Yamila Fernández

PROGRAMACIÓN II

Trabajo Práctico 6

Colecciones y Sistema de Stock

OBJETIVO GENERAL

Desarrollar estructuras de datos dinámicas en Java mediante el uso de colecciones (ArrayList) y enumeraciones (enum), implementando un sistema de stock con funcionalidades progresivas que refuerzan conceptos clave de la programación orientada a objetos.

MARCO TEÓRICO

Concepto	Aplicación en el proyecto
ArrayList	Estructura principal para almacenar productos en el inventario.
Enumeraciones (enum)	Representan las categorías de productos con valores predefinidos.
Relaciones 1 a N	Relación entre Inventario (1) y múltiples Productos (N).
Métodos en enum	Inclusión de descripciones dentro del enum para mejorar legibilidad.
Ciclo for-each	Recorre colecciones de productos para listado, búsqueda o filtrado.
Búsqueda y filtrado	Por ID y por categoría, aplicando condiciones.
Ordenamientos y reportes	Permiten organizar la información y mostrar estadísticas útiles.
Encapsulamiento	Restringir el acceso directo a los atributos de una clase

Caso Práctico 1

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.

2. Clases a implementar Clase Producto

Atributos:

● id (String) → Identificador único del producto.

● nombre (String) → Nombre del producto.

● precio (double) → Precio del producto.

● cantidad (int) → Cantidad en stock.

● categoria (CategoriaProducto) → Categoría del producto.

Métodos:

● mostrarInfo() → Muestra en consola la información del producto.

Enum CategoriaProducto

Valores:

● **ALIMENTOS**

● **ELECTRONICA**

● **ROPA**

● **HOGAR**

Método adicional:

```
public enum CategoriaProducto {
 ALIMENTOS("Productos comestibles"),
 ELECTRONICA("Dispositivos electrónicos"),
 ROPA("Prendas de vestir"),
HOGAR("Artículos para el hogar");
private final String descripcion;
CategoriaProducto(String descripcion) {
this.descripcion = descripcion;
 }
 public String getDescripcion() {
return descripcion;
 }
}
```

```java
package CasoPractico1;

public class Producto {
    private String id; //Identificador único del producto
    private String nombre; //Nombre del producto
    private double precio; // Precio del Producto
    private int cantidad; //Cantidad en stock
    private CategoriaProducto categoria; // Categoria del producto

    // Constructor
    public Producto(String id, String nombre, double precio, int catidad, CategoriaProducto categoria) {
        if (id == null || id.isEmpty()) throw new IllegalArgumentException("id no puede ser vacío");
        if (precio < 0) throw new IllegalArgumentException("precio no puede ser negativo");
        if (cantidad < 0) throw new IllegalArgumentException("cantidad no puede ser negativa");

        this.id = id;
        this.nombre = nombre;
        this.precio = precio;
        this.cantidad = catidad;
        this.categoria = categoria;
    }
```

```java
// Getters y Setters
public String getId() {
    return id;
}
public String getNombre() {
    return nombre;
}
public void setNombre(String nombre) {
    this.nombre = nombre;
}
public double getPrecio() {
    return precio;
}
public void setPrecio(double precio) {
    if (precio < 0) throw new IllegalArgumentException("precio no puede ser negativo");
    this.precio = precio;
}
public int getCantidad() {
    return cantidad;
}
public void setCantidad(int cantidad) {
    if (cantidad < 0) throw new IllegalArgumentException("cantidad no puede ser negativa");
    this.cantidad = cantidad;
}
public CategoriaProducto getCategoria() {
    return categoria;
}
public void setCategoria(CategoriaProducto categoria) {
    this.categoria = categoria;
}
```

```java
// Método para mostrar info en consola
public void mostrarInfo() {
    System.out.printf("ID: %s | Nombre: %s | Categoria: %s | Precio: %.2f | Cantidad: %d%n",
                      id, nombre, categoria, precio, cantidad);
}
```

```java
package CasoPractico1;

import java.util.ArrayList;

public class Inventario {

    private ArrayList<Producto> productos; //Esto representa la lista dinámica donde vas a guardar todos los productos del inventario.

    public Inventario() {
        this.productos = new ArrayList<>();
    }
```

