Ejercicio 1:

1. Crea una función crearUsuario que reciba un nombre y un apellido, y devuelva un objeto con esas propiedades.
2. La función debe también incluir una propiedad nombreCompleto que retorne "Nombre Apellido".
3. La variable nombreCompleto debe definirse dentro de la función (Scope local).

Ejercicio 2:

1. Crea un objeto coche con las siguientes propiedades:
    a. marca: "Toyota"
    b. modelo: "Corolla"
    c. año: 2022
    d. detalles(): Devuelve "Toyota Corolla - Año 2022"
2. Agrega dinámicamente la propiedad color con valor "Rojo".
3. Modifica el modelo a "Camry".
4. Borra la propiedad año.

Ejercicio 4:

1. Dado un array de objetos representando productos, ordénalo por precio de menor a mayor.

```
let productos = [
{ nombre: "Teclado", precio: 25 },
{ nombre: "Ratón", precio: 15 },
{ nombre: "Monitor", precio: 200 },
{ nombre: "Auriculares", precio: 50 }
];
```

Ejercicio 5:

Crea una función flecha llamada **esPar** que reciba un número y devuelva true si es par o false si es impar.

Ejercicio 6:

Desarrolla un programa en JavaScript que permita a un usuario jugar contra el ordenador al clásico juego de Piedra, Papel o Tijera. El usuario podrá elegir entre "piedra", "papel" o "tijera", mientras que la maquina seleccionará aleatoriamente una de estas opciones. El programa determinará el ganador según las reglas del juego:

Piedra gana contra Tijera, Tijera gana contra Papel y Papel gana contra Piedra; si ambos jugadores eligen la misma opción, el juego termina en empate. La función debe recibir como parámetro la elección del usuario y devolver un mensaje indicando la elección de la computadora y el resultado de la partida. Además, el código debe manejar entradas no válidas asegurándose de que el usuario solo pueda ingresar una de las tres opciones permitidas.

Ejercicio 7:

Desarrolla un programa en JavaScript que simule el funcionamiento de un cajero automático mediante un objeto que gestione el saldo de una cuenta bancaria. El sistema debe permitir al usuario realizar tres operaciones principales: ingresar dinero, retirar dinero y consultar el saldo disponible. Para ello, el objeto debe contar con un atributo saldo y tres métodos: ingresar(cantidad), que suma la cantidad ingresada al saldo; retirar(cantidad), que descuenta la cantidad solicitada si hay saldo suficiente, mostrando un mensaje de error en caso contrario; y consultarSaldo(), que muestra el saldo actual. Asegúrate de validar que las cantidades ingresadas sean números positivos y maneja posibles errores, como intentos de retirar más dinero del disponible.