break;
            case 3:
                menuExportarReservasHotel();
                break;
            case 0:
                System.out.println("Cerrando programa...");
                sc.close();
                return;
            default:
                opcionInvalida();
                continue;
        }
        pausar();
    }

private static int mostrarMenuElegirFormato() {
    System.out.println("... ELIGE UN FORMATO ...");
    System.out.println("1. Exportar a CSV");
    System.out.println("2. Exportar a XML");
    System.out.println("3. Exportar a JSON");
    System.out.println("4. Exportar a TODOS los formatos");
    System.out.println("0. Volver al menú principal");
}

```

```

        return leerOpcion();
    }

    private static void menuExportarEstudiantes() {
        int opcion = mostrarMenuElegirFormato();
        String nombreArchivo = "estudiantes";
        boolean exito;
        System.out.println();
        switch (opcion) {
            case 1:
                exito =
                    ExportadorEstudiantesCSV.exportar(estudiantes, nombreArchivo);
                break;
            case 2:
                exito =
                    ExportadorEstudiantesXML.exportar(estudiantes, nombreArchivo);
                break;
            case 3:
                exito =
                    ExportadorEstudiantesJSON.exportar(estudiantes, nombreArchivo);
                break;
            case 4:
                int exitosos = 0;
                System.out.println("1/3 Exportando a CSV...");
                if (ExportadorEstudiantesCSV.exportar(estudiantes,
                        nombreArchivo)) exitosos++;
                System.out.println("2/3 Exportando a XML...");
                if (ExportadorEstudiantesXML.exportar(estudiantes,
                        nombreArchivo)) exitosos++;
                System.out.println("3/3 Exportando a JSON...");
                if
                    (ExportadorEstudiantesJSON.exportar(estudiantes, nombreArchivo))
                exitosos++;
                System.out.println("\nRESUMEN DE EXPORTACIÓN:");
                System.out.println("Formatos exportados: " +
                        exitosos + "/3\n");
                exito = true;
                if (exitosos != 3) System.out.println("⚠ No se
                        han podido crear todos los archivos.");
                break;
            case 0:
                System.out.println("Volviendo al menú
                        principal...");
                return;
            default:
                opcionInvalida();
                return;
        }
    }
}

```

```

        mostrarMensajeExito(exito);
    }

    private static void menuExportarLibros() {
        int opcion = mostrarMenuElegirFormato();
        String nombreArchivo = "libros";
        boolean exito;
        System.out.println();
        switch (opcion) {
            case 1:
                exito = ExportadorLibrosCSV.exportar(libros,
nombreArchivo);
                break;
            case 2:
                exito = ExportadorLibrosXML.exportar(libros,
nombreArchivo);
                break;
            case 3:
                exito = ExportadorLibrosJSON.exportar(libros,
nombreArchivo);
                break;
            case 4:
                int exitosos = 0;
                System.out.println("1/3 Exportando a CSV...");
                if (ExportadorLibrosCSV.exportar(libros,
nombreArchivo)) exitosos++;
                System.out.println("2/3 Exportando a XML...");
                if (ExportadorLibrosXML.exportar(libros,
nombreArchivo)) exitosos++;
                System.out.println("3/3 Exportando a JSON...");
                if (ExportadorLibrosJSON.exportar(libros,
nombreArchivo)) exitosos++;
                System.out.println("\nRESUMEN DE EXPORTACIÓN:");
                System.out.println("Formatos exportados: " +
exitosos + "/3\n");
                exito = true;
                if (exitosos != 3) System.out.println("⚠ No se
han podido crear todos los archivos.");
                break;
            case 0:
                System.out.println("Volviendo al menú
principal...");
                return;
            default:
                opcionInvalida();
                return;
        }
        mostrarMensajeExito(exito);
    }
}

```

