Resumen de los métodos de objetos y arrays con explicaciones y ejemplos.

Métodos de Objetos

1. Object.keys()
- Descripción: Devuelve un array con las claves (keys) de un objeto.
- Uso: Se utiliza cuando necesitas acceder o iterar sobre las propiedades de un objeto.
- Ejemplo:
  typescript
  const producto = { nombre: "Laptop", precio: 1000, stock: 10 };
  const keys = Object.keys(producto);
  console.log(keys); // ["nombre", "precio", "stock"]

2. Object.values()
- Descripción: Devuelve un array con los valores de las propiedades de un objeto.
- Uso: Útil para acceder a los valores de las propiedades sin las claves.
- Ejemplo:
  typescript
  const producto = { nombre: "Laptop", precio: 1000, stock: 10 };
  const values = Object.values(producto);
  console.log(values); // ["Laptop", 1000, 10]

3. Object.entries()
- Descripción: Devuelve un array de arrays, donde cada sub-array es un par [clave, valor] del objeto.
- Uso: Ideal para iterar sobre las propiedades y sus valores al mismo tiempo.
- Ejemplo:
  typescript
  const producto = { nombre: "Laptop", precio: 1000, stock: 10 };
  const entries = Object.entries(producto);
  console.log(entries); // [["nombre", "Laptop"], ["precio", 1000], ["stock", 10]]

Métodos de Arrays

1. forEach()
- Descripción: Ejecuta una función por cada elemento de un array. No devuelve un nuevo array.
- Uso: Ideal para ejecutar acciones en cada elemento de un array sin modificarlo.
- Ejemplo:
  typescript
  const productos = ["Laptop", "Mouse", "Teclado"];
  productos.forEach((producto) => console.log(producto));
  // Output: "Laptop", "Mouse", "Teclado"

2. map()
- Descripción: Crea un nuevo array con los resultados de aplicar una función a cada elemento del array original.
- Uso: Se usa para transformar los elementos de un array.
- Ejemplo:
  typescript
  const precios = [1000, 20, 30];
  const preciosConImpuesto = precios.map((precio) => precio 1.21);
  console.log(preciosConImpuesto); // [1210, 24.2, 36.3]

3. filter()
- Descripción: Crea un nuevo array con los elementos que cumplen una condición.
- Uso: Se usa para filtrar elementos que cumplen un criterio.
- Ejemplo:
  typescript
  const productos = [
    { nombre: "Laptop", precio: 1000 },
    { nombre: "Mouse", precio: 20 },
    { nombre: "Teclado", precio: 30 }
  ];
  const productosBaratos = productos.filter((producto) => producto.precio < 100);
  console.log(productosBaratos); // [{ nombre: "Mouse", precio: 20 }, { nombre: "Teclado", precio: 30 }]

4. find()
- Descripción: Devuelve el primer elemento que cumpla con una condición, o undefined si no lo encuentra.
- Uso: Útil para encontrar un solo elemento en un array.
- Ejemplo:
  typescript
  const productos = [
    { nombre: "Laptop", precio: 1000 },
    { nombre: "Mouse", precio: 20 },
    { nombre: "Teclado", precio: 30 }
  ];
  const teclado = productos.find((producto) => producto.nombre === "Teclado");
  console.log(teclado); // { nombre: "Teclado", precio: 30 }

5. reduce()
- Descripción: Aplica una función a un acumulador y a cada elemento del array para reducir el array a un solo valor.
- Uso: Ideal para sumar, concatenar o realizar cálculos acumulativos sobre arrays.
- Ejemplo:

```
typescript
const precios = [1000, 20, 30];
const total = precios.reduce((acumulador, precio) => acumulador + precio, 0);
console.log(total); // 1050
```

Gestión de Tienda

Estructura del Objeto:
Crear un objeto tienda que contenga un array de productos.
Cada producto debe tener tres propiedades: nombre, precio, y stock.
Tipar tanto el objeto tienda como los productos usando interfaces de TypeScript.

Métodos del Objeto Tienda:

Crear producto:
Crea un producto, recibe nombre, precio y stock.

Actualizar Precio:
Crear un método actualizarPrecio que reciba el nombre de un producto y un nuevo precio. El método debe buscar el producto en el array y actualizar su precio si se encuentra.

Vender Producto:
Crear un método venderProducto que reciba el nombre del producto y la cantidad vendida. El método debe verificar si el producto existe y si el stock es suficiente, luego reducir el stock en la cantidad vendida.

Productos en Oferta:
Añadir un método productosEnOferta que reciba un precio máximo. Este método debe filtrar los productos cuyo precio sea menor o igual a ese valor y mostrarlos.

Mostrar Inventario:
Usar Object.keys() para mostrar el nombre y cantidad en stock de cada producto.

Valor Total del Inventario:
Calcular el valor total de todos los productos en stock usando Object.values() y reduce().

Pruebas del Sistema:

Realizar una prueba llamando a los métodos actualizarPrecio, venderProducto, productosEnOferta, mostrarInventario, y valorTotalInventario para verificar que todo funcione correctamente.