

Reporte de Evaluación - Fork de GitHub

Información General

Estudiante: SergioAndrés Montoya Monsalve
Repositorio: SerelbochaD3773/act_web1_s7
Fecha de evaluación: 5/10/2025, 12:31:38
Evaluado por: Sistema de Evaluación Masiva

Resumen de Calificaciones

Calificación general: 4.7/5.0
Actividades completadas: 10/10
Porcentaje de completitud: 100.0%

Detalle de Actividades

#	Descripción	Archivo	Encontrado	Calificación
1	Gestión de Inventario Básico - Crea un a...	src/ejercicio_01.js	Sí	5.0
2	Filtrado de Productos por Categoría - Us...	src/ejercicio_02.js	Sí	5.0
3	Transformación de Datos con map() - Crea...	src/ejercicio_03.js	Sí	5.0
4	Análisis de Ventas con reduce() - Dado u...	src/ejercicio_04.js	Sí	4.0
5	Búsqueda y Verificación - Crea un array ...	src/ejercicio_05.js	Sí	5.0
6	Manipulación de Arrays - Crea un array i...	src/ejercicio_06.js	Sí	5.0
7	Ordenamiento y Reversión - Crea arrays d...	src/ejercicio_07.js	Sí	5.0
8	Desestructuración de Arrays - Dado el ar...	src/ejercicio_08.js	Sí	4.0
9	Desestructuración de Objetos - Crea un o...	src/ejercicio_09.js	Sí	4.0
10	Métodos de Objeto - Crea un objeto y dem...	src/ejercicio_10.js	Sí	5.0

Retroalimentación Detallada

Actividad 1: Gestión de Inventario Básico - Crea un array de objetos que represente un inventario de productos. Cada producto debe tener: id, nombre, precio, categoria, stock. Declara al menos 5 productos y muestra todos los productos, total de productos en inventario y valor total del inventario.

Archivo esperado: src/ejercicio_01.js
Estado: Archivo encontrado
Calificación: 5.0/5.0
Retroalimentación:
Excelente solución. El código es claro, conciso y cumple con todos los requisitos de la actividad. Bien hecho.

Actividad 2: Filtrado de Productos por Categoría - Usando el inventario del ejercicio anterior, utiliza el método filter() para mostrar solo productos de la categoría 'Electrónicos', productos con stock menor a 10 y productos con precio mayor a \$500.

Archivo esperado: src/ejercicio_02.js
Estado: Archivo encontrado
Calificación: 5.0/5.0
Retroalimentación:
La solución es correcta y clara, cumpliendo todos los requisitos. El código es legible y bien estructurado, utilizando 'filter' de manera eficiente.

Actividad 3: Transformación de Datos con map() - Crea un array de estudiantes con nombre, edad, notas (array de números). Usa map() para crear un nuevo array con solo los nombres, crear un array con el promedio de cada estudiante y agregar una propiedad estado ('Aprobado' si promedio ≥ 70 , 'Reprobado' si < 70).

Archivo esperado: src/ejercicio_03.js

Estado: Archivo encontrado

Calificación: 5.0/5.0

Retroalimentación:

Excelente solución. El código es claro, conciso y utiliza `map` y `reduce` de forma correcta para cumplir con todos los requerimientos. La lógica para calcular promedios y asignar estados es impecable.

Actividad 4: Análisis de Ventas con reduce() - Dado un array de ventas con producto, cantidad, precio, fecha. Usa reduce() para calcular total de ingresos, producto más vendido (por cantidad) y promedio de venta por transacción.

Archivo esperado: src/ejercicio_04.js

Estado: Archivo encontrado

Calificación: 4.0/5.0

Retroalimentación:

La solución calcula correctamente el total de ingresos y encuentra el producto más vendido. Sin embargo, el cálculo del promedio por transacción es incorrecto y el uso de `filter` para obtener el producto más vendido es redundante. Podrías simplificar la lógica para el producto más vendido y corregir el promedio.

Actividad 5: Búsqueda y Verificación - Crea un array de usuarios con id, nombre, email, activo. Implementa búsquedas usando find() para buscar usuario por email, findIndex() para obtener posición de usuario por id, some() para verificar si hay usuarios inactivos y every() para verificar si todos tienen email válido (contiene @).

Archivo esperado: src/ejercicio_05.js

Estado: Archivo encontrado

Calificación: 5.0/5.0

Retroalimentación:

La solución es correcta, clara y concisa. Utiliza las funciones `find`, `findIndex`, `some` y `every` de forma eficiente para cumplir con los requisitos del ejercicio. ¡Excelente trabajo!

Actividad 6: Manipulación de Arrays - Crea un array inicial [1, 2, 3, 4, 5] y demuestra push() y pop() (agregar y quitar del final), shift() y unshift() (agregar y quitar del inicio), splice() (insertar elementos en posición específica) y slice() (extraer porción sin modificar original).

Archivo esperado: src/ejercicio_06.js

Estado: Archivo encontrado

Calificación: 5.0/5.0

Retroalimentación:

La solución es correcta y completa. Demuestra el uso adecuado de todos los métodos de array solicitados y el código es legible.

Actividad 7: Ordenamiento y Reversión - Crea arrays de números desordenados (ordena ascendente y descendente), nombres de personas (ordena alfabéticamente), objetos con propiedad edad (ordena por edad) y usa reverse() para invertir el orden.

Archivo esperado: src/ejercicio_07.js

Estado: Archivo encontrado

Calificación: 5.0/5.0

Retroalimentación:

La solución es correcta y completa, cumpliendo con todos los requisitos. El código es legible y bien estructurado, demostrando comprensión de los métodos de ordenamiento y reversión.

Actividad 8: Desestructuración de Arrays - Dado el array ['JavaScript', 'Python', 'Java', 'C++', 'Go']: extrae los primeros 3 lenguajes, extrae el primero y el último, usa rest operator para separar el primero del resto e intercambia dos variables usando desestructuración.

Archivo esperado: src/ejercicio_08.js

Estado: Archivo encontrado

Calificación: 4.0/5.0

Retroalimentación:

La solución es correcta y funcional. Podrías usar desestructuración para extraer el último elemento directamente y evitar el acceso por índice. Además, documentar mejor el intercambio con desestructuración sería útil.

Actividad 9: Desestructuración de Objetos - Crea un objeto persona con propiedades anidadas (dirección, contacto). Demuestra desestructuración básica, renombrado de variables, valores por defecto, desestructuración anidada y rest operator en objetos.

Archivo esperado: src/ejercicio_09.js

Estado: Archivo encontrado

Calificación: 4.0/5.0

Retroalimentación:

La solución es correcta y demuestra un buen entendimiento de la desestructuración. Podrías mejorar la consistencia en la desestructuración (ej: la ciudad debería desestructurarse directamente de persona y no de persona.direccion, que no existe). Considera también el uso de template literals para imprimir los resultados.

Actividad 10: Métodos de Objeto - Crea un objeto y demuestra Object.keys() (obtener claves), Object.values() (obtener valores), Object.entries() (obtener pares clave-valor) e iterar sobre el objeto con forEach().

Archivo esperado: src/ejercicio_10.js

Estado: Archivo encontrado

Calificación: 5.0/5.0

Retroalimentación:

Excelente solución. El código es claro, conciso y demuestra el uso correcto de Object.keys(), Object.values(), Object.entries() y la iteración con forEach().

Resumen General

Excelente trabajo. Completó 10/10 actividades (100%) con una calificación promedio de 4.7/5. Demuestra buen dominio de los conceptos.

Recomendaciones

- Continuar con el excelente trabajo y mantener la calidad del código