```

    }

    private static void menuExportarReservasHotel() {
        int opcion = mostrarMenuElegirFormato();
        String nombreArchivo = "reservasHotel";
        boolean exito;
        System.out.println();
        switch (opcion) {
            case 1:
                exito = ExportadorReservasCSV.exportar(reservas,
nombreArchivo);
                break;
            case 2:
                exito = ExportadorReservasXML.exportar(reservas,
nombreArchivo);
                break;
            case 3:
                exito = ExportadorReservasJSON.exportar(reservas,
nombreArchivo);
                break;
            case 4:
                int exitosos = 0;
                System.out.println("1/3 Exportando a CSV...");
                if (ExportadorReservasCSV.exportar(reservas,
nombreArchivo)) exitosos++;
                System.out.println("2/3 Exportando a XML...");
                if (ExportadorReservasXML.exportar(reservas,
nombreArchivo)) exitosos++;
                System.out.println("3/3 Exportando a JSON...");
                if (ExportadorReservasJSON.exportar(reservas,
nombreArchivo)) exitosos++;
                System.out.println("\nRESUMEN DE EXPORTACIÓN:");
                System.out.println("Formatos exportados: " +
exitosos + "/3\n");
                exito = true;
                if (exitosos != 3) System.out.println("⚠ No se
han podido crear todos los archivos.");
                break;
            case 0:
                System.out.println("Volviendo al menú
principal...");
                return;
            default:
                opcionInvalida();
                return;
        }
        mostrarMensajeExito(exito);
    }
}

```

```
public static void mostrarMensajeExito(boolean exito) {  
    if (exito) {  
        System.out.println("¡EXPORTACIÓN COMPLETADA :) !");  
        System.out.println("(Ubicación: exportaciones/)");  
    } else {  
        System.out.println("No se ha podido completar la  
exportación.");  
    }  
}
```

EJECUCIÓN

Al ejecutar el programa aparece el **menú principal**

```
..... MENÚ PRINCIPAL .....\n1. Exportar estudiantes\n2. Exportar libros\n3. Exportar reservas de hotel\n0. Salir\n↳ Seleccione una opción:
```

Entramos en la opción 1. Exportar estudiantes

```
..... MENÚ PRINCIPAL .....
1. Exportar estudiantes
2. Exportar libros
3. Exportar reservas de hotel
0. Salir
↳ Seleccione una opción: 1

... ELIGE UN FORMATO ...
1. Exportar a CSV
2. Exportar a XML
3. Exportar a JSON
4. Exportar a TODOS los formatos
0. Volver al menú principal
↳ Seleccione una opción: |
```

Exportamos la lista de estudiantes en formato **CSV**

```
... ELIGE UN FORMATO ...
1. Exportar a CSV
2. Exportar a XML
3. Exportar a JSON
4. Exportar a TODOS los formatos
0. Volver al menú principal
↳ Seleccione una opción: 1
```

```
¡EXPORTACIÓN COMPLETADA :)!
(Ubicación: exportaciones/)
```

```
Presiona ENTER para continuar...
```

```
✗ exportaciones
  └── estudiantes_20251026_200339.csv
```

```
1 ID;Nombre;Apellidos;Edad;Nota
2 1;Lucia;García López;20;8.50
3 2;Carlos;Martínez Ruiz;22;7.20
4 3;Ana;Sánchez Fernández;19;9.00
5 4;Miguel;Pérez Gómez;21;6.80
6 5;Sofia;Hernández Diaz;20;8.00
7
8 # Nota media;7.90
9
```

Exportamos la lista de estudiantes en formato **XML**

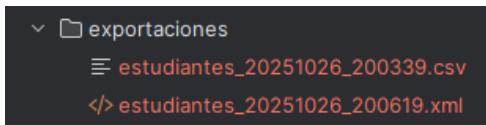
```

... ELIGE UN FORMATO ...
1. Exportar a CSV
2. Exportar a XML
3. Exportar a JSON
4. Exportar a TODOS los formatos
0. Volver al menú principal
↳ Seleccione una opción: 2

¡EXPORTACIÓN COMPLETADA :)!
(Ubicación: exportaciones)

