AE3_ABPRO-Ejercicio grupal [Actividad Evaluada]

AE3_ABPRO-Ejercicio grupal [Actividad Evaluada]

Contexto

El equipo ha sido asignado para desarrollar una parte de una aplicación de gestión de clientes para una empresa. El objetivo es utilizar arreglos y objetos para almacenar y gestionar la información de los clientes. El sistema debe permitir agregar, editar, eliminar y consultar datos de los clientes, al mismo tiempo que permite realizar operaciones con sus datos de manera eficiente utilizando arreglos y ciclos.

Actividad

En este ejercicio grupal, cada miembro del equipo será responsable de una parte del desarrollo, trabajando en conjunto para implementar el sistema de gestión de clientes. Las tareas incluyen el uso de objetos, arreglos, ciclos y buenas prácticas de programación.

1. **Definir un objeto cliente:**
   - Crear una estructura de objeto cliente con las siguientes propiedades:
     - **id** (número)
     - **nombre** (string)
     - **apellido** (string)
     - **email** (string)
     - **telefono** (string)
     - **activo** (booleano: **true** o **false**)
2. **Crear un arreglo de objetos cliente:**
   - Crear un **arreglo vacío** donde se almacenarán los objetos cliente.
   - Definir al menos 3 objetos cliente dentro de este arreglo, cada uno con diferentes datos.
3. **Acceder a la información de un cliente:**
   - Usar un ciclo **for** para iterar sobre el arreglo de clientes y **mostrar los datos de cada cliente** (nombre, apellido, email, teléfono).
4. **Contar la cantidad de clientes activos:**
   - Utilizar un ciclo **for** o **forEach** para contar cuántos clientes tienen el valor **activo** igual a **true**.
5. **Agregar un nuevo cliente:**
   - Crear una **función** que reciba los datos de un nuevo cliente y lo **agregue al arreglo** utilizando el método **push().**

6. **Eliminar un cliente:**

   - Crear una **función** que elimine un cliente por su **id** utilizando el método **splice()** para eliminar el objeto del arreglo.
   - Asegúrese de que, al eliminar el cliente, el arreglo se actualice correctamente.

7. **Modificar los datos de un cliente:**

   - Crear una **función** que modifique los datos de un cliente existente en el arreglo, por ejemplo, cambiar el teléfono o el estado de **activo** de **false** a **true**.

8. **Consultar los clientes inactivos:**

   - Usar el método **filter()** para **crear un nuevo arreglo** con los clientes que tienen el campo **activo** igual a **false**.

9. **Álgebra con arreglos y objetos:**

   - Crear una función que reciba dos arreglos de clientes (por ejemplo, uno con clientes nuevos y otro con clientes existentes) y realice una **unión** de ambos, agregando los nuevos clientes al arreglo original.
   - Crear otra función para **filtrar clientes duplicados** (con el mismo **id**), mostrando solo los clientes únicos.

10. **Ciclos while, for y forEach:**

    - Usar un ciclo **while** para simular una operación repetitiva en el sistema, por ejemplo, consultar los datos de un cliente hasta que se ingrese un **id** válido.
    - Usar **for** o **forEach** para realizar una consulta masiva de todos los clientes, mostrando los datos solo de aquellos que están activos.

11. **Combinación de ciclos con if/else:**

    - Crear una **condición dentro de un ciclo** que determine si un cliente es activo o inactivo, y mostrar un mensaje diferente dependiendo de su estado.

12. **Código limpio y buenas prácticas:**

    - Asegúrese de que el código sea **limpio, modular y fácil de entender**. Utilicen **convenciones** claras para nombrar las variables, funciones y objetos.
    - Cada miembro del equipo debe **documentar su código** con comentarios explicativos sobre qué hace cada parte.

Entrega

- Archivo zip o enlace Github con el código.
- Duración: 1 jornada de clases.
- Entrega: Grupal.

Estado de la entrega

Estado de la entrega	Todavía no se han realizado envíos
Estado de la calificación	Sin calificar
Última modificación	-

