

# Reporte de Evaluación - Fork de GitHub

## Información General

Estudiante: Julio Eduardo Correa  
Repositorio: Jcorrea24/act\_web1\_s4  
Fecha de evaluación: 21/8/2025, 16:38:43  
Evaluado por: Sistema de Evaluación

## Resumen de Calificaciones

Calificación general: 4.3/5.0  
Actividades completadas: 18/20  
Porcentaje de completitud: 90.0%

## Detalle de Actividades

#	Descripción	Archivo	Encontrado	Calificación
1	Calcular el cuadrado de un número - Crea...	ejercicios/ejercicio_01.js	Sí	5.0
2	Saludar con nombre por defecto - Crea un...	ejercicios/ejercicio_02.js	Sí	5.0
3	Sumar elementos de un arreglo - Crea una...	ejercicios/ejercicio_03.js	Sí	3.0
4	Ejecutar un mensaje instantáneamente - C...	ejercicios/ejercicio_04.js	Sí	5.0
5	Contar vocales en una cadena - Crea una ...	ejercicios/ejercicio_05.js	Sí	5.0
6	Filtrar números mayores a 10 - Crea una ...	ejercicios/ejercicio_06.js	Sí	0.0
7	Convertir a mayúsculas - Crea una funció...	ejercicios/ejercicio_07.js	Sí	0.0
8	Temporizador con mensaje - Crea una func...	ejercicios/ejercicio_08.js	Sí	5.0
9	Crear un contador privado - Crea una fun...	ejercicios/ejercicio_09.js	Sí	5.0
10	Calcular factorial - Crea una función qu...	ejercicios/ejercicio_10.js	Sí	5.0
11	Verificar si un número es par - Crea una...	ejercicios/ejercicio_11.js	Sí	4.0
12	Multiplicar elementos de un arreglo - Cr...	ejercicios/ejercicio_12.js	Sí	5.0
13	Reemplazar espacios por guiones - Crea u...	ejercicios/ejercicio_13.js	Sí	5.0
14	Generar un ID único - Crea una función q...	ejercicios/ejercicio_14.js	Sí	4.0
15	Invertir una cadena - Crea una función q...	ejercicios/ejercicio_15.js	Sí	5.0
16	Sumar argumentos variables - Crea una fu...	ejercicios/ejercicio_16.js	Sí	5.0
17	Ejecutar operación personalizada - Crea ...	ejercicios/ejercicio_17.js	Sí	5.0
18	Validar correo electrónico - Crea una fu...	ejercicios/ejercicio_18.js	Sí	4.0
19	Retrasar ejecución de un mensaje - Crea ...	ejercicios/ejercicio_19.js	Sí	5.0
20	Calcular promedio de un arreglo - Crea u...	ejercicios/ejercicio_20.js	Sí	5.0

## Retroalimentación Detallada

### Actividad 1: Calcular el cuadrado de un número - Crea una función que reciba un número y devuelva su cuadrado. (Tipo de función: Declaración de función)

Archivo esperado: ejercicios/ejercicio\_01.js

Estado: Archivo encontrado

Calificación: 5.0/5.0

Retroalimentación:

La solución es correcta y eficiente. El código es claro y fácil de entender. La función cumple con los requisitos de la actividad.

### Actividad 2: Saludar con nombre por defecto - Crea una función que salude a una persona por su nombre. Si no se proporciona un nombre, usa 'Invitado'. (Tipo de función: Expresión de función)

Archivo esperado: ejercicios/ejercicio\_02.js

Estado: Archivo encontrado

Calificación: 5.0/5.0

Retroalimentación:

Excelente solución. El código es conciso, correcto y utiliza la sintaxis de valor por defecto de manera efectiva. Cumple con todos los requisitos de la actividad.

### Actividad 3: Sumar elementos de un arreglo - Crea una función que sume todos los números de un arreglo. (Tipo de función: Función flecha)

Archivo esperado: ejercicios/ejercicio\_03.js

Estado: Archivo encontrado

Calificación: 3.0/5.0

Retroalimentación:

La función solo suma los primeros tres elementos del arreglo y asume una longitud fija. Debería iterar sobre el arreglo completo para ser una solución general.