Presiona ENTER para continuar...

```



```

</> estudiantes_20251026_200619.xml ×
1   <?xml version="1.0" encoding="UTF-8"?>
2   <cclase>
3     <metadata>
4       <version>1.0</version>
5       <fecha>26/10/2025 20:06:19</fecha>
6       <totalEstudiantes>5</totalEstudiantes>
7     </metadata>
8     <estudiantes>
9       <estudiante id="1">
10      <nombre>Lucía</nombre>
11      <apellidos>García López</apellidos>
12      <edad>20</edad>
13      <nota>8.50</nota>
14    </estudiante>
15    <estudiante id="2">
16      <nombre>Carlos</nombre>
17      <apellidos>Martínez Ruiz</apellidos>
18      <edad>22</edad>
19      <nota>7.20</nota>
20    </estudiante>
21    <estudiante id="3">
22      <nombre>Ana</nombre>
23      <apellidos>Sánchez Fernández</apellidos>
24      <edad>19</edad>
25      <nota>9.00</nota>
26    </estudiante>
27    <estudiante id="4">
28      <nombre>Miguel</nombre>
29      <apellidos>Pérez Gómez</apellidos>
30      <edad>21</edad>
31      <nota>6.80</nota>
32    </estudiante>
33    <estudiante id="5">
34      <nombre>Sofía</nombre>
35      <apellidos>Hernández Díaz</apellidos>
36      <edad>20</edad>
37      <nota>8.00</nota>
38    </estudiante>
39  </estudiantes>
40  <resumen>
41    <notaMedia>7.90</notaMedia>
42    <notaMaxima>9.00</notaMaxima>
43    <notaMinima>6.80</notaMinima>
44  </resumen>
45</cclase>
46

```

Exportamos la lista de estudiantes en formato **JSON**

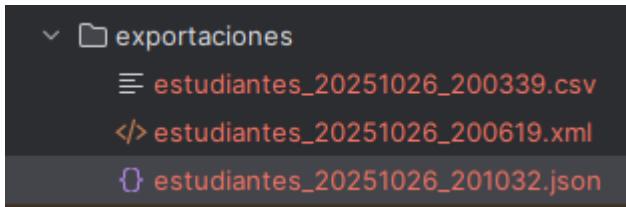
```

    ... ELIGE UN FORMATO ...
1. Exportar a CSV
2. Exportar a XML
3. Exportar a JSON
4. Exportar a TODOS los formatos
0. Volver al menú principal
↳ Seleccione una opción: 3

¡EXPORTACIÓN COMPLETADA :)!
(Ubicación: exportaciones/)

Presiona ENTER para continuar...

```



```

{
  "clase": {
    "metadata": {
      "version": "1.0",
      "fecha": "26/10/2025 20:10:32",
      "totalEstudiantes": 5
    },
    "estudiantes": [
      {
        "id": "1",
        "nombre": "Lucía",
        "apellidos": "García López",
        "edad": "20",
        "nota": "8.50"
      },
      {
        "id": "2",
        "nombre": "Carlos",
        "apellidos": "Martínez Ruiz",
        "edad": "22",
        "nota": "7.20"
      },
      {
        "id": "3",
        "nombre": "Ana",
        "apellidos": "Sánchez Fernández",
        "edad": "19",
        "nota": "9.00"
      },
      {
        "id": "4",
        "nombre": "Miguel",
        "apellidos": "Pérez Gómez",
        "edad": "21",
        "nota": "6.80"
      }
    ]
  }
}

```

A screenshot of a code editor showing the JSON file 'estudiantes_20251026_201032.json'. The code defines a class object with metadata and a list of five student objects. Each student object contains properties: id, nombre, apellidos, edad, and nota.

```
estudiantes_20251026_201032.json
1  "clase": {
2     "estudiantes": [
3         {
4             "id": "1",
5             "nombre": "Juan",
6             "apellidos": "García Pérez",
7             "edad": "19",
8             "nota": "8.50"
9         },
10        {
11            "id": "2",
12            "nombre": "Ana",
13            "apellidos": "Sánchez Fernández",
14            "edad": "19",
15            "nota": "9.00"
16        },
17        {
18            "id": "3",
19            "nombre": "Miguel",
20            "apellidos": "Pérez Gómez",
21            "edad": "21",
22            "nota": "6.80"
23        },
24        {
25            "id": "4",
26            "nombre": "Sofía",
27            "apellidos": "Hernández Díaz",
28            "edad": "20",
29            "nota": "8.00"
30        }
31    ],
32    "estadisticas": {
33        "notaMedia": 7.98,
34        "notaMaxima": 9.00,
35        "notaMinima": 6.80,
36        "aprobados": 3,
37        "suspenso": 0
38    }
39}
40}
```

Si seleccionamos la opción 4. Exportar a TODOS los formatos se generan los 3 archivos al mismo tiempo:

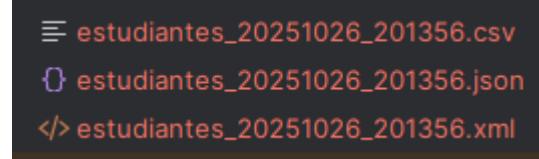
```
... ELIGE UN FORMATO ...
1. Exportar a CSV
2. Exportar a XML
3. Exportar a JSON
4. Exportar a TODOS los formatos
0. Volver al menú principal
↳ Seleccione una opción: 4

1/3 Exportando a CSV...
2/3 Exportando a XML...
3/3 Exportando a JSON...

