PROGRAMACIÓN II

Trabajo Práctico 6: Colecciones y Sistema de Stock

Alumno: Mauro Gaspar
Comisión: 4

Caso Práctico 1

Descripción general:

Se debe desarrollar un sistema de stock que permita gestionar productos en una tienda, controlando su disponibilidad, precios y categorías. La información se modelará utilizando clases, colecciones dinámicas y enumeraciones en Java.

Enum CategoriaProducto

```
public enum CategoriaProducto {

    ALIMENTOS("Productos comestibles"),
    ELECTRONICA("Dispositivos electrónicos"),
    ROPA("Prendas de vestir"),
    HOGAR("Artículos para el hogar"),
    LIBROS("Articulos de Librería"),
    JUGUETES("Articulos para niños");

    private final String descripcion;

    CategoriaProducto(String descripcion) {
        this.descripcion = descripcion;
    }

    public String getDescripcion() {
        return descripcion;
    }
}
```

Clase Producto:

```java
public class Producto {

    private String id;
    private String nombre;
    private double precio;
    private int cantidad;
    private CategoriaProducto categoria;

    // Constructor completo
    public Producto(String id, String nombre, double precio, int cantidad, CategoriaProducto categoria) {
        this.id = id;
        this.nombre = nombre;
        this.precio = precio;
        this.cantidad = cantidad;
        this.categoria = categoria;
    }

    // Constructor sobrecargado (cantidad por defecto = 0)
    public Producto(String id, String nombre, double precio, CategoriaProducto categoria) {
        this(id, nombre, precio, 0, categoria);
    }

    // Getters y setters
    public String getId() { return id; }
    public String getNombre() { return nombre; }
    public double getPrecio() { return precio; }
    public int getCantidad() { return cantidad; }
    public CategoriaProducto getCategoria() { return categoria; }

    public void setNombre(String nombre) { this.nombre = nombre; }
    public void setPrecio(double precio) { this.precio = precio; }
    public void setCantidad(int cantidad) { this.cantidad = cantidad; }
    public void setCategoria(CategoriaProducto categoria) { this.categoria = categoria; }

    // Muestra info en consola
    public void mostrarInfo() {
        System.out.println(this.toString());
    }

    @Override
    public String toString() {
        return String.format("ID: %s | Nombre: %s | Precio: $%.2f | Cantidad: %d | Categoria: %s",
                id, nombre, precio, cantidad, categoria);
    }
}
```

Clase Inventario:

```
package TP6;
import java.util.ArrayList;
import java.util.Comparator;
import java.util.List;
import java.util.stream.Collectors;

/**
 *
 * @author Mauro
 */
public class Inventario {

    private ArrayList<Producto> productos;

    // Se usa ArrayList<Producto> para manipular una colección dinámica (agregar, eliminar, iterar)
    public Inventario() {
        this.productos = new ArrayList<>();
    }

    // 1. agregarProducto
    public void agregarProducto(Producto p) {
        // evitar ids duplicados (opcional)
        if (buscarProductoPorId(p.getId()) != null) {
            System.out.println("Ya existe un producto con ID " + p.getId() + ". No se agregó.");
            return;
        }
        productos.add(p);
    }

    // 2. listarProductos
    public void listarProductos() {
        if (productos.isEmpty()) {
            System.out.println("Inventario vacío.");
            return;
        }
        System.out.println("Listado de productos:");
        for (Producto p : productos) {
            p.mostrarInfo();
        }
    }

    // 3. buscarProductoPorId
    public Producto buscarProductoPorId(String id) {
        for (Producto p : productos) {
            if (p.getId().equalsIgnoreCase(id)) {
                return p;
            }
        }
```

```java
// 4. eliminarProducto
public boolean eliminarProducto(String id) {
    Producto p = buscarProductoPorId(id);
    if (p != null) {
        productos.remove(p);
        return true;
    }
    return false;
}

// 5. actualizarStock
public boolean actualizarStock(String id, int nuevaCantidad) {
    Producto p = buscarProductoPorId(id);
    if (p != null) {
        p.setCantidad(nuevaCantidad);
        return true;
    }
    return false;
}

// 6. filtrarPorCategoria
public List<Producto> filtrarPorCategoria(CategoriaProducto categoria) {
    return productos.stream()
            .filter(p -> p.getCategoria() == categoria)
            .collect(Collectors.toList());
}

// 7. obtenerTotalStock
public int obtenerTotalStock() {
    int total = 0;
    for (Producto p : productos) total += p.getCantidad();
    return total;
}

// 8. obtenerProductoConMayorStock
public Producto obtenerProductoConMayorStock() {
    return productos.stream()
            .max(Comparator.comparingInt(Producto::getCantidad))
            .orElse(null);
}

// 9. filtrarProductosPorPrecio
public List<Producto> filtrarProductosPorPrecio(double min, double max) {
    return productos.stream()
            .filter(p -> p.getPrecio() >= min && p.getPrecio() <= max)
            .collect(Collectors.toList());
}
```