Método para agregar productos

```java
// Agrega un nuevo producto al inventario, pero sin permitir duplicados con el mismo id.
public void agregarProducto(Producto p) {
    // Primero verificamos si el producto ya existe
    for (Producto prod : productos) {
        if (prod.getId().equalsIgnoreCase(p.getId())) {
            System.out.println("❌ Ya existe un producto con el ID: " + p.getId());
            return; // salimos sin agregar
        }
    }
    // Si no se encontró duplicado, lo agregamos
    productos.add(p);
    System.out.println("✅ Producto agregado correctamente: " + p.getNombre());
}
```

Método para listar productos

```java
public void listarProductos() {
if (productos.isEmpty()) {
    System.out.println("No hay productos en el inventario.");
    return;
}

System.out.println("\n LISTA DE PRODUCTOS EN INVENTARIO:");
for (Producto p : productos) {
    p.mostrarInfo(); // usa el método mostrarInfo() de tu clase Producto
}
```

Método para buscar productos por Id

```java
public Producto buscarProductoPorId(String id) {
// Recorremos la lista de productos
for (Producto p : productos) {
    // Comparamos ignorando mayúsculas/minúsculas
    if (p.getId().equalsIgnoreCase(id)) {
        return p; // Si encontramos el producto, lo devolvemos
    }
}
// Si no se encontró ninguno, devolvemos null
return null;
```

Método Filtrar productos por categoría

```java
public void filtrarPorCategoria(CategoriaProducto categoria) {
    System.out.println("\n--- Productos en la categoría: " + categoria + " ---");
    boolean encontrado = false;

    for (Producto p : productos) {
        if (p.getCategoria() == categoria) {
            p.mostrarInfo(); // muestra la info del producto
            encontrado = true;
        }
    }

    if (!encontrado) {
        System.out.println("No hay productos en esta categoría.");
    }
}
```

Método eliminar productos por Id

```
public void eliminarProducto(String id) {
boolean eliminado = false;
var iterator = productos.iterator();

while (iterator.hasNext()) {
    Producto p = iterator.next();
    if (p.getId().equalsIgnoreCase(id)) {
        iterator.remove(); // forma segura de eliminar
        System.out.println("🗑 Producto eliminado: " + p.getNombre());
        eliminado = true;
        break;
    }
}

if (!eliminado) {
    System.out.println("⚠ No se encontró ningún producto con el ID: " + id);
}
```

Método actualizar Stock de producto

```
public void actualizarStock(String id, int nuevaCantidad) {
boolean actualizado = false;

for (Producto p : productos) {
    if (p.getId().equalsIgnoreCase(id)) {
        p.setCantidad(nuevaCantidad);
        System.out.println("🔄 Stock actualizado para el producto: " + p.getNombre());
        System.out.println("   Nueva cantidad: " + nuevaCantidad);
        actualizado = true;
        break; // salimos porque ya lo actualizamos
    }
}

if (!actualizado) {
    System.out.println("⚠ No se encontró ningún producto con el ID: " + id);
}
```

Método mostrar Stock disponible

```
public int obtenerTotalStock() {
int total = 0;

for (Producto p : productos) {
    total += p.getCantidad(); // sumamos la cantidad de cada producto
}

return total;
```

Método para mostrar el producto con mayor stock

```java
public Producto obtenerProductoConMayorStock() {
if (productos.isEmpty()) {
    return null; // si la lista está vacía, no hay producto
}

Producto mayorStock = productos.get(0); // asumimos que el primero es el mayo

for (Producto p : productos) {
    if (p.getCantidad() > mayorStock.getCantidad()) {
        mayorStock = p; // si encontramos uno con más stock, lo actualizamos
    }
}
```

Método para filtrar productos entre 1000 y 3000

```java
public void filtrarProductosPorPrecio(double min, double max) {
 System.out.println("\n--- Productos con precio entre $" + min + " y $" + max + " ---");
 boolean encontrado = false;

 for (Producto p : productos) {
     if (p.getPrecio() >= min && p.getPrecio() <= max) {
         p.mostrarInfo();
         encontrado = true;
     }
 }

 if (!encontrado) {
     System.out.println("⚠ No se encontraron productos en ese rango de precios.");
 }
```