RESUMEN DE EXPORTACIÓN:
Formatos exportados: 3/3

¡EXPORTACIÓN COMPLETADA :)!
(Ubicación: exportaciones/)

Presiona ENTER para continuar...
```



Opción ‘0. Volver al menú principal’

```
... ELIGE UN FORMATO ...
1. Exportar a CSV
2. Exportar a XML
3. Exportar a JSON
4. Exportar a TODOS los formatos
0. Volver al menú principal
↳ Seleccione una opción: 0

Volviendo al menú principal...

Presiona ENTER para continuar...

..... MENÚ PRINCIPAL .....
1. Exportar estudiantes
2. Exportar libros
3. Exportar reservas de hotel
0. Salir
↳ Seleccione una opción: |
```

Seleccionamos la opción ‘2. Exportar libros’

```
..... MENÚ PRINCIPAL .....
1. Exportar estudiantes
2. Exportar libros
3. Exportar reservas de hotel
0. Salir
↳ Seleccione una opción: 2
```

Exportamos todos los libros de la biblioteca

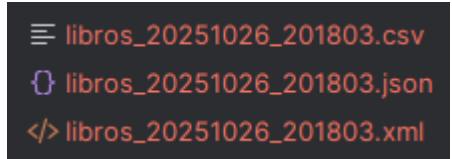
```
... ELIGE UN FORMATO ...
1. Exportar a CSV
2. Exportar a XML
3. Exportar a JSON
4. Exportar a TODOS los formatos
0. Volver al menú principal
↳ Seleccione una opción: 4

1/3 Exportando a CSV...
2/3 Exportando a XML...
3/3 Exportando a JSON...

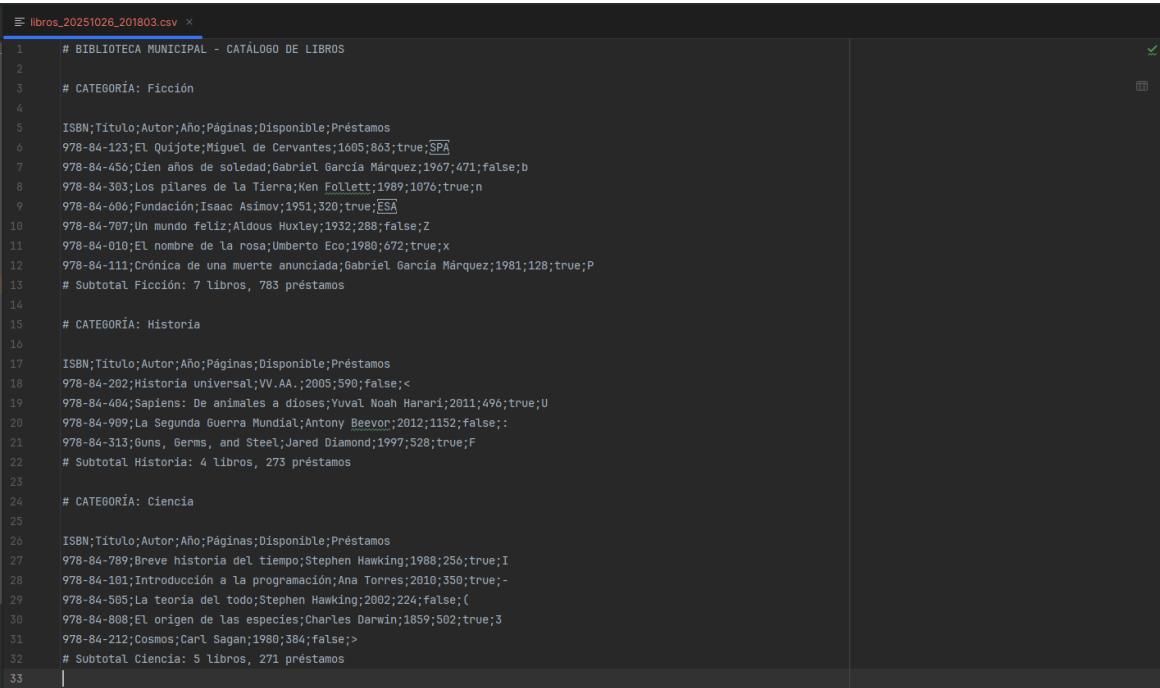
RESUMEN DE EXPORTACIÓN:
Formatos exportados: 3/3

¡EXPORTACIÓN COMPLETADA :)!
(Ubicación: exportaciones/)