Clase SistemaStockMain:

```
public class SIstemaStockMain {

    public static void main(String[] args) {

        // Ejecuta las tareas solicitadas en orden y muestra resultados por consola.
        Inventario inventario = new Inventario();

        // 1. Crear al menos cinco productos con diferentes categorías y agregarlos al inventario.
        Producto p1 = new Producto("P001", "Arroz Integral 1kg", 850.0, 50, CategoriaProducto.ALIMENTOS);
        Producto p2 = new Producto("P002", "Smartwatch Pro S9", 35000.0, 15, CategoriaProducto.ELECTRONICA
        Producto p3 = new Producto("P003", "Pantalón Jean", 7800.0, 30, CategoriaProducto.ROPA);
        Producto p4 = new Producto("P004", "Licuadora", 4150.0, 12, CategoriaProducto.HOGAR);
        Producto p5 = new Producto("P005", "Fideos Secos 500g", 620.0, 90, CategoriaProducto.ALIMENTOS);
        Producto p6 = new Producto("P006", "Auriculares Inalámbricos", 1500.0, 45, CategoriaProducto.ELECT
        Producto p7 = new Producto("P007", "Camisa Formal Blanca", 9200.0, 20, CategoriaProducto.ROPA);
        Producto p8 = new Producto("P008", "Atril de Lectura", 2100.0, 8, CategoriaProducto.LIBROS);
        Producto p9 = new Producto("P009", "Set de Bloques Armables", 2990.0, 60, CategoriaProducto.JUGUET
        Producto p10 = new Producto("P010", "Toallas de Baño Algodón", 1850.0, 40, CategoriaProducto.HOGAR

        inventario.agregarProducto(p1);
        inventario.agregarProducto(p2);
        inventario.agregarProducto(p3);
        inventario.agregarProducto(p4);
        inventario.agregarProducto(p5);
        inventario.agregarProducto(p6);
        inventario.agregarProducto(p7);
        inventario.agregarProducto(p8);
        inventario.agregarProducto(p9);
        inventario.agregarProducto(p10);

        // 2. Listar todos los productos mostrando su información y categoría.
        System.out.println("\n-- 2. Listar productos --");
        inventario.listarProductos();

        // 3. Buscar un producto por ID y mostrar su información.
        System.out.println("\n-- 3. Buscar producto por ID 'P007' --");
        Producto buscado = inventario.buscarProductoPorId("P007");
        if (buscado != null) buscado.mostrarInfo();
        else System.out.println("Producto no encontrado.");

        // 4. Filtrar y mostrar productos que pertenezcan a una categoría específica.
        System.out.println("\n-- 4. Filtrar por categoria ELECTRONICA --");
        List<Producto> electronicos = inventario.filtrarPorCategoria(CategoriaProducto.ELECTRONICA);
        electronicos.forEach(Producto::mostrarInfo);
```

```
        // 3. Buscar un producto por ID y mostrar su información.
        System.out.println("\n-- 3. Buscar producto por ID 'P007' --");
        Producto buscado = inventario.buscarProductoPorId("P007");
        if (buscado != null) buscado.mostrarInfo();
        else System.out.println("Producto no encontrado.");

        // 4. Filtrar y mostrar productos que pertenezcan a una categoria especifica.
        System.out.println("\n-- 4. Filtrar por categoria ELECTRONICA --");
        List<Producto> electronicos = inventario.filtrarPorCategoria(CategoriaProducto.ELECTRONICA);
        electronicos.forEach(Producto::mostrarInfo);

        // 5. Eliminar un producto por su ID y listar los productos restantes.
        System.out.println("\n-- 5. Eliminar producto 'P004' (Licuadora) y listar resto --");
        boolean eliminado = inventario.eliminarProducto("P004");
        System.out.println("Eliminado P004? " + eliminado);
        inventario.listarProductos();

        // 6. Actualizar el stock de un producto existente.
        System.out.println("\n-- 6. Actualizar stock de 'P002' (Smartwatch Pro S9) a 25 unidades --");
        boolean actualizado = inventario.actualizarStock("P002", 25);
        System.out.println("Actualizado P002? " + actualizado);
        System.out.println("P002 ahora: " + inventario.buscarProductoPorId("P002"));

        // 7. Mostrar el total de stock disponible.
        System.out.println("\n-- 7. Total de stock disponible --");
        System.out.println("Total stock: " + inventario.obtenerTotalStock());

        // 8. Obtener y mostrar el producto con mayor stock.
        System.out.println("\n-- 8. Producto con mayor stock --");
        Producto mayor = inventario.obtenerProductoConMayorStock();
        if (mayor != null) mayor.mostrarInfo();

        // 9. Filtrar productos con precios entre $1000 y $3000.
        System.out.println("\n-- 9. Filtrar por precio entre $1000 y $3000 --");
        List<Producto> rango = inventario.filtrarProductosPorPrecio(1000, 3000);
        rango.forEach(Producto::mostrarInfo);

        // 10. Mostrar las categorías disponibles con sus descripciones.
        System.out.println("\n-- 10. Categorías disponibles --");
        inventario.mostrarCategoriasDisponibles();
    }
}
```