DESARROLLO AE3_ABPRO-Ejercicio grupa
SALA - 2

```
// AE3_ABPRO-Ejercicio grupal  - SALA 2
// ------------------------------------------------
// #1 - CREAR UN OBJETO CLIENTE
// ------------------------------------------------

const cliente = {
    id: Number,
    nombre: String,
    apellido: String,
    email: String,
    telefono: String,
    activo: Boolean,
};

// ------------------------------------------------
// # 2 - CREAR UN ARREGLO DE OBJETOS CLIENTE
// ------------------------------------------------let clientes = [];

let clientes = [];
clientes.push({
    id: 1,
    nombre: 'Juan',
    apellido: 'Pérez',
    email: 'juan.perez@example.com',
    telefono: '123456789',
    activo: true,
});

clientes.push({
    id: 2,
    nombre: 'María',
    apellido: 'Gómez',
    email: 'maria.gomez@example.com',
    telefono: '987654321',
    activo: false,
});

clientes.push({
    id: 3,
    nombre: 'Luis',
    apellido: 'Martínez',
    email: 'luis.martinez@example.com',
    telefono: '456789123',
    activo: true,
});
```

```
// ------------------------------------------------
// #3 ACCEDER A INFORMACIÓN DE CLIENTES
// ------------------------------------------------
function mostrarInformacion() {
   for (let i = 0; i < clientes.length; i++) {
      const c = clientes[i];
      console.log(`ID: ${c.id} | Nombre: ${c.nombre} ${c.apellido} | Email: ${c.email} | Teléfono:
${c.telefono} | Activo: ${c.activo ? 'Sí' : 'No'}`);
   }
}
MostrarInformacion();

// ------------------------------------------------
// #3 VERSIÓN 2 SIN FUNCTION
// ------------------------------------------------
for (let i = 0; i < clientes.length; i++) {
   console.log(`
         cliente ${clientes[i].id}
         nombre: ${clientes[i].nombre}
         apellido: ${clientes[i].apellido}
         email: ${clientes[i].email}
         teléfono: ${clientes[i].telefono}
   `);
}

// ------------------------------------------------
// #4 CONTAR LA CANTIDAD DE CLIENTES ACTIVOS
// ------------------------------------------------
function contarActivos() {
   let clientesActivos = 0;
   for (let i = 0; i < clientes.length; i++) {
      if (clientes[i].activo) {
         clientesActivos++;
      }
   }
   console.log('Número de clientes activos: ' + clientesActivos);
}
ContarActivos();

// ------------------------------------------------
// #5 AGREGAR UN NUEVO CLIENTE
// ------------------------------------------------
function agregarCliente() {
   let nombre = prompt('Ingrese un nombre');
   let apellido = prompt(`Ingrese apellido de ${nombre}`);
   let email = prompt(`Ingrese email de ${nombre}`);
   let telefono = prompt(`Ingrese telefono de ${nombre}`);
   let activo = prompt(`¿${nombre} es activo? true/false`) === 'true';

   // como obtener el id y ponerlo adecuadamente sin preguntar con prompt?
   let id = clientes.length + 1;
   clientes.push({ id, nombre, apellido, email, telefono, activo });
```

```
}

agregarCliente();
console.table(clientes);

// ------------------------------------------------
// #6 ELIMINAR UN CLIENTE POR ID
// ------------------------------------------------
function eliminarPorID(id) {
    const index = clientes.findIndex((cliente) => cliente.id === id);
    if (index !== -1) {
        clientes.splice(index, 1);
        console.log(`Cliente con ID ${id} eliminado.`);
    } else {
        console.log(`Cliente con ID ${id} no encontrado.`);
    }
}

eliminarPorID(2);
mostrarInformacion();

// ------------------------------------------------
// #7 MODIFICAR UN CLIENTE
// ------------------------------------------------
function modificarCliente(id, clienteModificado) {
    const index = clientes.findIndex((cliente) => cliente.id === id);
    if (index !== -1) {
        // Aquí puedes modificar las propiedades directamente
        if (clienteModificado.nombre !== undefined) clientes[index].nombre = clienteModificado.nombre;
        if (clienteModificado.apellido !== undefined) clientes[index].apellido = clienteModificado.apellido;
        if (clienteModificado.email !== undefined) clientes[index].email = clienteModificado.email;
        if (clienteModificado.telefono !== undefined) clientes[index].telefono = clienteModificado.telefono;
        if (clienteModificado.activo !== undefined) clientes[index].activo = clienteModificado.activo;
        console.log(`Cliente con ID ${id} modificado.`);
    } else {
        console.log(`Cliente con ID ${id} no encontrado.`);
    }
}

modificarCliente(1, {
    nombre: 'Carlos',
    apellido: 'Ramírez',
    email: 'carlos.ramirez@example.com',
    telefono: '111222333',
    activo: false,
});
mostrarInformacion();
```

```
// ---------------------------------------------------
// #8 CONSULTAR CLIENTES INACTIVOS
// ---------------------------------------------------