### Actividad 4: Ejecutar un mensaje instantáneamente - Crea una función que imprima '¡Bienvenido!' en la consola al definirse. (Tipo de función: IIFE)

Archivo esperado: ejercicios/ejercicio\_04.js

Estado: Archivo encontrado

Calificación: 5.0/5.0

Retroalimentación:

Solución correcta y concisa. Cumple con el uso de una IIFE para imprimir el mensaje al definirse.

### Actividad 5: Contar vocales en una cadena - Crea una función que cuente las vocales (a, e, i, o, u) en una cadena. (Tipo de función: Función recursiva)

Archivo esperado: ejercicios/ejercicio\_05.js

Estado: Archivo encontrado

Calificación: 5.0/5.0

Retroalimentación:

Excelente solución. La función recursiva es concisa y eficiente. Bien hecho.

### Actividad 6: Filtrar números mayores a 10 - Crea una función que reciba un arreglo y devuelva solo los números mayores a 10. (Tipo de función: Función de orden superior)

Archivo esperado: ejercicios/ejercicio\_06.js

Estado: Archivo encontrado

Calificación: 0.0/5.0

Retroalimentación:

La solución proporcionada no resuelve el problema planteado. El código dado convierte una cadena a mayúsculas en lugar de filtrar números mayores a 10 de un arreglo. Necesitas implementar la lógica de filtrado correctamente.

**Actividad 7: Convertir a mayúsculas - Crea una función que convierta una cadena a mayúsculas. (Tipo de función: Función flecha)**

Archivo esperado: ejercicios/ejercicio\_07.js

Estado: Archivo encontrado

Calificación: 0.0/5.0

Retroalimentación:

El código no resuelve el problema planteado, simplemente establece un temporizador. Falta la lógica para convertir una cadena a mayúsculas usando una función flecha.

**Actividad 8: Temporizador con mensaje - Crea una función que imprima un mensaje después de 3 segundos usando setTimeout. (Tipo de función: Función anónima)**

Archivo esperado: ejercicios/ejercicio\_08.js

Estado: Archivo encontrado

Calificación: 5.0/5.0

Retroalimentación:

La solución es correcta y concisa. Utiliza `setTimeout` con una función anónima para imprimir el mensaje después de 3 segundos, cumpliendo con los requisitos del ejercicio.

**Actividad 9: Crear un contador privado - Crea una función que devuelva otra función para contar incrementos, manteniendo el contador privado. (Tipo de función: IIFE con closure)**

Archivo esperado: ejercicios/ejercicio\_09.js

Estado: Archivo encontrado

Calificación: 5.0/5.0

Retroalimentación:

Excelente solución. El código implementa correctamente el closure para mantener el contador privado y funciona como se espera. La estructura es clara y concisa.

**Actividad 10: Calcular factorial - Crea una función que calcule el factorial de un número. (Tipo de función: Función recursiva)**

Archivo esperado: ejercicios/ejercicio\_10.js

Estado: Archivo encontrado

Calificación: 5.0/5.0

Retroalimentación:

La solución es correcta y utiliza recursión de manera eficiente para calcular el factorial. El código es limpio y funcional.

**Actividad 11: Verificar si un número es par - Crea una función que determine si un número es par. (Tipo de función: Declaración de función)**

Archivo esperado: ejercicios/ejercicio\_11.js

Estado: Archivo encontrado

Calificación: 4.0/5.0

Retroalimentación:

La función es correcta y funcional. Se podría mejorar retornando valores booleanos (true/false) en lugar de strings para mayor flexibilidad y consistencia en el tipo de dato de retorno. Considera eliminar el console.log del archivo de la función, ya que se enfoca en la lógica principal.

**Actividad 12: Multiplicar elementos de un arreglo - Crea una función que multiplique todos los números de un arreglo. (Tipo de función: Función de orden superior)**

Archivo esperado: ejercicios/ejercicio\_12.js

Estado: Archivo encontrado

Calificación: 5.0/5.0

Retroalimentación:

Solución correcta y concisa utilizando `reduce`. El código es limpio y funcional, cumpliendo con los requisitos de la actividad.

