Nombre: Ana Vertedor **Fecha:** *14/11/2025*

Ciclo: *Desarrollo de Aplicaciones Web* / ***Asignatura:*** *Desarrollo Web en Entorno Cliente*

UD2: *Estructuras de Control y Manipulación del DOM*

Caso práctico 1 — UD 2
Carrito de la compra de un Supermercado

1. Introducción

En este caso práctico se desarrolla una aplicación web sencilla que simula el funcionamiento de un carrito de la compra en un supermercado. El objetivo principal es registrar los productos introducidos por el usuario, calcular los subtotales y el total general de la compra, y mostrar un resumen final tanto en la consola como en la propia página web.

Para presentar el ejercicio de forma más clara y visual, he integrado la solución dentro de una página web que ya había creado previamente para practicar maquetación y estilos con HTML y CSS. Esta página cuenta con una estructura completa (encabezado, secciones y estilos personalizados), lo que permite que el ejercicio no solo funcione correctamente, sino que también se muestre de manera organizada y estética al usuario.

Dentro de esta misma página añadí una nueva sección dedicada exclusivamente al carrito de la compra, con un botón que inicia el proceso y un contenedor donde se imprime dinámicamente el resumen mediante JavaScript. De esta manera he podido combinar los contenidos de diseño web trabajados anteriormente con los conceptos de programación de la UD2.

Como la página original estaba redactada en inglés, para mantener coherencia visual y estilística he añadido también los textos del ejercicio en inglés (por ejemplo, los mensajes de los botones y encabezados). No afecta al funcionamiento, pero ayuda a mantener la presentación unificada y más profesional.

Este ejercicio permite poner en práctica conceptos clave de JavaScript vistos en la UD2, como el uso de variables, estructuras de control, bucles, conversiones de tipo, arrays de objetos, validaciones y manipulación básica del DOM, mostrando los resultados de forma dinámica en la página y en la consola del navegador.

2. Objetivos del ejercicio

2.1. Solicitar al cliente, mediante un prompt, el número de productos que quiere registrar en su compra.

2.2. Para cada producto introducido, solicitar:

2.2.1. El nombre del producto.

2.2.2. El precio del producto.

2.2.3. La cantidad comprada.

2.3. Al finalizar la introducción de datos, calcular:

2.3.1. El subtotal de cada producto (precio x cantidad).

2.3.2. El total general de la compra (suma de todos los subtotales).

2.4. Para cada producto introducido, solicitar:

2.4.1. El nombre del producto, su precio unitario, la cantidad y el subtotal.

2.4.2. El total general de la compra.

Además de cumplir estos requisitos obligatorios, he añadido dos mejoras opcionales para enriquecer el ejercicio:

2.5. Mostrar también el resultado de forma visual en la página web mediante manipulación del DOM, utilizando document.getElementById() e innerHTML para insertar dinámicamente el resumen en el HTML.

2.6. *Utilizar correctamente estructuras de control, arrays y buenas prácticas de programación, almacenando cada producto como un objeto dentro de un array y organizando la lógica del cálculo de manera clara y estructurada.*

3. Desarrollo de la Solución

Para resolver el caso práctico se ha implementado un archivo HTML con un botón que inicia el proceso de compra.

Al pulsarlo, el script en JavaScript ejecuta las siguientes operaciones:

3.1. Capturade datos

Se solicita mediante prompt():

1. Número total de productos a introducir.
2. Para cada producto:
    - Nombre
    - Precio unitario
    - Cantidad comprada

Los valores numéricos introducidos por el usuario se convierten utilizando parseInt() y parseFloat().

Para evitar errores en el cálculo, se valida el contenido con isNaN().

En caso de que el usuario introduzca valores no numéricos, el programa asigna automáticamente el valor 0 y continúa la ejecución sin detenerse.

Este método permite manejar entradas incorrectas sin que el programa falle, lo cual es apropiado para el nivel de validación requerido en la UD2.

3.2. Almacenamiento del array

Cada producto se guarda como un objeto dentro de un array:

```
{ nombre, precio, cantidad, subtotal }
```

3.3. Cálculo de subtotales y total

En cada iteración del bucle se calcula:

subtotal = precio * cantidad

Y posteriormente se obtiene el total general sumando los subtotales.

3.4. *Salida de datos en la consola*

Se utiliza console.log() para mostrar:

- Nombre del producto
- Precio
- Cantidad
- Subtotal
- Total final

3.5. *Salida de datos en el DOM*

Además, se selecciona un contenedor de la página con:

```
const resultadoDiv = document.getElementById("resultado-carrito");
```

Y se inserta dentro el resumen completo utilizando:

```
resultadoDiv.innerHTML = salida;
```

Esta manipulación del DOM permite al usuario visualizar el resumen de forma clara, organizada y estilizada dentro de la web.

