Reporte de Evaluación - Fork de GitHub

Información General

Estudiante: Estudiante desconocido
Repositorio: Estibenmanco31/act_web1_s3
Fecha de evaluación: 21/8/2025, 14:11:34
Evaluado por: Sistema de Evaluación Masiva

Resumen de Calificaciones

Calificación general: 3.8/5.0
Actividades completadas: 20/20
Porcentaje de completitud: 100.0%

Detalle de Actividades

#	Descripción	Archivo	Encontrado	Calificación
1	Crea un programa que use un ciclo while ...	ejercicios-while/ejercicio_while_01.js	Sí	1.0
2	Utiliza un ciclo while para calcular la ...	ejercicios-while/ejercicio_while_02.js	Sí	5.0
3	Implementa un sistema que use un ciclo w...	ejercicios-while/ejercicio_while_03.js	Sí	2.0
4	Crea un programa que use un ciclo while ...	ejercicios-while/ejercicio_while_04.js	Sí	3.0
5	Desarrolla una función que use un ciclo ...	ejercicios-while/ejercicio_while_05.js	Sí	4.0
6	Dado un array de nombres, usa un ciclo w...	ejercicios-while/ejercicio_while_06.js	Sí	4.0
7	Implementa un programa que use un ciclo ...	ejercicios-while/ejercicio_while_07.js	Sí	5.0
8	Utiliza un ciclo while para generar los ...	ejercicios-while/ejercicio_while_08.js	Sí	4.0
9	Crea un juego que use un ciclo while don...	ejercicios-while/ejercicio_while_09.js	Sí	4.0
10	Desarrolla un programa que use un ciclo ...	ejercicios-while/ejercicio_while_10.js	Sí	4.0
11	Usa un ciclo for anidado para crear las ...	ejercicios-for/ejercicio_for_01.js	Sí	5.0
12	Implementa un programa que use ciclos fo...	ejercicios-for/ejercicio_for_02.js	Sí	3.0
13	Dado un array de 20 números aleatorios, ...	ejercicios-for/ejercicio_for_03.js	Sí	4.0
14	Utiliza ciclos for anidados para encontr...	ejercicios-for/ejercicio_for_04.js	Sí	5.0
15	Crea un programa que use un ciclo for pa...	ejercicios-for/ejercicio_for_05.js	Sí	5.0
16	Implementa una función que use un ciclo ...	ejercicios-for/ejercicio_for_06.js	Sí	4.0
17	Usa ciclos for anidados para crear una m...	ejercicios-for/ejercicio_for_07.js	Sí	5.0
18	Desarrolla un programa que use un ciclo ...	ejercicios-for/ejercicio_for_08.js	Sí	2.0
19	Utiliza un ciclo for para generar la sig...	ejercicios-for/ejercicio_for_09.js	Sí	1.0
20	Implementa un programa que use un ciclo ...	ejercicios-for/ejercicio_for_10.js	Sí	5.0

Retroalimentación Detallada

Actividad 1: Crea un programa que use un ciclo while para mostrar una cuenta regresiva desde 20 hasta 1, mostrando cada número en la consola. Al final debe mostrar '¡Despegue!'.

Archivo esperado: ejercicios-while/ejercicio_while_01.js

Estado: Archivo encontrado

Calificación: 1.0/5.0

Retroalimentación:

El código no resuelve el problema planteado en la descripción. Implementa un programa que encuentra números divisibles por 3 y 5, no una cuenta regresiva desde 20 hasta 1. Revisa el enunciado y corrige el código.

Actividad 2: Utiliza un ciclo while para calcular la suma de todos los números pares entre 1 y 50. Muestra el resultado final y cuántos números pares se sumaron.

Archivo esperado: ejercicios-while/ejercicio_while_02.js

Estado: Archivo encontrado

Calificación: 5.0/5.0

Retroalimentación:

La solución es correcta y eficiente. El código es limpio, fácil de entender y cumple con todos los requisitos del ejercicio. Excelente trabajo.

Actividad 3: Implementa un sistema que use un ciclo while para solicitar una contraseña al usuario hasta que ingrese 'admin123'. Debe mostrar cuántos intentos fallidos hubo antes del acceso correcto.

Archivo esperado: ejercicios-while/ejercicio_while_03.js

Estado: Archivo encontrado

Calificación: 2.0/5.0

Retroalimentación:

El código incluye dos implementaciones distintas del mismo problema, una con `prompt` y otra con `readline`. La primera implementación no valida correctamente la contraseña contra el criterio especificado en el comentario inicial del ejercicio, y la segunda no cumple con el requerimiento de usar un ciclo `while` como se solicitó en la descripción de la actividad. Considera enfocarte en una única implementación que use un ciclo `while` y valide la contraseña 'admin123' o los criterios actualizados.