**Actividad 13: Reemplazar espacios por guiones - Crea una función que reemplace los espacios en una cadena por guiones. (Tipo de función: Expresión de función)**

Archivo esperado: ejercicios/ejercicio\_13.js

Estado: Archivo encontrado

Calificación: 5.0/5.0

Retroalimentación:

La solución es correcta y concisa, utilizando el método `replace` con una expresión regular para reemplazar todos los espacios. El código es legible y cumple con el objetivo de la actividad.

**Actividad 14: Generar un ID único - Crea una función que genere un ID único basado en un contador interno. (Tipo de función: IIFE con closure)**

Archivo esperado: ejercicios/ejercicio\_14.js

Estado: Archivo encontrado

Calificación: 4.0/5.0

Retroalimentación:

La solución es correcta y utiliza una IIFE con closure como se solicitó. Sin embargo, el ID inicial es 5, cuando debería empezar en 1. Se puede mejorar la legibilidad añadiendo comentarios.

**Actividad 15: Invertir una cadena - Crea una función que invierta una cadena de texto. (Tipo de función: Función recursiva)**

Archivo esperado: ejercicios/ejercicio\_15.js

Estado: Archivo encontrado

Calificación: 5.0/5.0

Retroalimentación:

La solución es correcta y eficiente. Utiliza recursión de manera concisa para invertir la cadena.

**Actividad 16: Sumar argumentos variables - Crea una función que sume un número variable de argumentos. (Tipo de función: Función flecha con parámetro rest)**

Archivo esperado: ejercicios/ejercicio\_16.js

Estado: Archivo encontrado

Calificación: 5.0/5.0

Retroalimentación:

Excelente solución. Utiliza correctamente el parámetro rest y reduce para sumar los argumentos de forma concisa y eficiente.

**Actividad 17: Ejecutar operación personalizada - Crea una función que reciba dos números y una función callback para realizar una operación. (Tipo de función: Función de orden superior)**

Archivo esperado: ejercicios/ejercicio\_17.js

Estado: Archivo encontrado

Calificación: 5.0/5.0

Retroalimentación:

La solución es concisa y correcta. Cumple con los requisitos de la actividad de forma eficiente.

**Actividad 18: Validar correo electrónico - Crea una función que valide si una cadena es un correo electrónico básico (contiene @ y .com). (Tipo de función: Declaración de función)**

Archivo esperado: ejercicios/ejercicio\_18.js

Estado: Archivo encontrado

Calificación: 4.0/5.0

Retroalimentación:

La función cumple con la validación básica requerida. Sin embargo, la validación de correos electrónicos es más compleja y esta función no cubre todos los casos válidos. Considera usar expresiones regulares para una validación más robusta.

**Actividad 19: Retrasar ejecución de un mensaje - Crea una función que imprima un mensaje después de un tiempo definido por el usuario. (Tipo de función: Función anónima con setTimeout)**

Archivo esperado: ejercicios/ejercicio\_19.js

Estado: Archivo encontrado

Calificación: 5.0/5.0

Retroalimentación:

La solución es correcta y concisa. Utiliza una función anónima con `setTimeout` para cumplir con los requisitos. Buen trabajo.

**Actividad 20: Calcular promedio de un arreglo - Crea una función que calcule el promedio de un arreglo de números. (Tipo de función: Función flecha)**

Archivo esperado: ejercicios/ejercicio\_20.js

Estado: Archivo encontrado

Calificación: 5.0/5.0

Retroalimentación:

Excelente solución. El código es conciso, utiliza la función flecha y ``reduce`` de manera eficiente para calcular el promedio. La prueba en ``console.log`` es adecuada.

## Resumen General

Excelente trabajo. Completó 18/20 actividades (90%) con una calificación promedio de 4.3/5. Demuestra buen dominio de los conceptos.

## Recomendaciones

- Revisar y mejorar las actividades con calificación baja