PROGRAMACIÓN II

Trabajo Práctico 6: Colecciones y Sistema de Stock

Alumno: Angelelli, Rodrigo Martin
Comisión No 7
Link GitHub: https://github.com/roangelelli/Programacion2_TUPAD_Angelelli_Rodrigo

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.

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**:
    a. **Clase Producto**
    b. **Enum CategoriaProducto**
    c. **Clase Inventario**
3. **Tareas a realizar:**
    a. Crear al menos cinco productos con diferentes categorías y agregarlos al inventario.
    b. Listar todos los productos mostrando su información y categoría.
    c. Buscar un producto por ID y mostrar su información.
    d. Filtrar y mostrar productos que pertenezcan a una categoría específica.
    e. Eliminar un producto por su ID y listar los productos restantes.
    f. Actualizar el stock de un producto existente.
    g. Mostrar el total de stock disponible.
    h. Obtener y mostrar el producto con mayor stock.
    i. Filtrar productos con precios entre $1000 y $3000.
    j. Mostrar las categorías disponibles con sus descripciones.

Clase Producto:

```java
package ejercicio1.sistemastock;
public class Producto {

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

    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;
    }

    public String getId() {...3 lines }

    public String getNombre() {...3 lines }

    public double getPrecio() {...3 lines }

    public int getCantidad() {...3 lines }

    public CategoriaProducto getCategoria() {...3 lines }

    public void setCantidad(int cantidad) {...3 lines }

    public void mostrarInfo() {
        System.out.println("ID: " + id +
                " | Nombre: " + nombre +
                " | Precio: $" + precio +
                " | Cantidad: " + cantidad +
                " | Categoría: " + categoria +
                " (" + categoria.getDescripcion() + ")");
    }

    @Override
    public String toString() {
        return "Producto{" +
                "id='" + id + '\'' +
                ", nombre='" + nombre + '\'' +
                ", precio=" + precio +
                ", cantidad=" + cantidad +
                ", categoria=" + categoria +
                '}';
    }
}
```

Enum CategoriaProducto:

```java
package ejercicio1.sistemastock;
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;
    }
}
```

Clase Inventario:

```java
package ejercicio1.sistemastock;
import java.util.ArrayList;
public class Inventario {

    private ArrayList<Producto> productos;

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

    // 1) Agregar producto
    public void agregarProducto(Producto p) {
        productos.add(p);
    }
    // 2) Listar productos
    public void listarProductos() {
        if (productos.isEmpty()) {
            System.out.println("No hay productos en el inventario.");
        } else {
            for (Producto p : productos) {
                p.mostrarInfo();
            }}}
    // 3) Buscar producto por ID
    public Producto buscarProductoPorId(String id) {
        for (Producto p : productos) {
            if (p.getId().equals(id)) {
                return p;
            }
        }
        return null; // si no lo encuentra
    }
    // 4) Eliminar producto por ID
    public boolean eliminarProducto(String id) {
        Producto encontrado = buscarProductoPorId(id);
        if (encontrado != null) {
            productos.remove(encontrado);
            return true;
        }
        return false;
    }
```

```java
    // 5) Actualizar stock por ID
    public boolean actualizarStock(String id, int nuevaCantidad) {
        Producto p = buscarProductoPorId(id);
        if (p != null) {
            p.setCantidad(nuevaCantidad);
            return true;
        }
        return false;
    }
    // 6) Filtrar por categoría
    public ArrayList<Producto> filtrarPorCategoria(CategoriaProducto categoria) {
        ArrayList<Producto> filtrados = new ArrayList<>();
        for (Producto p : productos) {
            if (p.getCategoria() == categoria) {
                filtrados.add(p);
            }
        }
        return filtrados;
    }
    // 7) Obtener total de stock (suma de cantidades)
    public int obtenerTotalStock() {
        int total = 0;
        for (Producto p : productos) {
            total += p.getCantidad();
        }
        return total;
    }
    // 8) Obtener producto con mayor stock
    public Producto obtenerProductoConMayorStock() {
        if (productos.isEmpty()) {
            return null;
        }
        Producto max = productos.get(0);
        for (Producto p : productos) {
            if (p.getCantidad() > max.getCantidad()) {
                max = p;
            }
        }
        return max;
    }
    // 9) Filtrar productos por rango de precio
    public ArrayList<Producto> filtrarProductosPorPrecio(double min, double max) {
        ArrayList<Producto> filtrados = new ArrayList<>();
        for (Producto p : productos) {
            if (p.getPrecio() >= min && p.getPrecio() <= max) {
                filtrados.add(p);
            }
        }
        return filtrados;
    }
```

```java
    // 10) Mostrar categorías disponibles
    public void mostrarCategoriasDisponibles() {
        for (CategoriaProducto c : CategoriaProducto.values()) {
            System.out.println(c + " - " + c.getDescripcion());
        }
    }
}
```