Método para mostrar las categorías de los productos

```java
    public void mostrarCategoriasDisponibles() {
    System.out.println("\n--- Categorías disponibles ---");
    for (CategoriaProducto c : CategoriaProducto.values()) {
        System.out.println(c.name() + " → " + c.getDescripcion());
    }
}
```

3. Tareas a realizar

1. Crear al menos cinco productos con diferentes categorías y agregarlos al inventario.

```java
package CasoPracticol;

public class Principal {

    public static void main(String[] args) {

        // 1. Creo el inventario
        Inventario inventario = new Inventario();

        // 2. Creo productos con diferentes categorias
        Producto p1 = new Producto("P001", "Guitarra Fender", 120000, 5, CategoriaProducto.ELECTRONICA);
        Producto p2 = new Producto("P002", "Notebook Lenovo", 450000, 3, CategoriaProducto.ELECTRONICA);
        Producto p3 = new Producto("P003", "Zapatillas Nike", 85000, 10, CategoriaProducto.ROPA);
        Producto p4 = new Producto("P004", "Silla ergonómica", 95000, 4, CategoriaProducto.HOGAR);
        Producto p5 = new Producto("P005", "Pan integral", 110000, 8, CategoriaProducto.ALIMENTOS);

        // 3. Agrego al inventario
        inventario.agregarProducto(p1);
        inventario.agregarProducto(p2);
        inventario.agregarProducto(p3);
        inventario.agregarProducto(p4);
        inventario.agregarProducto(p5);
```

```
run:
? Producto agregado correctamente: Guitarra Fender
? Producto agregado correctamente: Notebook Lenovo
? Producto agregado correctamente: Zapatillas Nike
? Producto agregado correctamente: Silla ergon?mica
? Producto agregado correctamente: Pan integral
BUILD SUCCESSFUL (total time: 0 seconds)
```

2. Listar todos los productos mostrando su información y categoría.

```
        // 4. Listo productos para verificar que se agregaron
        inventario.listarProductos();
    }

}
```

```
ut
el10d - C:\Users\el10d ×   Practica 6 (run) ×

 LISTA DE PRODUCTOS EN INVENTARIO:
ID: P001 | Nombre: Guitarra Fender | Categoria: ELECTRONICA (Dispositivos electr?nicos) | Precio: 120000,00 | Cantidad: 5
ID: P002 | Nombre: Notebook Lenovo | Categoria: ELECTRONICA (Dispositivos electr?nicos) | Precio: 450000,00 | Cantidad: 3
ID: P003 | Nombre: Zapatillas Nike | Categoria: ROPA (Prendas de vestir) | Precio: 85000,00 | Cantidad: 10
ID: P004 | Nombre: Silla ergon?mica | Categoria: HOGAR (Art?culos para el hogar) | Precio: 95000,00 | Cantidad: 4
ID: P005 | Nombre: Pan integral | Categoria: ALIMENTOS (Productos comestibles) | Precio: 110000,00 | Cantidad: 8
```

3. Buscar un producto por ID y mostrar su información.

```
System.out.println("\n--- BUSCAR PRODUCTO POR ID ---");

Producto buscado = inventario.buscarProductoPorId("P002");

if (buscado != null) {
    System.out.println("Producto encontrado:");
    buscado.mostrarInfo(); // usa el método mostrarInfo() de la clase Producto
} else {
    System.out.println("No se encontró ningún producto con ese ID.");
}
```

```
--- BUSCAR PRODUCTO POR ID ---
Producto encontrado:
ID: P002 | Nombre: Notebook Lenovo | Categoria: ELECTRONICA (Dispositivos electr?nicos) | Precio: 450000,00 | Cantidad: 3
```

4. Filtrar y mostrar productos que pertenezcan a una categoría específica.

```
        System.out.println ("\n--- FILTRAR POR CATEGORÍA ---");
        inventario.filtrarPorCategoria (CategoriaProducto.ELECTRONICA);
    }
}
```

```
utput
el10d - C:\Users\el10d ×   Practica 6 (run) ×
ID: P005 | Nombre: Pan integral | Categoria: ALIMENTOS (Productos comestibles) | Precio: 110000,00 | Cantidad: 8

--- FILTRAR POR CATEGOR?A ---

--- Productos en la categor?a: ELECTRONICA (Dispositivos electr?nicos) ---
ID: P001 | Nombre: Guitarra Fender | Categoria: ELECTRONICA (Dispositivos electr?nicos) | Precio: 120000,00 | Cantidad: 5
ID: P002 | Nombre: Notebook Lenovo | Categoria: ELECTRONICA (Dispositivos electr?nicos) | Precio: 450000,00 | Cantidad: 3
BUILD SUCCESSFUL (total time: 0 seconds)
```

