

DEPARTAMENTO:	CIENCIAS DE LA COMPUTACION	CARRERA:	Tecnologías de la Información y comunicación		
ASIGNATURA:	PROGRAMACION ORIENTADA A OBJETOS	NIVEL:	2	FECHA:	14/05/2025
DOCENTE:	Ing. Jhon Cruz	PRÁCTICA N°:		CALIFICACIÓN:	

Archivos CSV y JSON

Corina Alejandra Acosta Oliva

¿Qué es CSV?

Un archivo CSV (*Comma Separated Values*) es básicamente un archivo de texto plano donde los datos están separados por comas. Es como una tabla de Excel pero en versión simple: cada línea es un registro, y cada valor está separado por comas. Lo bueno es que lo podemos abrir con cualquier editor o programa y es fácil de leer y escribir desde código. Ideal para guardar listas de cosas como productos, usuarios, notas, etc.

¿Qué es Json?

JSON es como una forma simple y ordenada de guardar y mandar datos. Usa llaves {}, corchetes [] y se basa en pares clave-valor, tipo: "nombre": "Juan". Es fácil de leer, funciona con casi todos los lenguajes y se usa mucho para enviar info entre programas, como de un servidor a una app.

EJERCICIO:

En este proyecto se modeló un sistema básico de supermercado orientado a objetos en Java. Se implementaron las clases Producto y Categoría con encapsulamiento completo, constructores obligatorios y validaciones internas. Cada producto pertenece a una categoría, estableciendo una relación de composición. Se incluyó persistencia de datos mediante escritura y lectura de archivos CSV y JSON, junto con manejo de excepciones de E/S. Además, se optimizó la estructura del código aplicando principios de código limpio y modularización, incluyendo validadores personalizados y métodos de presentación como mostrarResumen().

Código:

Main(P1supermercado):

