Seguimiento de Desarrollo de Aplicaciones Básicas en JavaScript

Objetivo del taller

Desarrollar habilidades fundamentales en programación utilizando JavaScript mediante la creación de tres aplicaciones básicas que empleen arreglos, funciones y estructuras de control, sin necesidad de conceptos avanzados como objetos.

Duración

Aproximadamente 4-6 horas de trabajo autónomo.

Requisitos previos

- Conocimiento básico de JavaScript (variables, console.log, entrada/salida básica como prompt o alert).
- Editor de texto (como VS Code) y un navegador para probar el código.

Instrucciones generales

1. Cada ejercicio debe implementarse en un archivo JavaScript separado (puedes usar app1.js, app2.js, app3.js).
2. Usa prompt() para recibir entrada del usuario y console.log() o alert() para mostrar resultados.
3. No uses objetos ni temas avanzados; limítate a arreglos, funciones y estructuras de control.
4. Prueba cada programa paso a paso para asegurarte de que funcione.
5. Sube a tu repositorio personal.

Ejercicio 1: Sistema de Reservas de Hotel

Descripción

Crea una aplicación que permita reservar habitaciones en un hotel. El hotel tiene 5 habitaciones disponibles (almacenadas en un arreglo). El usuario puede reservar una habitación o liberar una reservada.

Requisitos

- Usa un arreglo para almacenar el estado de las habitaciones (0 = libre, 1 = ocupada).
- Implementa funciones para reservar y liberar habitaciones.
- Usa estructuras de control para validar la opción del usuario y el estado de las habitaciones.

Código base sugerido

```
// Arreglo para las 5 habitaciones (0 = libre, 1 = ocupada)
let habitaciones = [0, 0, 0, 0, 0];
// Función para mostrar estado de habitaciones
const mostrarEstado = () => {
```

```
    let estado = "Estado de habitaciones:\n";
    for (let i = 0; i < habitaciones.length; i++) {
        estado += `Habitación ${i + 1}: ${habitaciones[i] === 0 ?
"Libre" : "Ocupada"}\n`;
    }
    alert(estado);
};
// Función para reservar una habitación
const reservarHabitacion = (num) => {
    if (num < 1 || num > 5) {
        alert("Número de habitación inválido. Usa 1-5.");
    } else if (habitaciones[num - 1] === 1) {
        alert("Habitación ya ocupada.");
    } else {
        habitaciones[num - 1] = 1;
        alert(`Habitación ${num} reservada con éxito.`);
    }
};
// Función para liberar una habitación
const liberarHabitacion = (num) => {
    if (num < 1 || num > 5) {
        alert("Número de habitación inválido. Usa 1-5.");
    } else if (habitaciones[num - 1] === 0) {
        alert("Habitación ya está libre.");
    } else {
        habitaciones[num - 1] = 0;
        alert(`Habitación ${num} liberada con éxito.`);
    }
};
// Menú principal
while (true) {
    let opcion = prompt("1. Ver estado\n2. Reservar\n3. Liberar\n4.
Salir\nElige una opción:");
    if (opcion === "1") {
        mostrarEstado();
    } else if (opcion === "2") {
        let num = parseInt(prompt("Ingresa el número de habitación
(1-5):"));
        reservarHabitacion(num);
```

```
    } else if (opcion === "3") {
        let num = parseInt(prompt("Ingresa el número de habitación
(1-5):"));
        liberarHabitacion(num);
    } else if (opcion === "4") {
        alert("Saliendo...");
        break;
    } else {
        alert("Opción inválida.");
    }
}
```

Tarea

- Ejecuta el código y prueba todas las opciones.
- Modifica el programa para que muestre cuántas habitaciones están libres después de cada operación.

Ejercicio 2: Cajero Automático de Banco

Descripción

Desarrolla un cajero automático que permita consultar saldo, depositar y retirar dinero. Usa un arreglo para almacenar las últimas 5 transacciones.

Requisitos

- Usa un arreglo para registrar las transacciones (números positivos para depósitos, negativos para retiros).
- Implementa funciones para consultar saldo, depositar y retirar.
- Usa estructuras de control para validar saldos y límites.

Tarea

- Implementa el programa con diferentes operaciones.
- Agrega una validación para que no se puedan retirar montos mayores a $500 en una sola transacción.

Ejercicio 3: Cola de Atención de Clientes en Supermercado

Descripción

Simula una cola de atención en un supermercado. Los clientes se agregan a la cola (arreglo) y son atendidos en orden (primero en llegar, primero en ser atendido).

Requisitos

- Usa un arreglo para almacenar los nombres de los clientes en la cola.

- Implementa funciones para agregar un cliente y atender al siguiente.
- Usa estructuras de control para manejar la cola vacía o llena.

Tarea

- Implementa las funciones del programa y prueba agregar y atender clientes.
- Modifica el programa para que la cola tenga una capacidad de 7 clientes en lugar de 5.

Ejercicio 4: Máquina Expendedora de Dulces y Gaseosas

Descripción

Desarrolla una máquina expendedora que permita al usuario insertar una moneda (valor fijo de $1) y seleccionar un producto mediante un código numérico. La máquina tiene un inventario de 5 productos almacenados en arreglos (nombres y cantidades). Si hay stock y el pago es correcto, entrega el producto y actualiza el inventario.

Requisitos

- Usa un arreglo para los nombres de los productos y otro para las cantidades disponibles.
- Implementa funciones para mostrar el inventario, procesar el pago y entregar el producto.
- Usa estructuras de control para validar el código, el pago y el stock.

Tarea

- implementa el programa comprando productos hasta agotar el stock de uno.
- Agrega una funcionalidad para que la máquina devuelva un mensaje si el usuario intenta comprar un producto agotado y sugiera otro producto disponible (por ejemplo, el primero con stock mayor a 0).

Evaluación del taller

1. **Funcionalidad**: ¿Cada programa cumple con los requisitos establecidos?
2. **Corrección**: ¿El código maneja correctamente los casos límite (ej. cola llena, saldo insuficiente)?
3. **Creatividad**: ¿El estudiante implementó la solución de manera creativa?

Siguientes pasos

Una vez completado, el estudiante puede:

- Integrar los tres programas en un solo menú principal.
- Mejorar la interfaz usando console.log con formato más claro.