

Reporte de Evaluación - Fork de GitHub

Información General

Estudiante: Mario Alonso Munera Velez
Repositorio: MarioMunera1993/act_web1_s4
Fecha de evaluación: 21/8/2025, 16:22:30
Evaluado por: Sistema de Evaluación

Resumen de Calificaciones

Calificación general: 4.5/5.0
Actividades completadas: 19/20
Porcentaje de completitud: 95.0%

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	5.0
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	5.0
7	Convertir a mayúsculas - Crea una funció...	ejercicios/ejercicio_07.js	Sí	5.0
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	No	0.0
11	Verificar si un número es par - Crea una...	ejercicios/ejercicio_11.js	Sí	3.0
12	Multiplicar elementos de un arreglo - Cr...	ejercicios/ejercicio_12.js	Sí	4.0
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	5.0
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	3.0
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 función calcula correctamente el cuadrado de un número. El código es limpio y fácil de entender.

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:

La solución es correcta y concisa. El código es legible y cumple con 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: 5.0/5.0

Retroalimentación:

La solución es correcta y eficiente. Utiliza una función flecha y un bucle for para sumar los elementos del arreglo, cumpliendo con los requisitos. El código es legible y bien estructurado.

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:

Excelente solución. El código es conciso y cumple perfectamente con la definición de una IIFE para imprimir el mensaje solicitado.

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:

La solución es correcta y eficiente al usar recursión. El código es limpio y fácil de entender, cumpliendo con el objetivo de la actividad. Buena implementación.

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: 5.0/5.0

Retroalimentación:

La solución es correcta y concisa. Utiliza la función `filter` de manera eficiente para cumplir con el objetivo.

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: 5.0/5.0

Retroalimentación:

La solución es correcta y concisa, utilizando una función flecha de manera eficiente para convertir la cadena a mayúsculas. El código es limpio y fácil de entender.

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 correctamente `setTimeout` con una función anónima y cumple con la descripción de la actividad. Buen trabajo.

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:

La solución es correcta, implementa una IIFE con closure para mantener el contador privado. El código es conciso y funcional.

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 no encontrado

Calificación: 0.0/5.0

Retroalimentación:

Error al evaluar: got status: 503 . {"error":{"code":503,"message":"The model is overloaded. Please try again later."},"status":"UNAVAILABLE"}}

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: 3.0/5.0

Retroalimentación:

La función no retorna un valor booleano (true/false) como debería. Además, imprime 'Es Impar' siempre, independientemente de si el número es par o no. Falta un `else` en la condicional.

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: 4.0/5.0

Retroalimentación:

La solución es correcta y utiliza `reduce` adecuadamente. Considera añadir validación para asegurar que la entrada sean números y mejorar la robustez del código ante entradas incorrectas.

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. Utiliza la función `replace` con una expresión regular para reemplazar todos los espacios, demostrando buen conocimiento del lenguaje.

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: 5.0/5.0

Retroalimentación:

Excelente solución. Utiliza correctamente una IIFE y un closure para mantener el estado del contador, generando IDs únicos como se esperaba. El código es limpio y funcional.

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 utiliza recursión de manera eficiente para invertir la cadena. El código es claro y conciso.

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. El código es conciso, claro y utiliza correctamente la función flecha con el parámetro rest y reduce para sumar los argumentos. Cumple con todos los requisitos de la actividad.

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:

Excelente solución. El código es conciso, claro y funcional, demostrando una correcta comprensión de las funciones de orden superior y callbacks en JavaScript.

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: 3.0/5.0

Retroalimentación:

La función cumple con los requisitos básicos, pero es demasiado simplista. No valida la estructura completa de un correo electrónico, solo la presencia de '@' y que termine en '.com'. Se recomienda usar una expresión regular 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:

Excelente solución. El código es conciso, legible y resuelve el problema planteado usando una función anónima con `setTimeout` correctamente.

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:

La solución es concisa, correcta y utiliza una función flecha como se solicitó. El código es legible y aplica buenas prácticas al usar `reduce` para calcular la suma.

Resumen General

Excelente trabajo. Completó 19/20 actividades (95%) con una calificación promedio de 4.5/5. Demuestra buen dominio de los conceptos.

Recomendaciones

- Completar los archivos faltantes: ejercicios/ejercicio_10.js