Ejercicio 1: Obtener claves y valores de un objeto

Utilizando el método Object.keys() y Object.values(), escribe una función que reciba un objeto producto (como product1) y devuelva un mensaje que muestre todas las claves y todos los valores.

Instrucciones:

1. Usa Object.keys() para obtener todas las claves del objeto.

2. Usa Object.values() para obtener todos los valores del objeto.

3. Muestra en consola un mensaje que combine las claves y los valores.

Ejemplo de entrada:

javascript

const product1 = {

serial: '111-111-111',

name: 'Portátil',

price: 750,

};

Salida esperada:

Claves: serial, name, price

Valores: 111-111-111, Portátil, 750

Ejercicio 2: Fusionar dos objetos usando Object.assign()

Tienes dos objetos de productos. Usa el método Object.assign() para fusionarlos en un solo objeto. Si una propiedad se repite, el segundo objeto tiene prioridad.

Instrucciones:

1. Declara dos objetos que representen productos diferentes.

2. Usa Object.assign() para combinarlos en uno solo.

3. Muestra el resultado en la consola.

Ejemplo:

javascript

const product1 = {

serial: '111-111-111',

name: 'Portátil',

price: 750,

};

const product2 = {

serial: '111-111-112',

name: 'Tablet',

stock: 20,

};

Salida esperada:

javascript

{

serial: '111-111-112',

name: 'Tablet',

price: 750,

stock: 20

}

Ejercicio 3: Crear una función constructora

Crea una función constructora llamada Product que permita crear objetos del mismo tipo que product1.

Instrucciones:

1. Crea una función constructora que reciba como parámetros el número de serie, el nombre y el precio.

2. Agrega un método a la función constructora llamado fullName que devuelva el número de serie y el nombre del producto en un string, como en el objeto product1.

3. Crea una nueva instancia de Product usando la función constructora.

Ejemplo de uso:

javascript

function Product(serial, name, price) {

// Tu código aquí

}

const product3 = new Product('111-111-113', 'Teléfono', 500);

console.log(product3.fullName());

Salida esperada:

111-111-113: Teléfono

Ejercicio 4: Crear un objeto usando Object.create()

Usa Object.create() para crear un nuevo objeto basado en otro. Asegúrate de que el nuevo objeto tenga una propiedad adicional llamada category.

Instrucciones:

1. Usa Object.create() para crear un nuevo objeto basado en product1.

2. Agrega una nueva propiedad category al objeto creado.

3. Muestra el nuevo objeto en consola.

Ejemplo de uso:

javascript

const product1 = {

serial: '111-111-111',

name: 'Portátil',

price: 750,

};

const newProduct = Object.create(product1);

newProduct.category = 'Electrónica';

console.log(newProduct);

console.log(newProduct.fullName()); // Método heredado de product1

Salida esperada:

javascript

{

category: 'Electrónica',

serial: '111-111-111',

name: 'Portátil',

price: 750

}

111-111-111: Portátil

Ejercicio 5: Comparar objetos y evitar duplicados

Crea una función que reciba dos objetos producto y los compare para evitar duplicados basados en el número de serie.

Instrucciones:

1. Crea una función que acepte dos objetos y compare sus números de serie.

2. Si los números de serie son iguales, la función debe devolver un mensaje que indique que el producto ya existe.

3. Si no son iguales, debe fusionarlos usando Object.assign() y mostrar el nuevo objeto.

Ejemplo de uso:

javascript

function mergeProducts(productA, productB) {

// Tu código aquí

}

const productA = { serial: '111-111-111', name: 'Portátil', price: 750 };

const productB = { serial: '111-111-111', name: 'Laptop', stock: 10 };

mergeProducts(productA, productB);

Salida esperada:

Producto ya existente.