FASE 1: Fundamentos Sólidos (25 Ejercicios)
Enfócate en la sintaxis y en entender qué devuelve cada método.

Nivel Básico:

1. **Suma total:** Dado un array de números, obtén la suma usando forEach.
2. **Duplicador:** Crea un nuevo array con los valores multiplicados por 2 usando map.
3. **Solo pares:** Filtra los números pares de un array usando filter.
4. **Buscador de nombres:** Dado un array de nombres, filtra los que tienen más de 5 letras.
5. **Convertidor a String:** Usa map para convertir un array de números en un array de strings.
6. **Contador de caracteres:** Dado un array de palabras, crea uno nuevo con la longitud de cada palabra.
7. **Existencia:** Verifica si el número 10 existe en un array usando includes.
8. **Primer negativo:** Encuentra el primer número negativo usando find.
9. **Todos positivos:** Verifica si todos los números son mayores a 0 usando every.
10. **Algún VIP:** Verifica si al menos un objeto usuario tiene la propiedad isVip: true usando some.

Manipulación de Objetos:

11. **Listar llaves:** Dado un objeto persona, imprime todas sus llaves usando Object.keys.
12. **Listar valores:** Dado un objeto producto, imprime sus valores con Object.values.
13. **Array de arrays:** Convierte un objeto en una lista de pares [llave, valor] con Object.entries.
14. **De pares a objeto:** Usa Object.fromEntries para convertir [['color', 'rojo'], ['talla', 'M']] en un objeto.
15. **Contar propiedades:** Función que retorne cuántas propiedades tiene un objeto sin usar un contador manual.

El temido reduce (Intro):

16. **Suma con reduce:** Multiplica todos los números de un array usando reduce.
17. **Concatenador:** Une un array de strings en una sola frase usando reduce.
18. **Máximo valor:** Encuentra el número más grande de un array usando reduce.
19. **Contar booleanos:** Dado un array de [true, false, true, true], cuenta cuántos true hay.
20. **Flatten simple:** Convierte [[1,2], [3,4], [5,6]] en [1,2,3,4,5,6].

Desafíos de Refactorización (Re-escritura):

21. **For a Map:** Toma un ejercicio de for que llenaba un array vacío y conviértelo a map.
22. **Filter a Reduce:** Filtra números mayores a 10 usando solo reduce.
23. **Find a For...of:** Implementa la lógica de find manualmente.
24. **Map a Reduce:** Crea un nuevo array con los cuadrados de los números usando reduce.
25. **Entries a ForEach:** Recorre un objeto usando Object.entries y un forEach.

FASE 2: Lógica Intermedia (20 Ejercicios)
Aquí el objetivo es la transformación y el manejo de estados.

26. **Agrupación por tipo:** Tienes un array de productos con categoria. Agrúpalos en un objeto donde las llaves sean las categorías.
27. **Frecuencia de palabras:** Dado un string, devuelve un objeto con cuántas veces aparece cada palabra.
28. **Carrito de compras:** Calcula el total a pagar de un array de objetos {precio, cantidad}.
29. **Ranking:** Ordena un array de jugadores por su puntaje de mayor a menor (sort).
30. **Eliminar duplicados:** Limpia un array de números repetidos usando filter e indexOf.
31. **Intersección de arrays:** Encuentra los elementos comunes entre dos arrays.
32. **Diferencia de arrays:** Encuentra los elementos que están en A pero no en B.
33. **Clonación de usuario:** Crea una función que reciba un objeto y devuelva una copia (shallow copy).
34. **Merge de perfiles:** Combina un objeto usuario con un objeto configuracion, donde la configuración prevalece.

35. **CSV a JSON:** Convierte un array de strings tipo ["nombre,apellido", "Juan,Perez"] en un array de objetos.
36. **Filtro multidimensional:** Filtra usuarios que sean mayores de 18 **Y** vivan en una ciudad específica.
37. **Update inmutable:** Dado un array de objetos, cambia el nombre de un usuario por su ID sin modificar el array original.
38. **Suma condicional:** Suma solo los precios de los productos que están en stock.
39. **Validador de esquema:** Verifica si un objeto tiene todas las llaves requeridas: ['id', 'name', 'email'].
40. **Extraer IDs:** Dado un objeto complejo con usuarios anidados, extrae todos los IDs a un array plano.

FASE 3: Nivel PRO (15 Ejercicios)
Pensamiento algorítmico, recursividad y optimización.

41. **Deep Clone:** Escribe una función que clone un objeto con múltiples niveles de profundidad (sin usar JSON.parse).
42. **Deep Equal:** Función que compare si dos objetos son idénticos en contenido, no solo en referencia.
43. **Indexación:** Convierte un array de 10,000 usuarios en un objeto donde la llave sea el id (para búsqueda $O(1)$).
44. **Normalización:** Transforma una respuesta de API con objetos anidados en un estado "plano" (estilo Redux).
45. **Árbol a Lista:** Convierte una estructura de carpetas anidadas en un array plano de rutas.
46. **Lista a Árbol:** Toma un array de objetos con id y parentId y conviértelo en una estructura jerárquica.
47. **Implementar MyMap:** Crea tu propia versión de .map() desde cero usando for.
48. **Implementar MyReduce:** Crea tu propia versión de .reduce() desde cero.
49. **Pipe Function:** Crea una función que reciba un valor y una serie de funciones, aplicándolas en orden.
50. **Deep Merge:** Combina dos objetos que tienen objetos anidados, uniendo las propiedades internas.
51. **Agrupación Multinivel:** Agrupa una lista de ventas por Año, y luego por Mes.
52. **Paginación manual:** Crea una función que reciba un array y un tamaño de página, y devuelva el "chunk" correspondiente.
53. **Diferencia de Objetos:** Compara dos objetos y devuelve un objeto que solo contenga las llaves que cambiaron.
54. **GroupBy personalizado:** Implementa una función groupBy(array, callback).
55. **Memoize simple:** Crea una función que guarde en caché el resultado de una operación costosa con objetos.

FASE 4: Mentalidad Senior (5 Desafíos de Arquitectura)
No solo que funcione, sino que sea excelente.

56. **Refactorización de Anidamiento:** Toma un código lleno de if/else anidados recorriendo arrays y conviértelo en una cadena de métodos funcionales legibles.
57. **Optimización de Búsqueda:** Tienes dos listas de 5,000 elementos cada una. Encuentra los elementos comunes. (Evita el bucle anidado $O(n^2)$, usa un Map o Set para $O(n)$).
58. **Inmutabilidad Extrema:** Realiza una operación compleja de actualización en un estado anidado (3 niveles) asegurando que **ningún** objeto intermedio sea mutado (usa Spread Operator correctamente).
59. **Analizador de Logs:** Procesa un array de 100,000 líneas de logs para obtener estadísticas (errores por hora, IP más frecuente) en una sola pasada de reduce.
60. **Desacoplamiento:** Crea una función de transformación de datos que sea "pura" (no dependa de variables externas) y fácil de testear.

Un consejo para tu entrenamiento:
Cuando llegues a la Fase 2, intenta **visualizar** los datos antes y después.