Esta es la clase principal, donde arranca todo el programa se encarga de mostrar el menú al usuario, leer inputs y manejar las acciones. Cambié el sistema viejo de escribir el nombre exacto de la categoría a un menú numerado (más rápido y sin errores). También añadí validaciones para evitar que se rompa todo si escriben mal.

```
package miespe.p1supermercado;

import java.util.ArrayList;
import java.util.HashMap;
import java.util.List;
import java.util.Map;
import java.util.Scanner;

public class P1Supermercado {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        Map<String, Categoría> categorias = new HashMap<>();
        int opcion;

        do {
            System.out.println("\n== MENU SUPERMERCADO ==");
            System.out.println("1. Mostrar Productos");
            System.out.println("2. Agregar Producto");
            System.out.println("3. Eliminar Producto");
            System.out.println("4. Actualizar Producto");
            System.out.println("5. Salir");
            System.out.print("Opción: ");
            opcion = scanner.nextInt();

            switch (opcion) {
                case 1:
                    mostrarProductos(categorias);
                    break;
                case 2:
                    agregarProducto(categorias);
                    break;
                case 3:
                    eliminarProducto(categorias);
                    break;
                case 4:
                    actualizarProducto(categorias);
                    break;
                case 5:
                    System.out.println("Saliendo...");
                    return;
                default:
                    System.out.println("Opción no válida. Intente nuevamente.");
            }
        } while (true);
    }

    private void mostrarProductos(Map<String, Categoría> categorias) {
        System.out.println("Productos registrados:");
        for (Map.Entry<String, Categoría> entry : categorias.entrySet()) {
            System.out.println(entry.getKey() + ": " + entry.getValue().getNombre());
        }
    }

    private void agregarProducto(Map<String, Categoría> categorias) {
        String nombre;
        double precio;
        int stock;
        String categoriaNombre;
        Categoría categoria;

        System.out.print("Nombre del producto: ");
        nombre = scanner.nextLine();
        System.out.print("Precio: ");
        precio = scanner.nextDouble();
        System.out.print("Stock: ");
        stock = scanner.nextInt();
        System.out.print("Categoría: ");
        categoriaNombre = scanner.nextLine();

        if (categorias.containsKey(categoriaNombre)) {
            categoria = categorias.get(categoriaNombre);
            categoria.agregarProducto(nombre, precio, stock);
        } else {
            System.out.println("Categoría no encontrada. Agregando nueva categoría...");
            categoria = new Categoría(categoriaNombre);
            categorias.put(categoriaNombre, categoria);
            categoria.agregarProducto(nombre, precio, stock);
        }
    }

    private void eliminarProducto(Map<String, Categoría> categorias) {
        String nombre;
        String categoriaNombre;
        Categoría categoria;

        System.out.print("Nombre del producto: ");
        nombre = scanner.nextLine();
        System.out.print("Categoría: ");
        categoriaNombre = scanner.nextLine();

        if (categorias.containsKey(categoriaNombre)) {
            categoria = categorias.get(categoriaNombre);
            categoria.eliminarProducto(nombre);
        } else {
            System.out.println("Categoría no encontrada. No se pudo eliminar el producto.");
        }
    }

    private void actualizarProducto(Map<String, Categoría> categorias) {
        String nombre;
        double nuevoPrecio;
        int nuevoStock;
        String categoriaNombre;
        Categoría categoria;

        System.out.print("Nombre del producto: ");
        nombre = scanner.nextLine();
        System.out.print("Nuevo precio: ");
        nuevoPrecio = scanner.nextDouble();
        System.out.print("Nuevo stock: ");
        nuevoStock = scanner.nextInt();
        System.out.print("Categoría: ");
        categoriaNombre = scanner.nextLine();

        if (categorias.containsKey(categoriaNombre)) {
            categoria = categorias.get(categoriaNombre);
            categoria.actualizarProducto(nombre, nuevoPrecio, nuevoStock);
        } else {
            System.out.println("Categoría no encontrada. No se pudo actualizar el producto.");
        }
    }
}
```

```
System.out.println("1. Agregar categoria");
System.out.println("2. Agregar producto a una categoria");
System.out.println("3. Ver categorias y productos");
System.out.println("4. Salir");

opcion = Validador.leerEntero(scanner, "Elige una opcion: ");

switch (opcion) {
    case 1:
        String nombreCat = Validador.leerTextoNoVacio(scanner, "Ingresa el nombre de la
categoria: ")
            .toLowerCase();

        if (categorias.containsKey(nombreCat)) {
            System.out.println(" ! Esa categoria ya existe.");
        } else {
            categorias.put(nombreCat, new Categoria(nombreCat));
            System.out.println("✓ Categoria agregada correctamente.");
        }
        break;

    case 2:
        if (categorias.isEmpty()) {
            System.out.println(" Primero debes agregar una categoria.");
            break;
        }

        System.out.println("Categorias disponibles:");
        int i = 1;
        List<String> listaCategorias = new ArrayList<>(categorias.keySet());
        for (String cat : listaCategorias) {
            System.out.println(i + ". " + cat);
            i++;
        }

        int indice = Validador.leerEntero(scanner, "¿A que categoria quieres agregar el
producto? (numero): ");
        if (indice < 1 || indice > listaCategorias.size()) {
            System.out.println(" ! Numero de categoría invalido.");
            break;
        }

        String catElegida = listaCategorias.get(indice - 1);

        if (!categorias.containsKey(catElegida)) {
            System.out.println(" ! Esa categoria no existe.");
            break;
        }

        String nombreProd = Validador.leerTextoNoVacio(scanner, "Nombre del producto: ");
        float precioProd = Validador.leerFloatPositivo(scanner, "Precio del producto (usa punto
para decimales): ");

        Producto nuevoProd = new Producto(nombreProd, precioProd);
        categorias.get(catElegida).agregarProducto(nuevoProd);

        System.out.println("✓ Producto agregado correctamente.");

        // Guardar datos
        ArchivoUtil.guardarComoCSV(categorias);
        ArchivoUtil.guardarComoJSON(categorias);
        break;

    case 3:
        if (categorias.isEmpty()) {
            System.out.println(" No hay categorias registradas.");
        } else {
            for (Categoria cat : categorias.values()) {
                System.out.println(cat);
            }
        }
}
```

```
        }

    break;

case 4:
    System.out.println("Saliendo del sistema... 🌟");
    break;

default:
    System.out.println(" ! Opcion invalida.");
}

} while (opcion != 4);
}
```

Clase Categoría()