Consola:

```
TP6 (run) ×    NetBeansProjects - D:\Mauro\Documents\NetBeansProjects ×
run:

-- 2. Listar productos --
Listado de productos:
ID: P001 | Nombre: Arroz Integral 1kg | Precio: $850,00 | Cantidad: 50 | Categoria: ALIMENTOS
ID: P002 | Nombre: Smartwatch Pro S9 | Precio: $35000,00 | Cantidad: 15 | Categoria: ELECTRONICA
ID: P003 | Nombre: Pantalón Jean | Precio: $7800,00 | Cantidad: 30 | Categoria: ROPA
ID: P004 | Nombre: Licuadora | Precio: $4150,00 | Cantidad: 12 | Categoria: HOGAR
ID: P005 | Nombre: Fideos Secos 500g | Precio: $620,00 | Cantidad: 90 | Categoria: ALIMENTOS
ID: P006 | Nombre: Auriculares Inalámbricos | Precio: $1500,00 | Cantidad: 45 | Categoria: ELECTRONICA
ID: P007 | Nombre: Camisa Formal Blanca | Precio: $9200,00 | Cantidad: 20 | Categoria: ROPA
ID: P008 | Nombre: Atril de Lectura | Precio: $2100,00 | Cantidad: 8 | Categoria: LIBROS
ID: P009 | Nombre: Set de Bloques Armables | Precio: $2990,00 | Cantidad: 60 | Categoria: JUGUETES
ID: P010 | Nombre: Toallas de Baño Algodón | Precio: $1850,00 | Cantidad: 40 | Categoria: HOGAR

-- 3. Buscar producto por ID 'P007' --
ID: P007 | Nombre: Camisa Formal Blanca | Precio: $9200,00 | Cantidad: 20 | Categoria: ROPA

-- 4. Filtrar por categoria ELECTRONICA --
ID: P002 | Nombre: Smartwatch Pro S9 | Precio: $35000,00 | Cantidad: 15 | Categoria: ELECTRONICA
ID: P006 | Nombre: Auriculares Inalámbricos | Precio: $1500,00 | Cantidad: 45 | Categoria: ELECTRONICA

-- 5. Eliminar producto 'P004' (Licuadora) y listar resto --
Eliminado P004? true
Listado de productos:
ID: P001 | Nombre: Arroz Integral 1kg | Precio: $850,00 | Cantidad: 50 | Categoria: ALIMENTOS
ID: P002 | Nombre: Smartwatch Pro S9 | Precio: $35000,00 | Cantidad: 15 | Categoria: ELECTRONICA
ID: P003 | Nombre: Pantalón Jean | Precio: $7800,00 | Cantidad: 30 | Categoria: ROPA
ID: P005 | Nombre: Fideos Secos 500g | Precio: $620,00 | Cantidad: 90 | Categoria: ALIMENTOS
ID: P006 | Nombre: Auriculares Inalámbricos | Precio: $1500,00 | Cantidad: 45 | Categoria: ELECTRONICA
ID: P007 | Nombre: Camisa Formal Blanca | Precio: $9200,00 | Cantidad: 20 | Categoria: ROPA
ID: P008 | Nombre: Atril de Lectura | Precio: $2100,00 | Cantidad: 8 | Categoria: LIBROS
ID: P009 | Nombre: Set de Bloques Armables | Precio: $2990,00 | Cantidad: 60 | Categoria: JUGUETES
ID: P010 | Nombre: Toallas de Baño Algodón | Precio: $1850,00 | Cantidad: 40 | Categoria: HOGAR
-- 6. Actualizar stock de 'P002' (Smartwatch Pro S9) a 25 unidades --
Actualizado P002? true
P002 ahora: ID: P002 | Nombre: Smartwatch Pro S9 | Precio: $35000,00 | Cantidad: 25 | Categoria: ELECTRONICA

-- 7. Total de stock disponible --
Total stock: 368

-- 8. Producto con mayor stock --
ID: P005 | Nombre: Fideos Secos 500g | Precio: $620,00 | Cantidad: 90 | Categoria: ALIMENTOS

-- 9. Filtrar por precio entre $1000 y $3000 --
ID: P006 | Nombre: Auriculares Inalámbricos | Precio: $1500,00 | Cantidad: 45 | Categoria: ELECTRONICA
ID: P008 | Nombre: Atril de Lectura | Precio: $2100,00 | Cantidad: 8 | Categoria: LIBROS
ID: P009 | Nombre: Set de Bloques Armables | Precio: $2990,00 | Cantidad: 60 | Categoria: JUGUETES
ID: P010 | Nombre: Toallas de Baño Algodón | Precio: $1850,00 | Cantidad: 40 | Categoria: HOGAR

-- 10. Categorías disponibles --
Categorías disponibles:
- ALIMENTOS : Productos comestibles
- ELECTRONICA : Dispositivos electrónicos
- ROPA : Prendas de vestir
- HOGAR : Artículos para el hogar
- LIBROS : Articulos de Librería
- JUGUETES : Articulos para niños
BUILD SUCCESSFUL (total time: 0 seconds)
```

Ejercicio Propuesto 2: Biblioteca y Libros

Se debe desarrollar un sistema para gestionar una biblioteca, en la cual se registren los libros disponibles y sus autores. La relación central es de composición 1 a N: una Biblioteca contiene múltiples Libros, y cada Libro pertenece obligatoriamente a una Biblioteca. Si la Biblioteca se elimina, también se eliminan sus Libros.