Presiona ENTER para continuar...
```



Archivo libros CSV



```
# BIBLIOTECA MUNICIPAL - CATÁLOGO DE LIBROS
# CATEGORÍA: Ficción
ISBN;Título;Autor;Año;Páginas;Disponible;Préstamos
978-84-123;El Quijote;Miguel de Cervantes;1605;863;true;SPA
978-84-450;Cien años de soledad;Gabriel García Márquez;1967;471;false;b
978-84-303;Los pilares de la Tierra;Ken Follett;1989;1076;true;n
978-84-600;Fundación;Isaac Asimov;1951;320;true;ESA
978-84-707;Un mundo feliz;Aldous Huxley;1932;288;false;z
978-84-010;El nombre de la rosa;Umberto Eco;1980;672;true;x
978-84-111;Crónica de una muerte anunciada;Gabriel García Márquez;1981;128;true;P
# Subtotal Ficción: 7 libros, 783 préstamos

# CATEGORÍA: Historia
ISBN;Título;Autor;Año;Páginas;Disponible;Préstamos
978-84-202;Historia universal;VV.AA.;2005;599;false;<
978-84-404;Sapiens: De animales a dioses;Yuval Noah Harari;2011;496;true;U
978-84-909;La Segunda Guerra Mundial;Antony Beevor;2012;1152;false;:
978-84-313;Guns, Germs, and Steel;Jared Diamond;1997;528;true;F
# Subtotal Historia: 4 libros, 273 préstamos

# CATEGORÍA: Ciencia
ISBN;Título;Autor;Año;Páginas;Disponible;Préstamos
978-84-789;Breve historia del tiempo;Stephen Hawking;1988;256;true;I
978-84-101;Introducción a la programación;Ana Torres;2010;350;true;-
978-84-505;La teoría del todo;Stephen Hawking;2002;224;false;(
978-84-808;El origen de las especies;Charles Darwin;1859;502;true;3
978-84-212;Cosmos;Carl Sagan;1980;384;false;>
# Subtotal Ciencia: 5 libros, 271 préstamos
```

Archivo XML libros

```
</> libros_20251026_201803.xml
1   <?xml version="1.0" encoding="UTF-8"?>
2   <biblioteca>
3       <informacion>
4           <nombre>Biblioteca Municipal</nombre>
5           <fecha>26/10/2025</fecha>
6           <totalLibros>16</totalLibros>
7       </informacion>
8       <categorias>
9           <categoria nombre="Ficción" totalLibros="7">
10              <libro isbn="978-84-123" disponible="true">
11                  <titulo>El Quijote</titulo>
12                  <autor>Miguel de Cervantes</autor>
13                  <año>1605</año>
14                  <páginas>863</páginas>
15                  <prestamos>150</prestamos>
16              </libro>
17              <libro isbn="978-84-456" disponible="false">
18                  <titulo>Cien años de soledad</titulo>
19                  <autor>Gabriel García Márquez</autor>
20                  <año>1967</año>
21                  <páginas>471</páginas>
22                  <prestamos>98</prestamos>
23              </libro>
24              <libro isbn="978-84-303" disponible="true">
25                  <titulo>Los pilares de la Tierra</titulo>
26                  <autor>Ken Follett</autor>
27                  <año>1989</año>
28                  <páginas>1076</páginas>
29                  <prestamos>110</prestamos>
30              </libro>
31              <libro isbn="978-84-606" disponible="true">
32                  <titulo>Fundación</titulo>
33                  <autor>Isaac Asimov</autor>
34                  <año>1951</año>
35                  <páginas>320</páginas>
36                  <prestamos>135</prestamos>
37              </libro>
38          </categoria>
39      </categorias>
40  </biblioteca>
```

```
</> libros_20251026_201803.xml
1   <?xml version="1.0" encoding="UTF-8"?>
2   <biblioteca>
3       <informacion>
4           <nombre>Biblioteca Municipal</nombre>
5           <fecha>26/10/2025</fecha>
6           <totalLibros>16</totalLibros>
7       </informacion>
8       <categorias>
9           <categoria nombre="Ciencia" totalLibros="5">
10              <libro isbn="978-84-505" disponible="false">
11                  <titulo>La teoría del todo</titulo>
12                  <autor>Stephen Hawking</autor>
13                  <año>2002</año>
14                  <páginas>224</páginas>
15                  <prestamos>40</prestamos>
16              </libro>
17              <libro isbn="978-84-808" disponible="true">
18                  <titulo>El origen de las especies</titulo>
19                  <autor>Charles Darwin</autor>
20                  <año>1859</año>
21                  <páginas>502</páginas>
22                  <prestamos>51</prestamos>
23              </libro>
24              <libro isbn="978-84-212" disponible="false">
25                  <titulo>Cosmos</titulo>
26                  <autor>Carl Sagan</autor>
27                  <año>1980</año>
28                  <páginas>384</páginas>
29                  <prestamos>62</prestamos>
30              </libro>
31              <estadisticas>
32                  <totalPrestamos>271</totalPrestamos>
33                  <prestamosMedio>54.20</prestamosMedio>
34              </estadisticas>
35          </categoria>
36      </categorias>
37      <resumenGlobal>
38          <totalCategorias>3</totalCategorias>
39          <totalLibros>16</totalLibros>
40          <librosDisponibles>10</librosDisponibles>
41          <librosPrestados>6</librosPrestados>
42      </resumenGlobal>
43  </biblioteca>
```

Archivo JSON libros