Clase que representa una categoría (tipo: Lácteos, Carnes, etc) Cada categoría tiene un nombre y una lista de productos. Relación: una categoría puede tener muchos productos (composición). También está bien encapsulada, y su método `toString()` me sirve para imprimir bonito en el menú.

```
/*
 * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license
 * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to edit this template
 */
package miespe.p1supermercado;

/**
 *
 * @author espe
 */
import java.util.ArrayList;
import java.util.List;

public class Categoria {
    private String nombre;
    private List<Producto> productos;

    public Categoria(String nombre) {
        this.nombre = nombre.toLowerCase().trim();
        this.productos = new ArrayList<>();
    }

    public String getNombre() {
        return nombre;
    }

    public List<Producto> getProductos() {
        return productos;
    }

    public void agregarProducto(Producto producto) {
        productos.add(producto);
    }

    @Override
    public String toString() {
        StringBuilder sb = new StringBuilder("Categoria: " + nombre + "\n");
        if (productos.isEmpty()) {
            sb.append(" (sin productos)");
        } else {
            for (Producto p : productos) {
                sb.append(" - ").append(p).append("\n");
            }
        }
        return sb.toString();
    }
}
```

Clase Producto()

Esta clase representa un producto del supermercado . Tiene ID autogenerado, nombre, precio y ahora una categoría como objeto (composición) Aplíquese encapsulamiento full: atributos privados + getters/setters con validación. El constructor obliga a inicializar todo desde el inicio (código limpio). Validación extra: el nombre no puede estar vacío y el precio tiene que ser mayor a 0. También tiene un método mostrarResumen() para mostrar los datos del producto + la categoría.

```
/*
 * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license
 * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to edit this template
 */
package misespe.p1supermercado;

/**
 *
 * @author espe
 */
public class Producto {
    private static int contadorId = 1; // ID autoincremental
    private int id;
    private String nombre;
    private float precio;

    public Producto(String nombre, float precio) {
        this.id = contadorId++;
        this.nombre = nombre;
        this.precio = precio;
    }

    public int getId() {
        return id;
    }

    public String getNombre() {
        return nombre;
    }

    public float getPrecio() {
        return precio;
    }

    public void setNombre(String nombre) {
        this.nombre = nombre;
    }

    public void setPrecio(float precio) {
        this.precio = precio;
    }

    @Override
    public String toString() {
        return "ID: " + id + " - " + nombre + " - $" + precio;
    }
}
```

Clase Validador()

Clase utilitaria que hice para validar entradas del usuario (evita que explote el programa si meten letras en vez de números, por ejemplo) Tiene métodos para leer enteros, floats positivos y texto que no esté vacío Lo uso en el main para que todos los inputs sean seguros y bien formateados.

```
/*
 * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license
 * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to edit this template
 */
package miespe.p1supermercado;

/**
 *
 * @author caaco
 */

import java.util.Scanner;

public class Validador {

    public static int leerEntero(Scanner scanner, String mensaje) {
        int numero;
        while (true) {
            System.out.print(mensaje);
            String input = scanner.nextLine().trim();
            try {
                numero = Integer.parseInt(input);
                break;
            } catch (NumberFormatException e) {
                System.out.println(" ! Por favor ingresa un numero entero valido.");
            }
        }
        return numero;
    }

    public static float leerFloatPositivo(Scanner scanner, String mensaje) {
        float numero;
        while (true) {
            System.out.print(mensaje);
            String input = scanner.nextLine().trim();
            try {
                numero = Float.parseFloat(input);
                if (numero <= 0) {
                    System.out.println(" ! El numero debe ser mayor a 0.");
                } else {
                    break;
                }
            } catch (NumberFormatException e) {
                System.out.println(" ! Por favor ingresa un numero decimal valido (usa punto).");
            }
        }
        return numero;
    }

    public static String leerTextoNoVacio(Scanner scanner, String mensaje) {
        String texto;
        while (true) {
            System.out.print(mensaje);
            texto = scanner.nextLine().trim();
            if (texto.isEmpty()) {
                System.out.println(" ! El texto no puede estar vacio.");
            } else {
                break;
            }
        }
    }
}
```

```
        return texto;
    }
}
```

Clase ArchivoUtil()