Clase Autor:

```java
public class Autor {

    private String nombre;
    private String nacionalidad;

    public Autor(String nombre, String nacionalidad) {
        this.nombre = nombre;
        this.nacionalidad = nacionalidad;
    }

    public String getNombre() {
        return nombre;
    }

    public String getNacionalidad() {
        return nacionalidad;
    }

    @Override
    public String toString() {
        return nombre + " (" + nacionalidad + ")";
    }

}
```

Clase Libro:

```java
public class Libro {

    private String isbn;
    private String titulo;
    private Autor autor;
    private int anioPublicacion;
    private String genero;
    private boolean disponible;

    public Libro(String isbn, String titulo, Autor autor, int anioPublicacion, String genero, boolean disp
        this.isbn = isbn;
        this.titulo = titulo;
        this.autor = autor;
        this.anioPublicacion = anioPublicacion;
        this.genero = genero;
        this.disponible = disponible;
    }

    // Getters y Setters
    public String getIsbn() { return isbn; }
    public String getTitulo() { return titulo; }
    public Autor getAutor() { return autor; }
    public int getAnioPublicacion() { return anioPublicacion; }
    public String getGenero() { return genero; }
    public boolean isDisponible() { return disponible; }

    public void setDisponible(boolean disponible) { this.disponible = disponible; }

    // Mostrar info completa
    public void mostrarInfo() {
        System.out.printf(
                "ISBN: %s | Título: %s | Autor: %s | Año: %d | Género: %s | Disponible: %s%n",
                isbn, titulo, autor.toString(), anioPublicacion, genero, disponible ? "Si" : "No"
        );
    }

   @Override
    public String toString() {
        return String.format("[%s] %s - %s (%d) - %s [%s]",
                isbn, titulo, autor.getNombre(), anioPublicacion, genero,
                disponible ? "Disponible" : "Prestado");
    }
}
```

Clase Biblioteca:

```java
public class Biblioteca {

    private ArrayList<Libro> libros;

    public Biblioteca() {
        libros = new ArrayList<>();
    }

    // 1. Agregar libro
    public void agregarLibro(Libro libro) {
        if (buscarPorISBN(libro.getIsbn()) != null) {
            System.out.println("Alerta!! Ya existe un libro con el mismo ISBN.");
            return;
        }
        libros.add(libro);
    }

    // 2. Listar todos los libros
    public void listarLibros() {
        if (libros.isEmpty()) {
            System.out.println("No hay libros cargados.");
        } else {
            System.out.println("Listado de libros:");
            for (Libro l : libros) l.mostrarInfo();
        }
    }

    // 3. Buscar libro por ISBN
    public Libro buscarPorISBN(String isbn) {
        for (Libro l : libros) {
            if (l.getIsbn().equalsIgnoreCase(isbn)) {
                return l;
            }
        }
        return null;
    }
```

```java
// 4. Eliminar libro por ISBN
public boolean eliminarLibro(String isbn) {
    Libro l = buscarPorISBN(isbn);
    if (l != null) {
        libros.remove(l);
        return true;
    }
    return false;
}

// 5. Filtrar por año
public List<Libro> filtrarPorAnio(int anio) {
    return libros.stream()
            .filter(l -> l.getAnioPublicacion() == anio)
            .collect(Collectors.toList());
}

// 6. Contar total de libros
public int contarLibros() {
    return libros.size();
}

// 7. Filtrar libros por autor
public List<Libro> filtrarPorAutor(Autor autor) {
    return libros.stream()
        .filter(l -> l.getAutor().equals(autor))
        .collect(Collectors.toList());


// 7. Listar autores de los libros disponibles
public void listarAutoresDisponibles() {
    List<String> autores = libros.stream()
            .filter(Libro::isDisponible)
            .map(l -> l.getAutor().toString())
            .distinct()
            .collect(Collectors.toList());

    System.out.println("Autores con libros disponibles:");
    for (String a : autores) {
        System.out.println("- " + a);
    }
}
```

