UTNMDP
Regional Mar del Plata

Actividad Práctica 3 – Introducción a JS

1. Crear un arreglo llamado **productos** que contenga al menos 5 objetos literales con las propiedades: id, nombre, precio y stock.
    - Agregar un nuevo producto usando push().
    - Eliminar el último producto con pop().
    - Mostrar el listado final en consola.

2. Usar **filter()** con una función **anónima** para obtener los productos cuyo stock sea mayor a 10. Luego, guardar el resultado en un nuevo arreglo productosEnStock y mostrarlo.

3. Usar **map()** con una **arrow function** para crear un nuevo arreglo donde todos los productos tengan el precio incrementado en un 20%.Mostrar el nuevo arreglo.

4. Usar **find()** (investigar método) con una función callback para encontrar un producto por su nombre. Si lo encuentra, mostrarlo en consola. Si no, mostrar el mensaje "Producto no encontrado".

5. Usar **reduce()** con una **arrow function** para calcular el precio total de todos los productos. Luego, usar **reduce()** nuevamente para calcular el promedio de precios.

6. Usar **some()** para verificar si existe al menos un producto con stock igual a 0. Luego, usar **every()** para comprobar si todos los productos cuestan más de 100. Mostrar ambos resultados. Investigar ambos métodos.

7. Crear un arreglo clientes con al menos 3 objetos literales que tengan las propiedades: id, nombre, edad, compras (array de strings). Usar **forEach()** para mostrar en consola el nombre de cada cliente junto con la cantidad de compras que realizó.

8. Crear una función *procesarClientes(clientes, callback)* que reciba el arreglo de clientes y una función de callback. Llamar a procesarClientes con distintos callbacks:

- Un callback que muestre solo los nombres.
- Otro callback que muestre solo la cantidad total de compras.

9. Crear un arreglo de números y ordenarlo en forma ascendente con **sort()** y una **arrow function.** Ordenar los mismos números en forma descendente.

10. Crear un objeto literal tienda que tenga:

- Un arreglo productos.
- Un método vender(idProducto, cantidad) que:
  - Busque el producto por id usando find().
  - Si hay stock suficiente, reste la cantidad al stock y muestre "Venta realizada".
  - Si no hay stock, muestre "Stock insuficiente".

Probar el método vendiendo algunos productos.

11. Crear un arreglo carrito vacío.

- Usar **push()** para agregar objetos con las propiedades: producto, cantidad, precioUnitario.
- Usar **reduce()** para calcular el total a pagar.
- Usar **map()** para generar un arreglo con el detalle de cada ítem en formato: "Producto X - Cantidad Y - Subtotal Z".
- Mostrar el detalle y el total en consola.

12. Crear un arreglo libros, cada elemento debe ser un objeto con: id, titulo, autor, genero, disponible (booleano). Luego:

- Usar **filter()** para obtener todos los libros de un género específico.
- Usar **map()** para generar un arreglo con solo los títulos en mayúscula.
- Crear una función prestarLibro(id) que:
  - Busque el libro con **find()**.
  - Si está disponible, lo marque como no disponible.
  - Si no, devuelva "No disponible".

13. Crear un objeto literal agenda con un arreglo contactos que guarde objetos { id, nombre, telefono }. Luego, implementar los siguientes métodos en la agenda:

- agregarContacto(contacto) usando **push()**.
- eliminarContacto(id) usando **filter()**.
- buscarContacto(nombre) usando **find()**.
- listarContactos() que muestre todos.

14. Crear un arreglo alumnos con objetos { id, nombre, notas: [números] }. Luego:

- Usar **map()** + **reduce()** para calcular el promedio de cada alumno.
- Generar un nuevo arreglo con objetos { nombre, promedio }.
- Filtrar solo los aprobados (promedio >= 6).
- Mostrar en consola la lista de aprobados.

15. Crear un arreglo productos con objetos { id, nombre, precio, stock }. Luego definir una función comprar(id, cantidad, callbackExito, callbackError) que:

- Si hay stock suficiente → descuente y ejecute callbackExito mostrando el detalle de la compra.
- Si no hay stock → ejecute callbackError.
- Probar la función con distintas compras.

16. Crear un objeto literal playlist con:

- Propiedad canciones (array de objetos { id, titulo, artista, duracion }).
- Método *agregarCancion()*.
- Método *eliminarCancion(id)* usando **filter()**.
- Método *duracionTotal()* usando **reduce()**.
- Método *buscarPorArtista(nombre)* usando **filter().**

17. Crear un arreglo de números al azar (al menos 15) harcodeados.

- Usar **filter()** para obtener los pares.
- Usar **map()** para elevarlos al cuadrado.
- Usar **reduce()** para calcular la suma de los cuadrados.

18. Crear un arreglo carrito vacío.

- Definir una función *agregarAlCarrito(producto, cantidad, precioUnitario)* que use **push()**.
- Definir una función *calcularTotal()* con *reduce().*
- Definir una función *filtrarPorPrecio(min, max)* que devuelva los productos dentro de ese rango.
- Definir una función *generarDetalle()* que devuelva un arreglo de strings como: "2x Mouse = $50".