Esta clase se encarga de guardar los productos en archivos externos tiene dos métodos: uno para guardar en CSV y otro en JSON (ambos súper útiles si quiero usar los datos después o compartirlos). Usé FileWriter y Gson (una librería para manejar JSON en Java) Maneja errores con try/catch por si hay problemas al escribir archivos. Agregué esto para que no se pierdan los productos después de cerrar el programa

```
/*
 * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license
 * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to edit this template
 */
package miespe.p1supermercado;

import java.io.FileWriter;
import java.io.IOException;
import java.util.Map;
//import java.util.List;

import com.google.gson.Gson;
import com.google.gson.GsonBuilder;

public class ArchivoUtil {

    public static void guardarComoCSV(Map<String, Categoría> categorías) {
        try (FileWriter writer = new FileWriter("productos.csv")) {
            writer.write("Categoria, ID, Nombre, Precio\n");
            for (Categoría cat : categorías.values()) {
                for (Producto prod : cat.getProductos()) {
                    writer.write(cat.getNombre() + "," + prod.getId() + "," + prod.getNombre() + "," + prod.getPrecio() + "\n");
                }
            }
        } catch (IOException e) {
            System.out.println("Error escribiendo archivo CSV: " + e.getMessage());
        }
    }

    public static void guardarComoJSON(Map<String, Categoría> categorías) {
        try (FileWriter writer = new FileWriter("productos.json")) {
            Gson gson = new GsonBuilder().setPrettyPrinting().create();
            writer.write(gson.toJson(categorías));
        } catch (IOException e) {
            System.out.println("Error escribiendo archivo JSON: " + e.getMessage());
        }
    }
}
```

EJECUCION:

```
==== MENU SUPERMERCADO ====
1. Agregar categoria
2. Agregar producto a una categoria
3. Ver categorias y productos
4. Salir
Elige una opcion: 1
Ingresa el nombre de la categoria: Lacteo
?? Categoria agregada correctamente.

==== MENU SUPERMERCADO ====
1. Agregar categoria
2. Agregar producto a una categoria
3. Ver categorias y productos
4. Salir
Elige una opcion: 1
Ingresa el nombre de la categoria: Frutas
?? Categoria agregada correctamente.

==== MENU SUPERMERCADO ====
1. Agregar categoria
2. Agregar producto a una categoria
3. Ver categorias y productos
4. Salir
Elige una opcion: 1
Ingresa el nombre de la categoria: Carnes
?? Categoria agregada correctamente.

==== MENU SUPERMERCADO ====
1. Agregar categoria
2. Agregar producto a una categoria
3. Ver categorias y productos
4. Salir
Elige una opcion: 1
Ingresa el nombre de la categoria: Trigo
?? Categoria agregada correctamente.
```

```
3. Ver categorias y productos
4. Salir
Elige una opcion: 2
Categorias disponibles:
1. lacteo
2. carnes
3. frutas
4. trigo
◆ A que categoria quieres agregar el producto? (numero): 1
Nombre del producto: leche
Precio del producto (usa punto para decimales): 1.10
?? Producto agregado correctamente.

==== MENU SUPERMERCADO ====
1. Agregar categoria
2. Agregar producto a una categoria
3. Ver categorias y productos
4. Salir
Elige una opcion: 2
Categorias disponibles:
1. lacteo
2. carnes
3. frutas
4. trigo
◆ A que categoria quieres agregar el producto? (numero): carnes
? Por favor ingresa un numero entero valido.
◆ A que categoria quieres agregar el producto? (numero): 2
Nombre del producto: Hueso
Precio del producto (usa punto para decimales): 3
?? Producto agregado correctamente.
```

```
==== MENU SUPERMERCADO ====
1. Agregar categoria
2. Agregar producto a una categoria
3. Ver categorias y productos
4. Salir
Elige una opcion: 3
Categoria: lacteo
- ID: 1 - leche - $1.1

Categoria: carnes
- ID: 2 - Hueso - $3.0

Categoria: frutas
(sin productos)
Categoria: trigo
(sin productos)

==== MENU SUPERMERCADO ====
1. Agregar categoria
2. Agregar producto a una categoria
3. Ver categorias y productos
4. Salir
Elige una opcion: 4
Saliendo del sistema...
-----
BUILD SUCCESS
-----
Total time: 03:29 min
Finished at: 2025-05-14T22:11:30-05:00
-----
```