Clase SistemaBibliotecaMain:

```
public class SistemaBibliotecaMain {

    public static void main(String[] args) {
        // 1. Creamos una biblioteca.
        Biblioteca biblioteca = new Biblioteca();
        System.out.println("--- INICIO DEL SISTEMA DE BIBLIOTECA ---");

        // 2. Crear al menos tres autores
        Autor autor1 = new Autor("J. D. Perón", "Argentino");
        Autor autor2 = new Autor("J.R.R. Tolkien", "Británico");
        Autor autor3 = new Autor("Philip K. Dick", "Estadounidense");
        Autor autor4 = new Autor("Nicolás Márquez", "Argentino");
        Autor autor5 = new Autor("Agustín Laje", "Argentino");
        Autor autor6 = new Autor("Salvador Borrego", "Mexicano");

        // 3. Agregar 5 libros asociados a alguno de los Autores
        System.out.println("\n--- 3. AGREGANDO LIBROS ---");
        biblioteca.agregarLibro(new Libro("935-1", "La hora de los pueblos", autor1, 1968, "Ensayo", true)
        biblioteca.agregarLibro(new Libro("935-2", "El Hobbit", autor2, 1937, "Fantasía", true));
        biblioteca.agregarLibro(new Libro("935-3", "1984", new Autor("George Orwell", "Británico"), 1949,
        biblioteca.agregarLibro(new Libro("935-4", "¿Es de Izquierda? ¡La Cultura!", autor4, 2024, "Políti
        biblioteca.agregarLibro(new Libro("935-5", "Generación de Cristal", autor5, 2022, "Ensayo", true))
        biblioteca.agregarLibro(new Libro("935-6", "Derrota Mundial", autor6, 1953, "Historia", false));
        biblioteca.agregarLibro(new Libro("935-7", "Blade Runner", autor3, 1968, "Ciencia Ficción", true))
        biblioteca.agregarLibro(new Libro("935-8", "Ubik", autor3, 1969, "Ciencia Ficción", true));
        biblioteca.agregarLibro(new Libro("935-9", "El Señor de los Anillos: La Comunidad", autor2, 1954,

        // 4. Listar todos los libros con su información y la del autor
        System.out.println("\n--- 4. LISTAR LIBROS ---");
        biblioteca.listarLibros();
```

```
// 5. Buscar un libro por su ISBN y mostrar su información
System.out.println("\n--- 5. BUSCAR LIBRO POR ISBN (935-7) ---");
Libro buscado = biblioteca.buscarPorISBN("935-7");
if (buscado != null) buscado.mostrarInfo();
else System.out.println("No se encontró el libro.");

// 6. Filtrar y mostrar los libros publicados en un año específico
System.out.println("\n--- 6. FILTRAR LIBROS PUBLICADOS EN 1968 ---");
List<Libro> publicados1968 = biblioteca.filtrarPorAnio(1968);
if (publicados1968.isEmpty())
    System.out.println("No hay libros publicados en ese año.");
else
    publicados1968.forEach(Libro::mostrarInfo);

// 7. Eliminar un libro por su ISBN y listar los libros restantes
System.out.println("\n--- 7. ELIMINAR LIBRO (935-4) ---");
boolean eliminado = biblioteca.eliminarLibro("935-4");
System.out.println(eliminado ? "Libro eliminado." : "No se encontró el libro.");
biblioteca.listarLibros();

// 8. Mostrar la cantidad total de libros en la biblioteca
System.out.println("\n--- 8. CANTIDAD TOTAL DE LIBROS ---");
System.out.println("Total de libros en biblioteca: " + biblioteca.contarLibros());

// 9. Listar todos los autores de los libros disponibles
System.out.println("\n--- 9. AUTORES DE LIBROS DISPONIBLES ---");
biblioteca.listarAutoresDisponibles();

 System.out.println("\n--- FIN DEL PROGRAMA ---");
```

Consola:

```
TP6 (run) ×    NetBeansProjects - D:\Mauro\Documents\NetBeansProjects ×
run:
--- INICIO DEL SISTEMA DE BIBLIOTECA ---

--- 3. AGREGANDO LIBROS ---

--- 4. LISTAR LIBROS ---
Listado de libros:
ISBN: 935-1 | Título: La hora de los pueblos | Autor: J. D. Perón (Argentino) | Año: 1968 | Género: Ensayo | Disponible: Sí
ISBN: 935-2 | Título: El Hobbit | Autor: J.R.R. Tolkien (Británico) | Año: 1937 | Género: Fantasía | Disponible: Sí
ISBN: 935-3 | Título: 1984 | Autor: George Orwell (Británico) | Año: 1949 | Género: Distopía | Disponible: Sí
ISBN: 935-4 | Título: ¿Es de Izquierda? ¡La Cultura! | Autor: Nicolás Márquez (Argentino) | Año: 2024 | Género: Política | D
ISBN: 935-5 | Título: Generación de Cristal | Autor: Agustín Laje (Argentino) | Año: 2022 | Género: Ensayo | Disponible: Sí
ISBN: 935-6 | Título: Derrota Mundial | Autor: Salvador Borrego (Mexicano) | Año: 1953 | Género: Historia | Disponible: No
ISBN: 935-7 | Título: Blade Runner | Autor: Philip K. Dick (Estadounidense) | Año: 1968 | Género: Ciencia Ficción | Disponib
ISBN: 935-8 | Título: Ubik | Autor: Philip K. Dick (Estadounidense) | Año: 1969 | Género: Ciencia Ficción | Disponible: Sí
ISBN: 935-9 | Título: El Señor de los Anillos: La Comunidad | Autor: J.R.R. Tolkien (Británico) | Año: 1954 | Género: Fantas

--- 5. BUSCAR LIBRO POR ISBN (935-7) ---
ISBN: 935-7 | Título: Blade Runner | Autor: Philip K. Dick (Estadounidense) | Año: 1968 | Género: Ciencia Ficción | Disponib

--- 6. FILTRAR LIBROS PUBLICADOS EN 1968 ---
ISBN: 935-1 | Título: La hora de los pueblos | Autor: J. D. Perón (Argentino) | Año: 1968 | Género: Ensayo | Disponible: Sí
ISBN: 935-7 | Título: Blade Runner | Autor: Philip K. Dick (Estadounidense) | Año: 1968 | Género: Ciencia Ficción | Disponib

--- 7. ELIMINAR LIBRO (935-4) ---
Libro eliminado.
Listado de libros:
ISBN: 935-1 | Título: La hora de los pueblos | Autor: J. D. Perón (Argentino) | Año: 1968 | Género: Ensayo | Disponible: Sí
ISBN: 935-2 | Título: El Hobbit | Autor: J.R.R. Tolkien (Británico) | Año: 1937 | Género: Fantasía | Disponible: Sí
ISBN: 935-3 | Título: 1984 | Autor: George Orwell (Británico) | Año: 1949 | Género: Distopía | Disponible: Sí
ISBN: 935-5 | Título: Generación de Cristal | Autor: Agustín Laje (Argentino) | Año: 2022 | Género: Ensayo | Disponible: Sí
ISBN: 935-6 | Título: Derrota Mundial | Autor: Salvador Borrego (Mexicano) | Año: 1953 | Género: Historia | Disponible: No
ISBN: 935-7 | Título: Blade Runner | Autor: Philip K. Dick (Estadounidense) | Año: 1968 | Género: Ciencia Ficción | Disponib
ISBN: 935-8 | Título: Ubik | Autor: Philip K. Dick (Estadounidense) | Año: 1969 | Género: Ciencia Ficción | Disponible: Sí
ISBN: 935-9 | Título: El Señor de los Anillos: La Comunidad | Autor: J.R.R. Tolkien (Británico) | Año: 1954 | Género: Fantas

--- 8. CANTIDAD TOTAL DE LIBROS ---
Total de libros en biblioteca: 8

--- 9. AUTORES DE LIBROS DISPONIBLES ---
Autores con libros disponibles:
- J. D. Perón (Argentino)
- J.R.R. Tolkien (Británico)
- George Orwell (Británico)
- Agustín Laje (Argentino)
- Philip K. Dick (Estadounidense)

--- FIN DEL PROGRAMA ---
BUILD SUCCESSFUL (total time: 0 seconds)
```