```
libros_20251026_201803.json x
1 | [ 2 |   "biblioteca": { 3 |     "informacion": { 4 |       "nombre": "Biblioteca Municipal", 5 |       "fecha": "26/10/2025", 6 |       "totalLibros": 16 7 |     }, 8 |     "categorias": { 9 |       "Ficción": { 10 |         "totalLibros": 7, 11 |         "libros": [ 12 |           { 13 |             "isbn": "978-84-123", 14 |             "titulo": "El Quijote", 15 |             "autor": "Miguel de Cervantes", 16 |             "año": 1605, 17 |             "paginas": 863, 18 |             "disponible": true, 19 |             "prestamos": 150 20 |           }, 21 |           { 22 |             "isbn": "978-84-456", 23 |             "titulo": "Cien años de soledad", 24 |             "autor": "Gabriel García Márquez", 25 |             "año": 1967, 26 |             "paginas": 471, 27 |             "disponible": false, 28 |             "prestamos": 98 29 |           }, 30 |           { 31 |             "isbn": "978-84-303", 32 |             "titulo": "Los pilares de la Tierra", 33 |             "autor": "Ken Follett", 34 |             "año": 1989, 35 |             "paginas": 1076, 36 |             "disponible": true, 37 |             "prestamos": 110 38 |           } 39 |         ] 40 |       } 41 |     } 42 |   } 43 | }
```

```
libros_20251026_201803.json x
2     "biblioteca": {
3         "categorias": {
4             "Ficción": {
5                 "libros": [
6                     {
7                         "isbn": "978-84-111",
8                         "titulo": "Crónica de una muerte anunciada",
9                         "autor": "Gabriel García Márquez",
10                        "año": 1981,
11                        "paginas": 128,
12                        "disponible": true,
13                        "prestamos": 80
14                    }
15                ],
16                "estadisticas": {
17                    "totalPrestamos": 783,
18                    "prestamosMedio": 111.86,
19                    "libroMasPrestado": "El Quijote"
20                }
21            },
22            "Historia": {
23                "totalLibros": 4,
24                "libros": [
25                    {
26                        "isbn": "978-84-202",
27                        "titulo": "Historia universal",
28                        "autor": "V.V.AA.",
29                        "año": 2005,
30                        "paginas": 590,
31                        "disponible": false,
32                        "prestamos": 60
33                    },
34                    {
35                        "isbn": "978-84-404",
36                        "titulo": "Sapiens: De animales a dioses",
37                        "autor": "Yuval Noah Harari",
38                    }
39                ]
40            }
41        }
42    }
43}
```

```
libros_20251026_201803.json x
2   "biblioteca": {
8     "categorias": {
82       "Historia": {
84         "libros": [
112           {
120             "isbn": "978-84-789",
121             "titulo": "Breve historia del tiempo",
122             "autor": "Stephen Hawking",
123             "año": 1988,
124             "paginas": 256,
125             "disponible": true,
126             "prestamos": 73
127           },
128           {
129             "isbn": "978-84-101",
130             "titulo": "Introducción a la programación",
131             "autor": "Ana Torres",
132             "año": 2010,
133             "paginas": 350,
134             "disponible": true,
135             "prestamos": 45
136           },
137           {
138             "isbn": "978-84-505",
139             "titulo": "La teoría del todo",
140             "autor": "Stephen Hawking",
141             "año": 2011,
142             "paginas": 320,
143             "disponible": true,
144             "prestamos": 32
145           }
146         ]
147       },
148       "Ciencia": {
149         "totalLibros": 5,
150         "libros": [
151           {
152             "isbn": "978-84-212",
153             "titulo": "Cosmos",
154             "autor": "Carl Sagan",
155             "año": 1980,
156             "paginas": 384,
157             "disponible": false,
158             "prestamos": 62
159           }
160         ]
161       }
162     },
163     "estadisticas": {
164       "totalPrestamos": 273,
165       "prestamosMedio": 68.25,
166       "libroMasPrestado": "Sapiens: De animales a dioses"
167     }
168   },
169   "resumenGlobal": {
170     "totalCategorias": 3,
171     "totalLibros": 10,
172     "librosDisponibles": 10,
173     "librosPrestados": 6,
174     "totalPrestamosHistorico": 1327
175   }
176 }
```

