Reporte de Evaluación - Fork de GitHub

Información General

Estudiante: paula andrea gil vargas
Repositorio: GGP113/act_web1_s4
Fecha de evaluación: 21/8/2025, 16:53:39
Evaluado por: Sistema de Evaluación

Resumen de Calificaciones

Calificación general: 4.1/5.0
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	3.0
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	4.0
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	4.0
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	4.0
13	Reemplazar espacios por guiones - Crea u...	ejercicios/ejercicio_13.js	Sí	3.0
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	4.0
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	4.0
20	Calcular promedio de un arreglo - Crea u...	ejercicios/ejercicio_20.js	Sí	4.0

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: 3.0/5.0

Retroalimentación:

La función calcula correctamente el cuadrado. Sin embargo, la función debería retornar el resultado en lugar de imprimirlo directamente en la consola. Falta la sentencia `return`.

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. Utiliza el parámetro por defecto de manera adecuada y la estructura del código es limpia.

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. El código es limpio y fácil de entender, cumpliendo con los requisitos de la actividad.

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:

La solución es correcta y utiliza una IIFE para imprimir el mensaje al definirse. El código es limpio y cumple con la descripción de la actividad.

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: 4.0/5.0

Retroalimentación:

La solución recursiva es correcta y concisa. El código es legible y funciona como se espera. Una alternativa iterativa podría ser más eficiente para cadenas largas.

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:

Error al procesar la evaluación

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: 4.0/5.0

Retroalimentación:

La función cumple el objetivo. Sin embargo, en lugar de imprimir el resultado directamente en la función, sería mejor que la función retornara la cadena en mayúsculas y luego imprimirla en la llamada a la función. Esto permite mayor flexibilidad y reutilización del código.

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, utilizando una función anónima autoejecutable y `setTimeout` para cumplir con los requisitos. El código es limpio y fácil de entender.

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 y concisa. Utiliza una IIFE con closure para mantener el contador privado, cumpliendo con los requisitos de la actividad.

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 eficiente, implementando la función factorial de manera recursiva como se solicitó. El código es limpio y fácil de entender.

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 resuelve correctamente el problema. Sin embargo, la instrucción `console.log` dentro de la función no es necesaria; la función debe limitarse a retornar el valor booleano. Considera eliminarla para mejorar la claridad del código.

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 función resuelve correctamente el problema. Se puede mejorar la legibilidad y eficiencia utilizando `reduce` para una solución más concisa y funcional, evitando la mutación de `numUno` dentro del bucle.

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: 3.0/5.0

Retroalimentación:

La función reemplaza los espacios por guiones, pero imprime el resultado en lugar de retornarlo, lo que limita su reutilización. Además, la lógica del bucle podría simplificarse usando `join('-')`.

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. La función IIFE con closure implementa correctamente la generación de IDs únicos y mantiene el estado interno del contador. El código es limpio y fácil de entender.

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: 4.0/5.0

Retroalimentación:

La función recursiva invierte correctamente la cadena. El código es funcional y cumple con el objetivo, aunque podría simplificarse evitando el uso de tantos parámetros en la función recursiva. Considera usar el índice directamente en lugar de contadores adicionales.

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 claro, conciso y cumple con todos los requisitos de la actividad, utilizando correctamente la función flecha y el parámetro rest.

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 correcta y concisa. El código es limpio y cumple con los requisitos de la actividad. Excelente trabajo.

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 la validación básica, pero podría mejorarse la lógica de verificación y evitar el uso de `console.log` dentro de la función. Considera usar métodos de string como `includes` o 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: 4.0/5.0

Retroalimentación:

La solución es correcta y utiliza una función anónima con setTimeout como se pedía. Sin embargo, la función retrasaMensaje no necesita retornar nada (undefined), y el console.log final imprime ese 'undefined'. Considera eliminar ese console.log final.

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: 4.0/5.0

Retroalimentación:

La solución es correcta y funcional. Se puede mejorar la legibilidad evitando redeclarar la variable 'promedio' dentro de la función. Usar 'const' para la variable 'suma' podría ser una buena práctica también.

Resumen General

Excelente trabajo. Completó 19/20 actividades (95%) con una calificación promedio de 4.1/5. Demuestra buen dominio de los conceptos.

Recomendaciones

- Revisar y mejorar las actividades con calificación baja