Main:

```java
package ejercicio1.sistemastock;
public class Ejercicio1SistemaStock {
    public static void main(String[] args) {
        Inventario inventario = new Inventario();

        // 1) Crear al menos cinco productos con diferentes categorías y agregarlos al inventario.
        Producto p1 = new Producto("P001", "Leche", 1200, 50, CategoriaProducto.ALIMENTOS);
        Producto p2 = new Producto("P002", "Celular", 250000, 10, CategoriaProducto.ELECTRONICA);
        Producto p3 = new Producto("P003", "Remera", 8000, 30, CategoriaProducto.ROPA);
        Producto p4 = new Producto("P004", "Silla", 15000, 20, CategoriaProducto.HOGAR);
        Producto p5 = new Producto("P005", "Galletitas", 900, 100, CategoriaProducto.ALIMENTOS);

        inventario.agregarProducto(p1);
        inventario.agregarProducto(p2);
        inventario.agregarProducto(p3);
        inventario.agregarProducto(p4);
        inventario.agregarProducto(p5);

        // 2) Listar todos los productos mostrando su información y categoría.
        System.out.println("==== LISTA DE PRODUCTOS ====");
        inventario.listarProductos();

        // 3) Buscar un producto por ID y mostrar su información.
        System.out.println("\n==== BUSCAR PRODUCTO POR ID (P003) ====");
        Producto buscado = inventario.buscarProductoPorId("P003");
        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==== PRODUCTOS DE CATEGORÍA ALIMENTOS ====");
        for (Producto p : inventario.filtrarPorCategoria(CategoriaProducto.ALIMENTOS)) {
            p.mostrarInfo();
        }

        // 5) Eliminar un producto por su ID y listar los productos restantes.
        System.out.println("\n==== ELIMINAR PRODUCTO P002 (Celular) ====");
        boolean eliminado = inventario.eliminarProducto("P002");
        System.out.println(eliminado ? "Producto eliminado correctamente." : "No se encontró el producto.");
        System.out.println("==== PRODUCTOS RESTANTES ====");
        inventario.listarProductos();

        // 6) Actualizar el stock de un producto existente.
        System.out.println("\n==== ACTUALIZAR STOCK DE P001 (Leche) A 80 ====");
        boolean actualizado = inventario.actualizarStock("P001", 80);
        System.out.println(actualizado ? "Stock actualizado." : "Producto no encontrado.");
        inventario.buscarProductoPorId("P001").mostrarInfo();

        // 7) Mostrar el total de stock disponible.
        System.out.println("\n==== TOTAL DE STOCK DISPONIBLE ====");
        System.out.println("Total de unidades en stock: " + inventario.obtenerTotalStock());

        // 8) Obtener y mostrar el producto con mayor stock.
        System.out.println("\n==== PRODUCTO CON MAYOR STOCK ====");
        Producto maxStock = inventario.obtenerProductoConMayorStock();
        if (maxStock != null) {
            maxStock.mostrarInfo();
        }

        // 9) Filtrar productos con precios entre $1000 y $3000.
        System.out.println("\n==== PRODUCTOS CON PRECIO ENTRE $1000 Y $3000 ====");
        for (Producto p : inventario.filtrarProductosPorPrecio(1000, 3000)) {
            p.mostrarInfo();
        }

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

1. **Tareas a realizar:**
   a. Crear al menos cinco productos con diferentes categorías y agregarlos al inventario.

```
// 1) Crear al menos cinco productos con diferentes categorías y agregarlos al inventario.
Producto p1 = new Producto("P001", "Leche", 1200, 50, CategoriaProducto.ALIMENTOS);
Producto p2 = new Producto("P002", "Celular", 250000, 10, CategoriaProducto.ELECTRONICA);
Producto p3 = new Producto("P003", "Remera", 8000, 30, CategoriaProducto.ROPA);
Producto p4 = new Producto("P004", "Silla", 15000, 20, CategoriaProducto.HOGAR);
Producto p5 = new Producto("P005", "Galletitas", 900, 100, CategoriaProducto.ALIMENTOS);

inventario.agregarProducto(p1);
inventario.agregarProducto(p2);
inventario.agregarProducto(p3);
inventario.agregarProducto(p4);
inventario.agregarProducto(p5);
```

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

```
// 2) Listar todos los productos mostrando su información y categoría.
System.out.println("==== LISTA DE PRODUCTOS ====");
inventario.listarProductos();
```

```
==== LISTA DE PRODUCTOS ====
ID: P001 | Nombre: Leche | Precio: $1200.0 | Cantidad: 50 | Categoría: ALIMENTOS (Productos comestibles)
ID: P002 | Nombre: Celular | Precio: $250000.0 | Cantidad: 10 | Categoría: ELECTRONICA (Dispositivos electrónicos)
ID: P003 | Nombre: Remera | Precio: $8000.0 | Cantidad: 30 | Categoría: ROPA (Prendas de vestir)
ID: P004 | Nombre: Silla | Precio: $15000.0 | Cantidad: 20 | Categoría: HOGAR (Artículos para el hogar)
ID: P005 | Nombre: Galletitas | Precio: $900.0 | Cantidad: 100 | Categoría: ALIMENTOS (Productos comestibles)
```

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