Ejercicio: Universidad, Profesor y Curso (bidireccional 1 a N)

Se debe modelar un sistema académico donde un Profesor dicta muchos Cursos y cada Curso tiene exactamente un Profesor responsable.

Clase Profesor:

```
public class Profesor {

    private String codigo;
    private String nombre;
    private String especialidad;
    private List<Curso> cursos;

    public Profesor(String codigo, String nombre, String especialidad) {
        this.codigo = codigo;
        this.nombre = nombre;
        this.especialidad = especialidad;
        this.cursos = new ArrayList<>();
    }

    public String getCodigo() { return codigo; }
    public String getNombre() { return nombre; }
    public String getEspecialidad() { return especialidad; }
    public List<Curso> getCursos() { return cursos; }

    // Agrega curso al profesor (si no está ya asignado)
    public void agregarCurso(Curso curso) {
        if (!cursos.contains(curso)) {
            cursos.add(curso);
            curso.setProfesor(this); // sincroniza el lado del curso
        }
    }

    // Remueve un curso del profesor
    public void removerCurso(Curso curso) {
        cursos.remove(curso);
    }

    public void mostrarInfo() {
        System.out.println("Profesor: " + nombre + " (" + codigo + ") - Especiali
        if (cursos.isEmpty()) {
            System.out.println("  Sin cursos asignados.");
        } else {
            System.out.println("  Cursos asignados:");
            for (Curso c : cursos) {
                System.out.println("    → " + c.getNombre());
            }
        }
    }

    @Override
    public String toString() {
        return nombre + " - " + especialidad;
    }
```

Clase Curso:

```java
public class Curso {

    private String codigo;
    private String nombre;
    private int duracionHoras;
    private Profesor profesor;

    public Curso(String codigo, String nombre, Profesor profesor, int duracionHoras) {
        this.codigo = codigo;
        this.nombre = nombre;
        this.profesor = profesor;
        this.duracionHoras = duracionHoras;

        // Si el curso tiene profesor al crearse, lo agregamos a su lista
        if (profesor != null) {
            profesor.agregarCurso(this);
        }
    }

    public String getCodigo() { return codigo; }
    public String getNombre() { return nombre; }
    public int getDuracionHoras() { return duracionHoras; }
    public Profesor getProfesor() { return profesor; }

    public void setProfesor(Profesor profesor) {
        this.profesor = profesor;
    }

    public void mostrarInfo() {
        System.out.printf("Curso: %-30s | Código: %s | Duración: %dh | Profesor: %s%n",
                nombre, codigo, duracionHoras,
                (profesor != null ? profesor.getNombre() : "Sin asignar"));
    }

    @Override
    public String toString() {
        return String.format("%s (%s) - %dh", nombre, codigo, duracionHoras);
    }

}
```