```
libros_20251026_201803.json x
2   "biblioteca": {
8     "categorias": {
82       "Ciencia": {
84         "libros": [
112           {
120             "isbn": "978-84-212",
121             "titulo": "Cosmos",
122             "autor": "Carl Sagan",
123             "año": 1980,
124             "paginas": 384,
125             "disponible": false,
126             "prestamos": 62
127           },
128           {
129             "isbn": "978-84-212",
130             "titulo": "Breve historia del tiempo",
131             "autor": "Stephen Hawking",
132             "año": 1988,
133             "paginas": 256,
134             "disponible": true,
135             "prestamos": 73
136           },
137           {
138             "isbn": "978-84-101",
139             "titulo": "Introducción a la programación",
140             "autor": "Ana Torres",
141             "año": 2010,
142             "paginas": 350,
143             "disponible": true,
144             "prestamos": 45
145           }
146         ]
147       },
148       "estadisticas": {
149         "totalPrestamos": 271,
150         "prestamosMedio": 54.20,
151         "libroMasPrestado": "Breve historia del tiempo"
152       }
153     },
154     "resumenGlobal": {
155       "totalCategorias": 3,
156       "totalLibros": 10,
157       "librosDisponibles": 10,
158       "librosPrestados": 6,
159       "totalPrestamosHistorico": 1327
160     }
161   }
162 }
```

Por último, vamos a la opción ‘3. Exportar reservas hotel’

```
..... MENÚ PRINCIPAL .....
1. Exportar estudiantes
2. Exportar libros
3. Exportar reservas de hotel
0. Salir
↳ Seleccione una opción: 3
```

Exportamos las reservas del hotel a TODOS los formatos

```
... ELIGE UN FORMATO ...
1. Exportar a CSV
2. Exportar a XML
3. Exportar a JSON
4. Exportar a TODOS los formatos
0. Volver al menú principal
↳ Seleccione una opción: 4

1/3 Exportando a CSV...
2/3 Exportando a XML...
3/3 Exportando a JSON...

RESUMEN DE EXPORTACIÓN:
Formatos exportados: 3/3

¡EXPORTACIÓN COMPLETADA :)!
(Ubicación: exportaciones/)

Presiona ENTER para continuar...
```

```
☰ reservasHotel_20251026_202811.csv
☰ reservasHotel_20251026_202811.json
</> reservasHotel_20251026_202811.xml
```

Archivo reservasHotel CSV

```
reservasHotel_20251026_202811.csv x
1 ID;ClienteNombre;ClienteEmail;ClienteTelefono;HabitacionNum;TipoHabitacion;PrecioNoche;FechaEntrada;FechaSalida;Noches;PrecioTotal;Estado
2 1;Juan García;juan@email.com;666111222;101;Doble;90.00;20/10/2025;23/10/2025;3;270.00;Confirmada
3 2;María López;maria@email.com;666222333;205;Suite;200.00;21/10/2025;25/10/2025;4;800.00;Confirmada
4 3;Pedro Sánchez;pedro@email.com;666333444;303;Individual;50.00;18/10/2025;29/10/2025;2;100.00;Completada
5 4;Juan García;juan@email.com;666111222;404;Doble;80.00;22/10/2025;24/10/2025;2;160.00;Cancelada
6 5;María López;maria@email.com;666222333;303;Individual;50.00;25/10/2025;28/10/2025;3;150.00;Confirmada
7 6;Pedro Sánchez;pedro@email.com;666333444;505;Suite;200.00;26/10/2025;29/10/2025;3;600.00;Confirmada
8 7;Juan García;juan@email.com;666111222;101;Doble;90.00;28/10/2025;31/10/2025;3;270.00;Completada
9 8;María López;maria@email.com;666222333;404;Doble;80.00;27/10/2025;30/10/2025;3;240.00;Confirmada
10 9;Pedro Sánchez;pedro@email.com;666333444;205;Suite;200.00;29/10/2025;01/11/2025;3;600.00;Confirmada
11 10;Juan García;juan@email.com;666111222;303;Individual;50.00;30/10/2025;01/11/2025;2;100.00;Confirmada
12
13 # Total reservas: 10
14 # Total ingresos (reservas): 3290.00
15 # Confirmadas: 7 | Canceladas: 1 | Completadas: 2
16
```

Archivo reservasHotel XML