5. Eliminar un producto por su ID y listar los productos restantes.

```
System.out.println("\n--- ELIMINAR PRODUCTO POR ID ---");
inventario.eliminarProducto("P003");
```

```
utput
el10d - C:\Users\el10d ×   Practica 6 (run) ×
--- ELIMINAR PRODUCTO POR ID ---
 Producto eliminado: Zapatillas Nike
```

6. Actualizar el stock de un producto existente.

```
System.out.println("\n--- ACTUALIZAR STOCK DE PRODUCTO ---");
inventario.actualizarStock("P002", 12); // actualiza la cantidad de Notebook Lenovo

// Verificamos el cambio
System.out.println("\n--- LISTA ACTUALIZADA ---");
inventario.listarProductos();
```

```
Output
el10d - C:\Users\el10d ×   Practica 6 (run) ×
--- LISTA ACTUALIZADA ---

 LISTA DE PRODUCTOS EN INVENTARIO:
ID: P001 | Nombre: Guitarra Fender | Categoria: ELECTRONICA (Dispositivos electr�nicos) | Precio: 120000,00 | Cantidad: 5
ID: P002 | Nombre: Notebook Lenovo | Categoria: ELECTRONICA (Dispositivos electr�nicos) | Precio: 450000,00 | Cantidad: 12
ID: P004 | Nombre: Silla ergon�mica | Categoria: HOGAR (Art�culos para el hogar) | Precio: 95000,00 | Cantidad: 4
ID: P005 | Nombre: Pan integral | Categoria: ALIMENTOS (Productos comestibles) | Precio: 110000,00 | Cantidad: 8
BUILD SUCCESSFUL (total time: 0 seconds)
```

7. Mostrar el total de stock disponible.

```
System.out.println("\n--- TOTAL DE STOCK DISPONIBLE ---");
int totalStock = inventario.obtenerTotalStock();
System.out.println("📦 Total de unidades disponibles en inventario: " + totalStock);
```

```
0d - C:\Users\el10d ×   Practica 6 (run) ×
-- TOTAL DE STOCK DISPONIBLE ---
 Total de unidades disponibles en inventario: 29
UILD SUCCESSFUL (total time: 0 seconds)
```

8. Obtener y mostrar el producto con mayor stock.

```
if (productoMayorStock != null) {
    System.out.println("El producto con mayor stock es:");
    productoMayorStock.mostrarInfo();
} else {
    System.out.println("No hay productos en el inventario.");
}
```

```
10d - C:\Users\el10d ×   Practica 6 (run) ×
--- PRODUCTO CON MAYOR STOCK ---
El producto con mayor stock es:
ID: P002 | Nombre: Notebook Lenovo | Categoria: ELECTRONICA (Dispositivos electr�nicos) | Precio: 450000,00 | Cantidad: 12
```

9. Filtrar productos con precios entre $1000 y $3000.

```
        System.out.println("\n--- FILTRAR PRODUCTOS POR PRECIO ---");
        inventario.filtrarProductosPorPrecio(1000, 3000);
```

```
ut
el10d - C:\Users\el10d ×   Practica 6 (run) ×
--- Productos con precio entre $1000.0 y $3000.0 ---
No se encontraron productos en ese rango de precios.
```

10. Mostrar las categorías disponibles con sus descripciones.

```
        System.out.println("\n--- MOSTRAR CATEGORÍAS DISPONIBLES ---");
        inventario.mostrarCategoriasDisponibles();

    }
}
```

```
utput
el10d - C:\Users\el10d ×   Practica 6 (run) ×
--- Categor?as disponibles ---
ALIMENTOS Productos comestibles
ELECTRONICA Dispositivos electr?nicos
ROPA Prendas de vestir
HOGAR Art?culos para el hogar
```