Clase Universidad:

```
public class Universidad {

    private String nombre;
    private List<Profesor> profesores;
    private List<Curso> cursos;

    public Universidad(String nombre) {
        this.nombre = nombre;
        this.profesores = new ArrayList<>();
        this.cursos = new ArrayList<>();
    }

    // Agregar profesor
    public void agregarProfesor(Profesor profesor) {
        if (!profesores.contains(profesor)) {
            profesores.add(profesor);
        }
    }

    // Agregar curso
    public void agregarCurso(Curso curso) {
        if (!cursos.contains(curso)) {
            cursos.add(curso);
        }
    }

    // 3. Asignar profesor a curso
    public void asignarProfesorACurso(String codigoCurso, Profesor profesor) {
        for (Curso c : cursos) {
            if (c.getCodigo().equalsIgnoreCase(codigoCurso)) {
                Profesor anterior = c.getProfesor();
                if (anterior != null) {
                    anterior.removerCurso(c); // quita del anterior
                }
                c.setProfesor(profesor);
                profesor.agregarCurso(c);
                System.out.println("Profesor " + profesor.getNombre() + " asignado
                return;
            }
        }
        System.out.println("Alerta!! Curso no encontrado: " + codigoCurso);
    }
```

```java
// Remover curso y sincronizar
public void removerCurso(String codigoCurso) {
    Curso cursoAEliminar = null;
    for (Curso c : cursos) {
        if (c.getCodigo().equalsIgnoreCase(codigoCurso)) {
            cursoAEliminar = c;
            break;
        }
    }
    if (cursoAEliminar != null) {
        Profesor prof = cursoAEliminar.getProfesor();
        if (prof != null) {
            prof.removerCurso(cursoAEliminar);
        }
        cursos.remove(cursoAEliminar);
        System.out.println("x Curso eliminado: " + codigoCurso);
    } else {
        System.out.println("x No se encontró el curso: " + codigoCurso);
    }
}

// Remover profesor (dejar sus cursos sin profesor)
public void removerProfesor(String codigoProfesor) {
    Profesor profesorAEliminar = null;
    for (Profesor p : profesores) {
        if (p.getCodigo().equalsIgnoreCase(codigoProfesor)) {
            profesorAEliminar = p;
            break;
        }
    }
    if (profesorAEliminar != null) {
        for (Curso c : profesorAEliminar.getCursos()) {
            c.setProfesor(null);
        }
        profesores.remove(profesorAEliminar);
        System.out.println(" Profesor eliminado: " + codigoProfesor);
    } else {
        System.out.println("Alerta!! Profesor no encontrado: " + codigoProfesor);
    }
}

// Listar cursos con su profesor
public void listarCursos() {

    for (Curso c : cursos) {
        c.mostrarInfo();
    }
}
```

Clase SistemaUniversitarioMain:

```java
public class SistemaUniversidadMain {

    public static void main(String[] args) {
        System.out.println("--- INICIO DEL SISTEMA UNIVERSITARIO ---");

        // Crear la universidad
        Universidad universidad = new Universidad("Universidad Tecnológica Nacional");

        // 2. Crear 3 profesores (codigo, Nombre, Especialidad)
        System.out.println("\n--- 1. CREANDO PROFESORES Y 5 CURSOS... ---");
        Profesor prof1 = new Profesor("616","Dr. Nicolas Tesla", "Física Teórica");
        Profesor prof2 = new Profesor("626", "Ing. Helena Poenix", "Análisis Matemático");
        Profesor prof3 = new Profesor("636", "Lic. Luz Andonian", "Programación");

        universidad.agregarProfesor(prof1);
        universidad.agregarProfesor(prof2);
        universidad.agregarProfesor(prof3);

        // 2. Crear 5 cursos y agregarlos a la universidad
        System.out.println("\n--- 2. AGREGANDO PROFESORES  Y CURSOS A LA UNIVERSIDAD... ---");
        Curso c1 = new Curso("FIS101", "Física I", null, 40);
        Curso c2 = new Curso("MAT102", "Análisis Matemático I", null, 50);
        Curso c3 = new Curso("QUI101", "Química General", null, 35);
        Curso c4 = new Curso("INF201", "Algoritmos y Estructura de Datos", null, 60);
        Curso c5 = new Curso("MAT203", "Análisis Matemático II", null, 45);

        universidad.agregarCurso(c1);
        universidad.agregarCurso(c2);
        universidad.agregarCurso(c3);
        universidad.agregarCurso(c4);
        universidad.agregarCurso(c5);

        // 3. Asignar profesores a cursos
        System.out.println("\n--- 3. ASIGNAR PROFESORES A CURSOS ---");
        universidad.asignarProfesorACurso("FIS101", prof1);
        universidad.asignarProfesorACurso("QUI101", prof1);
        universidad.asignarProfesorACurso("MAT102", prof2);
        universidad.asignarProfesorACurso("MAT203", prof2);
        universidad.asignarProfesorACurso("INF201", prof3);
```

```
        // 4. Listar cursos y profesores
        System.out.println("\n--- 4. LISTADO DE CURSOS ---");
        universidad.listarCursos();
        System.out.println("\n--- LISTADO DE PROFESORES ---");
        universidad.listarProfesores();

        // 5. Cambiar profesor de un curso y verificar sincronización
        System.out.println("\n--- 5. CAMBIANDO PROFESOR DEL CURSO FIS101 ---");
        universidad.asignarProfesorACurso("FIS101", prof3);

        universidad.listarCursos();
        universidad.listarProfesores();

        // 6. Remover un curso y confirmar que se elimina del profesor
        System.out.println("\n--- 6. ELIMINANDO CURSO QUI101 ---");
        universidad.removerCurso("QUI101");
        universidad.listarProfesores();

        // 7. Remover un profesor y dejar cursos con profesor = null
        System.out.println("\n--- 7. ELIMINANDO PROFESOR 626 (Helena Poenix) ---");
        universidad.removerProfesor("626");
        universidad.listarCursos();

        // 8. Reporte de cantidad de cursos por profesor
        System.out.println("\n--- 8. REPORTE: CANTIDAD DE CURSOS POR PROFESOR ---");
        universidad.reporteCursosPorProfesor();

        System.out.println("\n--- FIN DEL PROGRAMA ---");
    }
```

