/\* En el archivo tarea2.js podemos encontrar un código de un supermercado que vende

productos. El código contiene - una clase Producto que representa un producto que vende el

super - una clase Carrito que representa el carrito de compras de un cliente - una clase

ProductoEnCarrito que representa un producto que se agrego al carrito - una función

findProductBySku que simula una base de datos y busca un producto por su sku El código

tiene errores y varias cosas para mejorar / agregar Ejercicios 1) Arreglar errores existentes en el

código a) Al ejecutar agregarProducto 2 veces con los mismos valores debería agregar 1 solo

producto con la suma de las cantidades. b) Al ejecutar agregarProducto debería actualizar la

lista de categorías solamente si la categoría no estaba en la lista. c) Si intento agregar un

producto que no existe debería mostrar un mensaje de error. 2) Agregar la función

eliminarProducto a la clase Carrito a) La función eliminarProducto recibe un sku y una cantidad

(debe devolver una promesa) b) Si la cantidad es menor a la cantidad de ese producto en el

carrito, se debe restar esa cantidad al producto c) Si la cantidad es mayor o igual a la cantidad

de ese producto en el carrito, se debe eliminar el producto del carrito d) Si el producto no existe

en el carrito, se debe mostrar un mensaje de error e) La función debe retornar una promesa 3)

Utilizar la función eliminarProducto utilizando .then() y .catch() \*/ // Cada producto que vende el

super es creado con esta clase class Producto { sku; // Identificador único del producto nombre;

// Su nombre categoria; // Categoría a la que pertenece este producto precio; // Su precio stock;

// Cantidad disponible en stock constructor(sku, nombre, precio, categoria, stock) { this.sku =

sku; this.nombre = nombre; this.categoria = categoria; this.precio = precio; // Si no me definen

stock, pongo 10 por default if (stock) { this.stock = stock; } else { this.stock = 10; } } } // Creo

todos los productos que vende mi super const queso = new Producto('KS944RUR', 'Queso', 10,

'lacteos', 4); const gaseosa = new Producto('FN312PPE', 'Gaseosa', 5, 'bebidas'); const cerveza

= new Producto('PV332MJ', 'Cerveza', 20, 'bebidas'); const arroz = new Producto('XX92LKI',

'Arroz', 7, 'alimentos', 20); const fideos = new Producto('UI999TY', 'Fideos', 5, 'alimentos'); const

lavandina = new Producto('RT324GD', 'Lavandina', 9, 'limpieza'); const shampoo = new

Producto('OL883YE', 'Shampoo', 3, 'higiene', 50); const jabon = new Producto('WE328NJ',

'Jabon', 4, 'higiene', 3); // Genero un listado de productos. Simulando base de datos const

productosDelSuper = [queso, gaseosa, cerveza, arroz, fideos, lavandina, shampoo, jabon]; //

Cada cliente que venga a mi super va a crear un carrito class Carrito { productos; // Lista de

productos agregados categorias; // Lista de las diferentes categorías de los productos en el

carrito precioTotal; // Lo que voy a pagar al finalizar mi compra // Al crear un carrito, empieza

vació constructor() { this.precioTotal = 0; this.productos = []; this.categorias = []; } /\*\* \* función

que agrega @{cantidad} de productos con @{sku} al carrito \*/ async agregarProducto(sku,

cantidad) { console.log(`Agregando ${cantidad} ${sku}`); // Busco el producto en la "base de

datos" const producto = await findProductBySku(sku); console.log("Producto encontrado",

producto); // Creo un producto nuevo const nuevoProducto = new ProductoEnCarrito(sku,

producto.nombre, cantidad); this.productos.push(nuevoProducto); this.precioTotal =

this.precioTotal + (producto.precio \* cantidad); this.categorias.push(producto.categoria); } } //

Cada producto que se agrega al carrito es creado con esta clase class ProductoEnCarrito { sku;

// Identificador único del producto nombre; // Su nombre cantidad; // Cantidad de este producto

en el carrito constructor(sku, nombre, cantidad) { this.sku = sku; this.nombre = nombre;

this.cantidad = cantidad; } } // Función que busca un producto por su sku en "la base de datos"

function findProductBySku(sku) { return new Promise((resolve, reject) => { setTimeout(() => {

const foundProduct = productosDelSuper.find(product => product.sku === sku); if

(foundProduct) { resolve(foundProduct); } else { reject(`Product ${sku} not found`); } }, 1500); });

} const carrito = new Carrito(); carrito.agregarProducto('WE328NJ', 2);