ORIENTACIÓN

Si estás ejecutando un script en Node.js (fuera del navegador), puedes usar readline:

```
const readline = require("readline");

const rl = readline.createInterface({
 input: process.stdin,
 output: process.stdout
});

rl.question("Escribe tu nombre: ", function(nombre) {
 console.log("Hola, " + nombre);
 rl.close();
});
```

Ejemplo más completo (leer varios datos)

```
const readline = require("readline");

const rl = readline.createInterface({
 input: process.stdin,
 output: process.stdout
});

rl.question("¿Cuál es tu nombre? ", (nombre) => {
 rl.question("¿Cuántos años tienes? ", (edad) => {
  console.log(`Hola ${nombre}, tienes ${edad} años.`);
  rl.close();
 });
});
```

Práctica – Flujo & Funciones

Ejercicio 1: Control de flujo — Calculadora de descuentos

Crea un programa que pida al usuario:

- El precio original de un producto (por ejemplo 100).
- La categoría del cliente: "estudiante", "jubilado", "general".

Aplica los siguientes descuentos:

- Estudiante: 20%
- Jubilado: 15%
- General: 0%

Muestra el precio final. Usa if/else if/else.

Ejercicio 2: switch — Saludar por idiomas

Crea una función saludoPorIdioma(code) que reciba un código de idioma ('es', 'en', 'fr', 'de') y devuelva un saludo corto:

- 'es' → 'Hola'
- 'en' → 'Hello'
- 'fr' → 'Bonjour'
- 'de' → 'Hallo'

Si el código no está en la lista devolver 'Idioma no soportado'.

Ejercicio 3: Funciones y cierres — Contador personalizado

Crea una función crearContador(inicio, paso) que devuelva una función. Esta función interna debe:

- Incrementar un contador interno en el valor de paso cada vez que se llama.
- Devolver el valor actual del contador.

Usa cierres para mantener el estado del contador.

Ejercicio 4: Arrays y Objetos — Gestor de tareas

Crear un **gestor de tareas básico** utilizando **arrays y objetos en JavaScript**,

El programa debe permitir gestionar tareas mediante funciones que el usuario puede invocar manualmente.

1. Agregar tareas
2. Listar tareas
3. Marcar tareas como completadas
4. Eliminar tareas
5. Salir del programa

La estructura sería:

PROPIEDAD	TIPO	DESCRIPCIÓN
Id	Number	Identificador único
Descripción	String	Texto que describe la tarea
completada	boolean	Estado de la tarea. f*alse* = pendiente, *true* = completada

Ejemplo del array:

```
let tareas = [
  {
    id: 1718234567890,
    descripcion: "Aprender arrays",
    completada: true
  },
  {
    id: 1718234567895,
    descripcion: "Practicar métodos de array",
    completada: false
  },
  {
    id: 1718234567900,
    descripcion: "Hacer ejercicio",
    completada: false
  }
];
```