Consola:

```
TP6 (run) ×    NetBeansProjects - D:\Mauro\Documents\NetBeansProjects ×
run:
--- INICIO DEL SISTEMA UNIVERSITARIO ---

--- 1. CREANDO PROFESORES Y 5 CURSOS... ---

--- 2. AGREGANDO PROFESORES  Y CURSOS A LA UNIVERSIDAD... ---

--- 3. ASIGNAR PROFESORES A CURSOS ---
Profesor Dr. Nicolas Tesla asignado a Física I
Profesor Dr. Nicolas Tesla asignado a Química General
Profesor Ing. Helena Poenix asignado a Análisis Matemático I
Profesor Ing. Helena Poenix asignado a Análisis Matemático II
Profesor Lic. Luz Andonian asignado a Algoritmos y Estructura de Datos

--- 4. LISTADO DE CURSOS ---
Curso: Física I                      | Código: FIS101 | Duración: 40h | Profesor: Dr. Nicolas Tesla
Curso: Análisis Matemático I         | Código: MAT102 | Duración: 50h | Profesor: Ing. Helena Poenix
Curso: Química General               | Código: QUI101 | Duración: 35h | Profesor: Dr. Nicolas Tesla
Curso: Algoritmos y Estructura de Datos | Código: INF201 | Duración: 60h | Profesor: Lic. Luz Andonian
Curso: Análisis Matemático II        | Código: MAT203 | Duración: 45h | Profesor: Ing. Helena Poenix

--- LISTADO DE PROFESORES ---
Profesor: Dr. Nicolas Tesla (616) - Especialidad: Física Teórica
  Cursos asignados:
    → Física I
    → Química General
Profesor: Ing. Helena Poenix (626) - Especialidad: Análisis Matemático
  Cursos asignados:
    → Análisis Matemático I
    → Análisis Matemático II
Profesor: Lic. Luz Andonian (636) - Especialidad: Programación
  Cursos asignados:
    → Algoritmos y Estructura de Datos
```

```
--- 5. CAMBIANDO PROFESOR DEL CURSO FIS101 ---
Profesor Lic. Luz Andonian asignado a Física I
Curso: Física I                      | Código: FIS101 | Duración: 40h | Profesor: Lic. Luz Andonian
Curso: Análisis Matemático I         | Código: MAT102 | Duración: 50h | Profesor: Ing. Helena Poenix
Curso: Química General               | Código: QUI101 | Duración: 35h | Profesor: Dr. Nicolas Tesla
Curso: Algoritmos y Estructura de Datos | Código: INF201 | Duración: 60h | Profesor: Lic. Luz Andonian
Curso: Análisis Matemático II        | Código: MAT203 | Duración: 45h | Profesor: Ing. Helena Poenix
Profesor: Dr. Nicolas Tesla (616) - Especialidad: Física Teórica
  Cursos asignados:
    → Química General
Profesor: Ing. Helena Poenix (626) - Especialidad: Análisis Matemático
  Cursos asignados:
    → Análisis Matemático I
    → Análisis Matemático II
Profesor: Lic. Luz Andonian (636) - Especialidad: Programación
  Cursos asignados:
    → Algoritmos y Estructura de Datos
    → Física I

--- 6. ELIMINANDO CURSO QUI101 ---
x Curso eliminado: QUI101
Profesor: Dr. Nicolas Tesla (616) - Especialidad: Física Teórica
  Sin cursos asignados.
Profesor: Ing. Helena Poenix (626) - Especialidad: Análisis Matemático
  Cursos asignados:
    → Análisis Matemático I
    → Análisis Matemático II
Profesor: Lic. Luz Andonian (636) - Especialidad: Programación
  Cursos asignados:
    → Algoritmos y Estructura de Datos
    → Física I

--- 7. ELIMINANDO PROFESOR 626 (Helena Poenix) ---
 Profesor eliminado: 626
Curso: Física I                      | Código: FIS101 | Duración: 40h | Profesor: Lic. Luz Andonian
Curso: Análisis Matemático I         | Código: MAT102 | Duración: 50h | Profesor: Sin asignar
Curso: Algoritmos y Estructura de Datos | Código: INF201 | Duración: 60h | Profesor: Lic. Luz Andonian
Curso: Análisis Matemático II        | Código: MAT203 | Duración: 45h | Profesor: Sin asignar

--- 8. REPORTE: CANTIDAD DE CURSOS POR PROFESOR ---
Dr. Nicolas Tesla: 0 curso(s)
Lic. Luz Andonian: 2 curso(s)

--- FIN DEL PROGRAMA ---
BUILD SUCCESSFUL (total time: 0 seconds)
```

LINK:
https://github.com/27mau/UTN-Programacion-2/tree/main/TP6