```
reservasHotel_20251026_202811.csv      O reservasHotel_20251026_202811.json      </> reservasHotel_20251026_202811.xml x
1 <?xml version="1.0" encoding="UTF-8"?>
2 <hotel>
3   <informacion>
4     <nOMBRE>Hotel Paradise Iris Pérez</nOMBRE>
5     <fecha>26/10/2025</fecha>
6   </informacion>
7   <RESERVAS totalReservas="10">
8     <RESERVA id="1" estado="Confirmada">
9       <CLIENTE>
10         <id>1</id>
11         <nOMBRE>Juan García</nOMBRE>
12         <email>juan@email.com</email>
13         <telefono>666111222</telefono>
14       </CLIENTE>
15       <HABITACION numero="101" tipo="Doble">
16         <precioPorNoche>90.00</precioPorNoche>
17         <disponible>false</disponible>
18       </HABITACION>
19       <FECHAS>
20         <entrada>20/10/2025</entrada>
21         <salida>23/10/2025</salida>
22         <noches>3</noches>
23       </FECHAS>
24       <PRECIO>
25         <total>270.00</total>
26         <porNoche>90.00</porNoche>
27       </PRECIO>
28     </RESERVA>
29     <RESERVA id="2" estado="Confirmada">
30       <CLIENTE>
31         <id>2</id>
32         <nOMBRE>María López</nOMBRE>
33         <email>maria@email.com</email>
34         <telefono>666222333</telefono>
35       </CLIENTE>
36     </RESERVA>
37   </RESERVAS>
38 </hotel>
```

```
reservasHotel_20251026_202811.json      resertasHotel_20251026_202811.xml ×
```

```
2   <hotel>
7     <reservas totalReservas="10">
197       <reserva id="10" estado="Confirmada">
208         <fechas>
212           </fechas>
213           <precio>
214             <total>100.00</total>
215             <porNoche>50.00</porNoche>
216           </precio>
217         </reserva>
218       </reservas>
219       <estadisticas>
220         <porTipoHabitacion>
221           <Suite>
222             <totalReservas>3</totalReservas>
223             <ingresos>2000.00</ingresos>
224           </Suite>
225           <Doble>
226             <totalReservas>4</totalReservas>
227             <ingresos>940.00</ingresos>
228           </Doble>
229           <Individual>
230             <totalReservas>3</totalReservas>
231             <ingresos>350.00</ingresos>
232           </Individual>
233         </porTipoHabitacion>
234         <porEstado>
235           <Confirmada>7</Confirmada>
236           <Completada>2</Completada>
237           <Cancelada>1</Cancelada>
238         </porEstado>
239         <resumen>
240           <totalReservas>10</totalReservas>
241           <ingresosTotal>3290.00</ingresosTotal>
242           <nochesReservadas>28</nochesReservadas>
243         </resumen>
244       </estadisticas>
245     </hotel>
246
```

Archivo reservasHotel JSON

```
{} reservasHotel_20251026_202811.json ×
 2     "hotel": {
 51         "reservas": [
142             {
150                 "estado": "Confirmada"
151             }
152         ],
153         "estadisticas": {
154             "porTipoHabitacion": {
155                 "Suite": {
156                     "totalReservas": 3,
157                     "ingresos": 2000.00,
158                     "porcentaje": 30.00
159                 },
160                 "Doble": {
161                     "totalReservas": 4,
162                     "ingresos": 940.00,
163                     "porcentaje": 40.00
164                 },
165                 "Individual": {
166                     "totalReservas": 3,
167                     "ingresos": 350.00,
168                     "porcentaje": 30.00
169                 }
170             },
171             "porEstado": {
172                 "Confirmada": 7,
173                 "Completada": 2,
174                 "Cancelada": 1
175             },
176             "resumen": {
177                 "totalReservas": 10,
178                 "ingresosTotal": 3290.00,
179                 "nochesReservadas": 28,
180                 "ocupacionMedia": 280.00
181             }
182         }
183     }
184 }
```

Gracias por ver :)

Enlace al proyecto en GitHub:

https://github.com/IrisCampusFP/ActividadesAccesoADatos/tree/main/UD1-Persistencia_en_Ficheros/Act1.4-Exportacion_de_datos/Act1.4-Exportacion_de_datos