4. Código JavaScript utilizado

```
    alert("JS loaded");
    const botonCarrito =
document.getElementById("iniciar-carrito");
    const resultadoDiv =
document.getElementById("resultado-carrito");

    function exeCart() {

        let numProd = parseInt(
            prompt("How many products do you want to add to your
purchase?")
        );

        if (isNaN(numProd) || numProd <= 0) {
            alert("Please enter a valid number.");
            return;
        }

        let carrito = [];

        for (let i = 0; i < numProd; i++) {

            let nombre = prompt(`Name of product ${i + 1}:`);
```

```
            let precio = parseFloat(prompt(`Unit price of
"${nombre}":`));
            let cantidad = parseInt(prompt(`Units of
"${nombre}":`));

            if (isNaN(precio) || precio < 0) precio = 0;
            if (isNaN(cantidad) || cantidad < 0) cantidad = 0;

            let subtotal = precio * cantidad;

            carrito.push({ nombre, precio, cantidad, subtotal });
        }

        console.log("=== PURCHASE SUMMARY ===");

        let salida = "<h3>Purchase Summary</h3><ul>";
        let totalGeneral = 0;

        for (let p of carrito) {
            console.log(
                `Product: ${p.nombre} | Price: ${p.precio}€ |
Quantity: ${p.cantidad} | Subtotal: ${p.subtotal}€`
            );

            salida += `
                <li>
                    <strong>${p.nombre}</strong> —
                    Price: ${p.precio.toFixed(2)}€ |
                    Quantity: ${p.cantidad} |
                    Subtotal: ${p.subtotal.toFixed(2)}€
                </li>
            `;

            totalGeneral += p.subtotal;
        }

        salida += `</ul>
        <p><strong>Final Total:
${totalGeneral.toFixed(2)}€</strong></p>`;

        resultadoDiv.innerHTML = salida;

        console.log("----------------------------");
        console.log(`FINAL TOTAL: ${totalGeneral}€`);
```

```
        console.log("============================");
    }

    botonCarrito.addEventListener("click", exeCart);
```

5. Código HTML

```
        <!-- NUEVA SECCIÓN: CARRO DE LA COMPRA (CASO PRÁCTICO UD2)
**mantengo en inglés para conservar el estilo del resto de la
página**
        <section class="cart-section">
            <article>
                <h2>Shopping cart</h2>
                    <p> Press the button to register the products
of a purchase. <br>
                    You will be asked for name, price, and
quantity. The summary will appear below. </p>

                <button id="iniciar-carrito"
class="carrito-btn">Start purchase</button>

                <!-- Aquí pintamos el resultado con DOM -->
                <div id="resultado-carrito"
class="carrito-resultado"></div>
            </article>
        </section>
<!--Resto del código aquí -->
<script src="./servicesscript.js"></script>
```

6. Resultados

Tras introducir los datos mediante los prompts, la aplicación muestra:

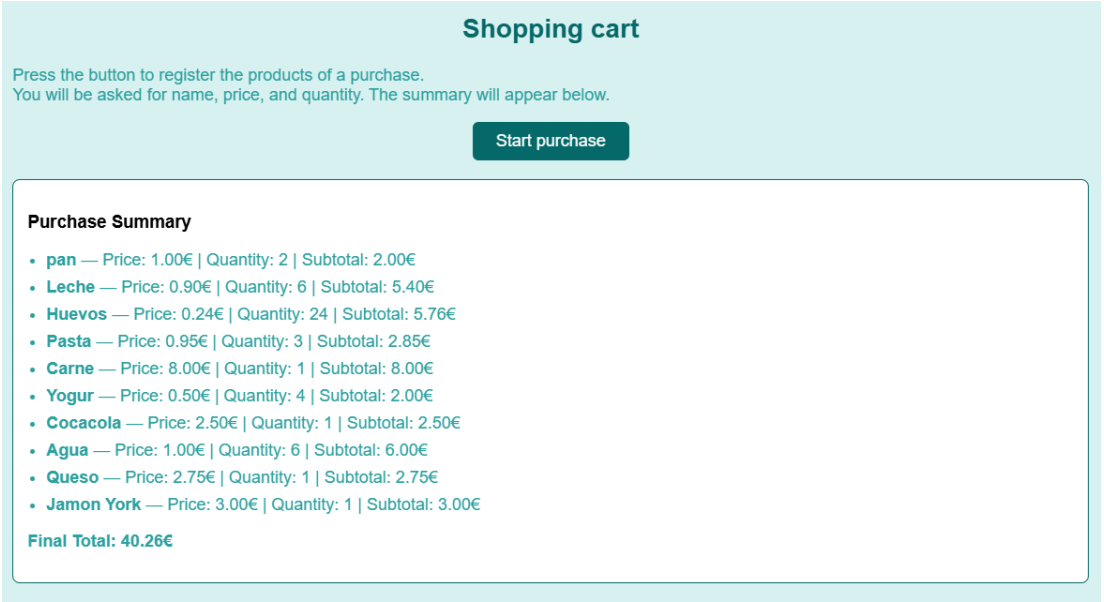
- *En consola:*

  Un resumen detallado con cada producto, su subtotal y el total general.

```
=== PURCHASE SUMMARY ===                                    servicesscript.js:32
Product: pan | Price: 1€ | Quantity: 2 | Subtotal: 2€      servicesscript.js:38
Product: Leche | Price: 0.9€ | Quantity: 6 | Subtotal:      servicesscript.js:38
5.4€
Product: Huevos | Price: 0.24€ | Quantity: 24 |            servicesscript.js:38
Subtotal: 5.76€
Product: Pasta | Price: 0.95€ | Quantity: 3 |             servicesscript.js:38
Subtotal: 2.8499999999999996€
Product: Carne | Price: 8€ | Quantity: 1 | Subtotal:      servicesscript.js:38
8€
Product: Yogur | Price: 0.5€ | Quantity: 4 | Subtotal:    servicesscript.js:38
2€
Product: Cocacola | Price: 2.5€ | Quantity: 1 |           servicesscript.js:38
Subtotal: 2.5€
Product: Agua | Price: 1€ | Quantity: 6 | Subtotal: 6€    servicesscript.js:38
Product: Queso | Price: 2.75€ | Quantity: 1 |             servicesscript.js:38
Subtotal: 2.75€
Product: Jamon York | Price: 3€ | Quantity: 1 |           servicesscript.js:38
Subtotal: 3€
---------------------------                                 servicesscript.js:59
FINAL TOTAL: 40.26€                                         servicesscript.js:60
===========================                                 servicesscript.js:61
```

- *En la página web (DOM):*

  Un bloque visual con la misma información, incluyendo una lista y un total destacado.



Esto mejora la experiencia del usuario y demuestra el uso combinado de consola y DOM.

7. Conclusiones

Este caso práctico ha permitido aplicar de forma práctica los contenidos fundamentales de JavaScript vistos en la UD2:

- Entrada de datos con prompt()
- Declaración y uso de variables
- Conversión de tipos (parseInt, parseFloat)
- Validación de datos con isNaN()
- Uso de bucles y arrays
- Creación de objetos
- Cálculos dinámicos
- Salida por consola
- Manipulación del DOM mediante innerHTML

La integración del carrito dentro de una página web real junto con un estilo adecuado contribuye a reforzar el aprendizaje y a demostrar la utilidad de JavaScript en entornos interactivos.

En conjunto, el ejercicio cumple los objetivos planteados en el enunciado y consolida los fundamentos esenciales de programación en el entorno cliente.

8. Posibles mejoras futuras

Aunque el ejercicio cumple todos los requisitos de la UD2, existen varias mejoras que podrían implementarse en versiones más avanzadas:

Sustituir prompt() por un formulario HTML, lo que permitiría una interacción más cómoda y visual, con campos específicos para nombre, precio y cantidad.

Añadir la posibilidad de editar o eliminar productos, permitiendo al usuario corregir valores sin tener que reiniciar todo el proceso.

Guardar los productos en localStorage, de manera que el carrito se mantenga aunque la página se recargue o el usuario cierre la pestaña.

Mostrar el carrito en tiempo real sin recargar la página, actualizando automáticamente la lista y los totales conforme el usuario añade productos.

Mejorar la validación de datos, de forma que, si el usuario introduce letras o valores no válidos en funciones como prompt(), el programa vuelva a pedir el dato hasta que se introduzca un número correcto. Esta validación más estricta garantizaría que los precios y cantidades siempre sean numéricos y positivos.

9. Referencias

W3Schools. “JavaScript Tutorial.” Disponible en: https://www.w3schools.com/js/

Consultado para repasar sintaxis básica, funciones del lenguaje, manejo de variables, bucles, estructuras de control y ejemplos de manipulación del DOM.

Mimo. Plataforma de aprendizaje interactivo de programación.

Utilizada como apoyo para reforzar conceptos de JavaScript y realizar ejercicios prácticos, especialmente relacionados con variables, bucles, arrays, condicionales y lógica de programación.