```
// 3) Buscar un producto por ID y mostrar su información.
System.out.println("\n==== BUSCAR PRODUCTO POR ID (P003) ====");
Producto buscado = inventario.buscarProductoPorId("P003");
if (buscado != null) {
    buscado.mostrarInfo();
} else {
    System.out.println("Producto no encontrado.");
}
```

```
==== BUSCAR PRODUCTO POR ID (P003) ====
ID: P003 | Nombre: Remera | Precio: $8000.0 | Cantidad: 30 | Categoría: ROPA (Prendas de vestir)
```

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

```
// 4) Filtrar y mostrar productos que pertenezcan a una categoría específica.
System.out.println("\n==== PRODUCTOS DE CATEGORÍA ALIMENTOS ====");
for (Producto p : inventario.filtrarPorCategoria(CategoriaProducto.ALIMENTOS)) {
    p.mostrarInfo();
}
```

```
==== PRODUCTOS DE CATEGORÍA ALIMENTOS ====
ID: P001 | Nombre: Leche | Precio: $1200.0 | Cantidad: 50 | Categoría: ALIMENTOS (Productos comestibles)
ID: P005 | Nombre: Galletitas | Precio: $900.0 | Cantidad: 100 | Categoría: ALIMENTOS (Productos comestibles)
```

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

```
// 5) Eliminar un producto por su ID y listar los productos restantes.
System.out.println("\n==== ELIMINAR PRODUCTO P002 (Celular) ====");
boolean eliminado = inventario.eliminarProducto("P002");
System.out.println(eliminado ? "Producto eliminado correctamente." : "No se encontró el producto.");
System.out.println("==== PRODUCTOS RESTANTES ====");
inventario.listarProductos();
```

```
==== ELIMINAR PRODUCTO P002 (Celular) ====
Producto eliminado correctamente.
==== PRODUCTOS RESTANTES ====
ID: P001 | Nombre: Leche | Precio: $1200.0 | Cantidad: 50 | Categoría: ALIMENTOS (Productos comestibles)
ID: P003 | Nombre: Remera | Precio: $8000.0 | Cantidad: 30 | Categoría: ROPA (Prendas de vestir)
ID: P004 | Nombre: Silla | Precio: $15000.0 | Cantidad: 20 | Categoría: HOGAR (Artículos para el hogar)
ID: P005 | Nombre: Galletitas | Precio: $900.0 | Cantidad: 100 | Categoría: ALIMENTOS (Productos comestibles)
```

f. Actualizar el stock de un producto existente.

```
// 6) Actualizar el stock de un producto existente.
System.out.println("\n==== ACTUALIZAR STOCK DE P001 (Leche) A 80 ====");
boolean actualizado = inventario.actualizarStock("P001", 80);
System.out.println(actualizado ? "Stock actualizado." : "Producto no encontrado.");
inventario.buscarProductoPorId("P001").mostrarInfo();
```

```
==== ACTUALIZAR STOCK DE P001 (Leche) A 80 ====
Stock actualizado.
ID: P001 | Nombre: Leche | Precio: $1200.0 | Cantidad: 80 | Categoría: ALIMENTOS (Productos comestibles)
```

g. Mostrar el total de stock disponible.

```
// 7) Mostrar el total de stock disponible.
System.out.println("\n==== TOTAL DE STOCK DISPONIBLE ====");
System.out.println("Total de unidades en stock: " + inventario.obtenerTotalStock());
```

```
==== TOTAL DE STOCK DISPONIBLE ====
Total de unidades en stock: 230
```

h. Obtener y mostrar el producto con mayor stock.

```
// 8) Obtener y mostrar el producto con mayor stock.
System.out.println("\n==== PRODUCTO CON MAYOR STOCK ====");
Producto maxStock = inventario.obtenerProductoConMayorStock();
if (maxStock != null) {
    maxStock.mostrarInfo();
}
```

```
==== PRODUCTO CON MAYOR STOCK ====
ID: P005 | Nombre: Galletitas | Precio: $900.0 | Cantidad: 100 | Categoría: ALIMENTOS (Productos comestibles)
```

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

```java
// 9) Filtrar productos con precios entre $1000 y $3000.
System.out.println("\n==== PRODUCTOS CON PRECIO ENTRE $1000 Y $3000 ====");
for (Producto p : inventario.filtrarProductosPorPrecio(1000, 3000)) {
    p.mostrarInfo();
}
```

```
==== PRODUCTOS CON PRECIO ENTRE $1000 Y $3000 ====
ID: P001 | Nombre: Leche | Precio: $1200.0 | Cantidad: 80 | Categoría: ALIMENTOS (Productos comestibles)
```

j. Mostrar las categorías disponibles con sus descripciones.

```java
    // 10) Mostrar las categorías disponibles con sus descripciones.
    System.out.println("\n==== CATEGORÍAS DISPONIBLES ====");
    inventario.mostrarCategoriasDisponibles();
}
```

```
==== CATEGORÍAS DISPONIBLES ====
ALIMENTOS - Productos comestibles
ELECTRONICA - Dispositivos electrónicos
ROPA - Prendas de vestir
HOGAR - Artículos para el hogar
BUILD SUCCESSFUL (total time: 0 seconds)
```