Actividad 4: Crea un programa que use un ciclo while para generar números aleatorios entre 1 y 100 hasta que salga un número mayor a 95. Muestra cada número generado y al final indica cuántos números se generaron.

Archivo esperado: ejercicios-while/ejercicio_while_04.js

Estado: Archivo encontrado

Calificación: 3.0/5.0

Retroalimentación:

El código no cumple con la condición de parada solicitada (número mayor a 95, no el 50). La estructura del código es buena, pero es necesario corregir la lógica del bucle para que se ajuste a la descripción del problema.

Actividad 5: Desarrolla una función que use un ciclo while para calcular el factorial de un número dado. Debe mostrar paso a paso cómo se va calculando el factorial.

Archivo esperado: ejercicios-while/ejercicio_while_05.js

Estado: Archivo encontrado

Calificación: 4.0/5.0

Retroalimentación:

La solución calcula correctamente el factorial usando un ciclo while. Podría mejorar mostrando el proceso de cálculo paso a paso en la consola, como lo indica la descripción de la actividad.

Actividad 6: Dado un array de nombres, usa un ciclo while para buscar un nombre específico. El programa debe mostrar en qué posición se encontró el nombre o indicar si no existe.

Archivo esperado: ejercicios-while/ejercicio_while_06.js

Estado: Archivo encontrado

Calificación: 4.0/5.0

Retroalimentación:

La solución es correcta y funcional. Podrías mejorar la legibilidad usando `const` o `let` de manera más consistente y usar tipos de datos numéricos en lugar de strings para el array de números.

Actividad 7: Implementa un programa que use un ciclo while para contar cuántos dígitos tiene un número entero positivo. Por ejemplo, 12345 tiene 5 dígitos.

Archivo esperado: ejercicios-while/ejercicio_while_07.js

Estado: Archivo encontrado

Calificación: 5.0/5.0

Retroalimentación:

Excelente solución. El código es claro, conciso y resuelve el problema correctamente, incluyendo validación de entrada y manejo del caso especial del cero.

Actividad 8: Utiliza un ciclo while para generar los primeros 15 números de la secuencia de Fibonacci. Muestra cada número de la secuencia.

Archivo esperado: ejercicios-while/ejercicio_while_08.js

Estado: Archivo encontrado

Calificación: 4.0/5.0

Retroalimentación:

El código funciona correctamente y genera la secuencia de Fibonacci. Se puede mejorar la legibilidad eliminando los condicionales `if` iniciales, ya que el cálculo se puede generalizar desde el inicio.

Actividad 9: Crea un juego que use un ciclo while donde el usuario debe adivinar un número entre 1 y 50. Proporciona pistas ('muy alto', 'muy bajo', 'cerca') y limita a 7 intentos máximo.

Archivo esperado: ejercicios-while/ejercicio_while_09.js

Estado: Archivo encontrado

Calificación: 4.0/5.0

Retroalimentación:

La lógica del juego es correcta y funcional. Se podría mejorar la validación de la entrada del usuario (ej: asegurarse de que sea un número) y generalizar el rango del número secreto para mayor flexibilidad.

Actividad 10: Desarrolla un programa que use un ciclo while para procesar calificaciones de estudiantes. Debe continuar pidiendo calificaciones hasta que se ingrese -1, luego calcular y mostrar el promedio, la calificación más alta y más baja.

Archivo esperado: ejercicios-while/ejercicio_while_10.js

Estado: Archivo encontrado

Calificación: 4.0/5.0

Retroalimentación:

La solución es correcta y bien estructurada. Se podrían agregar validaciones más robustas para la entrada, pero en general cumple con los requisitos de la actividad y presenta un código legible.

Actividad 11: Usa un ciclo for anidado para crear las tablas de multiplicar del 1 al 10. Cada tabla debe mostrarse claramente separada y formateada.

Archivo esperado: ejercicios-for/ejercicio_for_01.js

Estado: Archivo encontrado

Calificación: 5.0/5.0

Retroalimentación:

La solución es correcta y cumple con los requerimientos. El código es legible y bien estructurado, generando las tablas de multiplicar del 1 al 12 como se solicitó.