function consultarInactivos() {
    const inactivos = clientes.filter((cliente) => !cliente.activo);
    if (inactivos.length > 0) {
        console.log('Clientes inactivos:');
        inactivos.forEach((cliente) => {
            console.log(`ID: ${cliente.id}, Nombre: ${cliente.nombre} ${cliente.apellido}`);
        });
    } else {
        console.log('No hay clientes inactivos.');
    }
}

consultarInactivos();

// ---------------------------------------------------
// #8 ALTERNATIVA 2
// ---------------------------------------------------
let nuevoArreglo = clientes.filter((cliente) => cliente.activo === false);
console.table(nuevoArreglo);

// -----------------------------------------------------
// #9  FUNCIÓN PARA UNIR ARREGLOS DE CLIENTES
// -----------------------------------------------------
function unirClientes(nuevosClientes) {
    if (Array.isArray(nuevosClientes)) {
        clientes = [...clientes, ...nuevosClientes];
        console.log('Clientes unidos exitosamente.');
    } else {
        console.log('El argumento debe ser un arreglo de clientes.');
    }
}

unirClientes([
    {
        id: 5,
        nombre: 'Pedro',
        apellido: 'Sánchez',
        email: 'pedro.sanchez@example.com',
        telefono: '654321987',
        activo: true,
    },
]);
mostrarInformacion();
```

```
// ------------------------------------------------------------
// #9 FILTRAR CLIENTES ÚNICOS (SACAR LOS DUPLICADOS)
// ------------------------------------------------------------
function filtrarClientesUnicos(lista = []) {
   const map = new Map();
   lista.forEach((cliente) => {
      if (!map.has(cliente.id)) {
         map.set(cliente.id, cliente);
      }
   });
   return Array.from(map.values());
}

clientes = filtrarClientesUnicos(clientes);
console.log('Clientes únicos:', clientes);

// ------------------------------------------------------------------------------------------------------------
// #10 CICLO WHILE PARA CONSULTAR DATOS DE CLIENTE HASTA OBTENER ID VÁLIDO CON PROMPT
// ------------------------------------------------------------------------------------------------------------
//function buscarCliente() {
//   let idValido = false;
//   let cliente;
//   while (!idValido) {
//      const entrada = prompt('Ingrese el ID del cliente:');
//      const id = parseInt(entrada);
//      cliente = clientes.find((c) => c.id === id);
//      if (cliente) {
//         idValido = true;
//      } else {
//         alert(`No se encontró cliente con id ${id}. Intenta de nuevo`);
//      }
//   }
//   console.log('Cliente encontrado:');
//   console.table(cliente);
//}

// Simula intentos con un arreglo de IDs a consultar
console.log('------------------------------------------------------------------------------------------------------------');
console.log('#10 - 1 CICLO WHILE PARA CONSULTAR DATOS DE CLIENTE HASTA OBTENER ID VÁLIDO');
console.log('------------------------------------------------------------------------------------------------------------');
function consultarClienteHastaValido(idInicial) {
  const intentos = [, 99, 12, idInicial, 33, 1]; // lista simulada de intentos
  let cliente = null;
  let i = 0;

  while (!cliente && i < intentos.length) {
    const id = intentos[i];
    console.log(`Intento con ID: ${id}`);

    const encontrado = clientes.find(c => c.id === id && c.activo);
    if (encontrado) {
      cliente = encontrado;
```

```
      break;
    }
    i++;
  }

  if (cliente) {
    console.log("Cliente encontrado:");
    console.log(`ID: ${cliente.id}`);
    console.log(`Nombre: ${cliente.nombre} ${cliente.apellido}`);
    console.log(`Email: ${cliente.email}`);
    console.log(`Teléfono: ${cliente.telefono}`);
  } else {
    console.log("No se encontró un cliente válido.");
  }
}

consultarClienteHastaValido(3);
// ------------------------------------------------------------------------------------
// #10 FOR PARA REALIZAR CONSULTA MASIVA DE TODOS LOS CLIENTES ACTIVOS
// ------------------------------------------------------------------------------------
for (let i = 0; i < clientes.length; i++) {
    if (clientes[i].activo) {
        console.log(`
                ID: ${clientes[i].id}
                Nombre: ${clientes[i].nombre}
                Apellido: ${clientes[i].apellido}
                Email: ${clientes[i].email}
                Teléfono: ${clientes[i].telefono}
    `);
    }
}

// ------------------------------------------------
// #11 COMBINACION DE CICLO CON IF/ELSE
// ------------------------------------------------
clientes.forEach((cliente) => {
    if (cliente.activo) {
        console.log(`${cliente.nombre} está activo`);
    } else {
        console.log(`${cliente.nombre} está inactivo`);
    }
});
```