Reporte de Evaluación - Fork de GitHub

Información General

Estudiante: Esteban Garces Alzate
Repositorio: EstebanGarcesA/act_web1_s7
Fecha de evaluación: 11/9/2025, 20:28:22
Evaluado por: Sistema (Re-evaluación individual)

Resumen de Calificaciones

Calificación general: 4.3/5.0
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	3.0
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	4.0
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	4.0
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: 3.0/5.0

Retroalimentación:

La solución cumple con los requisitos básicos de la actividad, pero la llamada a la función `valorTotalInventario()` se imprime como texto en lugar de ejecutar la función e imprimir su resultado. Se podría mejorar la claridad incluyendo la impresión del valor total del inventario con un mensaje más descriptivo.

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 cumple con todos los requisitos. El código es legible y fácil de entender. Buen trabajo.

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: 4.0/5.0

Retroalimentación:

La solución es correcta y funcional. Se podría mejorar la legibilidad y eficiencia combinando los dos últimos `map` en uno solo para calcular el promedio y el estado simultáneamente, evitando iteraciones innecesarias.

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 es correcta y utiliza `reduce` de forma adecuada para calcular los valores solicitados. Se podría mejorar la legibilidad de `masVendido` inicializando `max` con la primera venta en el array y utilizar destructuring para simplificar el cálculo del 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: 4.0/5.0

Retroalimentación:

La solución es correcta y funcional. Sin embargo, la función `obtenerPosicion` podría tener un nombre de parámetro más descriptivo, y es preferible usar `const` para `verificarInactivos` ya que no se reasigna.

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, clara y concisa. Demuestra el uso correcto de todos los métodos del array solicitados. Excelente trabajo.

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, demostrando el uso adecuado de `sort()` y `reverse()` para diferentes tipos de datos. El código es claro y conciso.

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. Se podría mejorar la extracción del primer y último elemento usando desestructuración en lugar de acceder por índice y length.

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 actividad cumple con la desestructuración de objetos, incluyendo anidada y rest operator. Sin embargo, hay un `console.log` con valores hardcodeados (`console.log('manu', 30);`) que no corresponde y la variable `trabajo` no existe en el objeto `persona`, lo cual demuestra el valor por defecto, pero podría ser más claro añadiéndola al objeto persona con valor `null` o similar.

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. Demuestra el uso correcto de `Object.keys()`, `Object.values()`, `Object.entries()` e iteración con `forEach()`. El código es limpio y fácil de entender.

Resumen General

Excelente trabajo. Completó 10/10 actividades (100%) con una calificación promedio de 4.3/5. Demuestra buen dominio de los conceptos.

Recomendaciones

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