Actividad 12: Implementa un programa que use ciclos for anidados para crear un patrón de asteriscos en forma de diamante: *, **, *, ****, *****, ****, ***, **, ***

Archivo esperado: ejercicios-for/ejercicio_for_02.js

Estado: Archivo encontrado

Calificación: 3.0/5.0

Retroalimentación:

El código genera una pirámide, no un diamante como se solicitaba. Debe modificarse para generar la mitad superior y luego la inferior invertida del diamante. La lógica para la pirámide es correcta, pero necesita adaptación.

Actividad 13: Dado un array de 20 números aleatorios, usa un ciclo for para encontrar: el número mayor, el menor, la suma total, el promedio, y cuántos números son pares e impares.

Archivo esperado: ejercicios-for/ejercicio_for_03.js

Estado: Archivo encontrado

Calificación: 4.0/5.0

Retroalimentación:

La solución es correcta y funcional. El código es legible, pero la impresión de información en cada iteración del bucle for podría optimizarse para no generar una salida tan extensa.

Actividad 14: Utiliza ciclos for anidados para encontrar y mostrar todos los números primos entre 1 y 100. Debe indicar cuántos números primos se encontraron.

Archivo esperado: ejercicios-for/ejercicio_for_04.js

Estado: Archivo encontrado

Calificación: 5.0/5.0

Retroalimentación:

Excelente solución. El código es claro, eficiente y cumple con todos los requisitos de la actividad. Bien hecho!

Actividad 15: Crea un programa que use un ciclo for para invertir una cadena de texto carácter por carácter. Muestra tanto la cadena original como la invertida.

Archivo esperado: ejercicios-for/ejercicio_for_05.js

Estado: Archivo encontrado

Calificación: 5.0/5.0

Retroalimentación:

La solución es correcta y bien estructurada. El código es legible y proporciona una buena explicación paso a paso del proceso de inversión. Excelente trabajo.

Actividad 16: Implementa una función que use un ciclo for para calcular la potencia de un número sin usar Math.pow(). Debe mostrar el proceso paso a paso.

Archivo esperado: ejercicios-for/ejercicio_for_06.js

Estado: Archivo encontrado

Calificación: 4.0/5.0

Retroalimentación:

La función calcula la potencia correctamente, incluyendo exponentes negativos y cero. La salida paso a paso es clara. Podrías mejorar la claridad en el manejo de exponentes negativos para que el último log muestre el valor correcto antes de la división final.

Actividad 17: Usa ciclos for anidados para crear una matriz 5x5 donde cada elemento sea la suma de sus índices (fila + columna). Muestra la matriz formateada.

Archivo esperado: ejercicios-for/ejercicio_for_07.js

Estado: Archivo encontrado

Calificación: 5.0/5.0

Retroalimentación:

La solución es correcta y bien estructurada. El código es legible y muestra ejemplos de cálculo que facilitan la comprensión del resultado, demostrando un entendimiento completo del problema.

Actividad 18: Desarrolla un programa que use un ciclo for para contar cuántas vocales (a, e, i, o, u) hay en una frase dada. Debe mostrar el conteo de cada vocal por separado.

Archivo esperado: ejercicios-for/ejercicio_for_08.js

Estado: Archivo encontrado

Calificación: 2.0/5.0

Retroalimentación:

El código no cumple con la descripción de la actividad. Cuenta la frecuencia de todos los caracteres en lugar de solo las vocales. Deberías enfocarte en implementar la lógica específica para contar vocales.

Actividad 19: Utiliza un ciclo for para generar la siguiente secuencia: 2, 4, 8, 16, 32, 64, 128, 256, 512, 1024. Debe mostrar cada término y la suma total de la secuencia.

Archivo esperado: ejercicios-for/ejercicio_for_09.js

Estado: Archivo encontrado

Calificación: 1.0/5.0

Retroalimentación:

El código no resuelve el problema planteado en la descripción. Implementa un simulador de dados en lugar de generar la secuencia numérica solicitada. El código, aunque funcional como simulador de dados, no cumple con los requisitos de la actividad.

Actividad 20: Implementa un programa que use un ciclo for para verificar si una palabra o frase es un palíndromo (se lee igual de izquierda a derecha que de derecha a izquierda). Ignora espacios y mayúsculas.

Archivo esperado: ejercicios-for/ejercicio_for_10.js

Estado: Archivo encontrado

Calificación: 5.0/5.0

Retroalimentación:

La solución es correcta y bien estructurada. El código es claro, funcional y aplica buenas prácticas al limpiar la entrada y manejar mayúsculas/minúsculas. Excelente implementación.

Resumen General

Buen trabajo general. Completó 20/20 actividades (100%) con una calificación promedio de 3.8/5. Hay oportunidades de mejora en algunos aspectos.

Recomendaciones

- Revisar y mejorar las actividades con calificación baja