Reporte de Evaluación - Fork de GitHub

Información General

Estudiante: Juan Miguel Santamaría Múnera
Repositorio: PeperoniSword/act_ntp_s3
Fecha de evaluación: 22/8/2025, 18:12:44
Evaluado por: Sistema de Evaluación

Resumen de Calificaciones

Calificación general: 3.9/5.0
Actividades completadas: 19/20
Porcentaje de completitud: 95.0%

Detalle de Actividades

#	Descripción	Archivo	Encontrado	Calificación
1	Usando un ciclo for, imprime los números...	src/ejercicio_01.py	Sí	3.0
2	Mediante un ciclo while, imprime los núm...	src/ejercicio_02.py	Sí	5.0
3	Con un ciclo for, calcula la suma de tod...	src/ejercicio_03.py	Sí	5.0
4	Utilizando un ciclo while, solicita al u...	src/ejercicio_04.py	Sí	3.0
5	Con un ciclo for, imprime la tabla de mu...	src/ejercicio_05.py	Sí	4.0
6	Mediante un ciclo while, genera y muestr...	src/ejercicio_06.py	Sí	5.0
7	Con un ciclo for, cuenta cuántas letras ...	src/ejercicio_07.py	Sí	5.0
8	Usando un ciclo while, calcula y muestra...	src/ejercicio_08.py	Sí	5.0
9	Con un ciclo for, imprime todos los núme...	src/ejercicio_09.py	Sí	4.0
10	Mediante un ciclo while, solicita al usu...	src/ejercicio_10.py	Sí	4.0
11	Con un ciclo for, imprime cada carácter ...	src/ejercicio_11.py	Sí	4.0
12	Utilizando un ciclo while, calcula el fa...	src/ejercicio_12.py	Sí	0.0
13	Con un ciclo for, imprime los números de...	src/ejercicio_13.py	Sí	5.0
14	Mediante un ciclo while, implementa un j...	src/ejercicio_14.py	Sí	4.0
15	Con un ciclo for, imprime un triángulo r...	src/ejercicio_15.py	Sí	4.0
16	Utilizando un ciclo while, simula un rel...	src/ejercicio_16.py	Sí	3.0
17	Con un ciclo for, solicita al usuario qu...	src/ejercicio_17.py	Sí	4.0
18	Mediante un ciclo while, genera y muestr...	src/ejercicio_18.py	Sí	3.0
19	Con un ciclo for, cuenta cuántas vocales...	src/ejercicio_19.py	Sí	4.0
20	Utilizando un ciclo while, solicita al u...	src/ejercicio_20.py	Sí	4.0

Retroalimentación Detallada

Actividad 1: Usando un ciclo for, imprime los números enteros del 0 al 9, cada uno en una línea.

Archivo esperado: src/ejercicio_01.py

Estado: Archivo encontrado

Calificación: 3.0/5.0

Retroalimentación:

La solución utiliza una comprensión de lista para imprimir, lo cual no es el uso más común para tareas de impresión secuencial. Además, empieza en 1 en vez de 0, por lo que no cumple completamente con la descripción de la actividad.

Actividad 2: Mediante un ciclo while, imprime los números enteros del 10 al 1 en orden descendente, cada número en una línea.

Archivo esperado: src/ejercicio_02.py

Estado: Archivo encontrado

Calificación: 5.0/5.0

Retroalimentación:

La solución es correcta y concisa. El código es legible y cumple con el objetivo de la actividad utilizando un ciclo while de manera eficiente.

Actividad 3: Con un ciclo for, calcula la suma de todos los enteros del 1 al 100 (inclusive) y muestra el resultado.

Archivo esperado: src/ejercicio_03.py

Estado: Archivo encontrado

Calificación: 5.0/5.0

Retroalimentación:

La solución es correcta y concisa. El código es legible y cumple con el objetivo de la actividad.

Actividad 4: Utilizando un ciclo while, solicita al usuario que ingrese números. El proceso termina cuando el usuario escriba 0. Al final, muestra la suma total de todos los números ingresados.

Archivo esperado: src/ejercicio_04.py

Estado: Archivo encontrado

Calificación: 3.0/5.0

Retroalimentación:

El código resuelve el problema, pero falta acumular la suma de los números ingresados. Necesitas una variable para guardar la suma y actualizarla en cada iteración del ciclo.

Actividad 5: Con un ciclo for, imprime la tabla de multiplicar del 7, es decir, 7 × 1, 7 × 2, …, 7 × 10, cada resultado en una línea.

Archivo esperado: src/ejercicio_05.py

Estado: Archivo encontrado

Calificación: 4.0/5.0

Retroalimentación:

La solución es correcta y concisa, utilizando una comprensión de lista para imprimir la tabla del 7. Considera usar un bucle `for` tradicional para mayor legibilidad en este caso específico, aunque la solución actual funciona.

Actividad 6: Mediante un ciclo while, genera y muestra los primeros 15 múltiplos de 3, comenzando desde 3.

Archivo esperado: src/ejercicio_06.py

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 el objetivo de la actividad.

Actividad 7: Con un ciclo for, cuenta cuántas letras 'a' (minúscula) hay en la cadena texto = "manzana" y muestra el total.

Archivo esperado: src/ejercicio_07.py

Estado: Archivo encontrado

Calificación: 5.0/5.0

Retroalimentación:

La solución es correcta, clara y eficiente. Se utiliza una función para encapsular la lógica, lo cual es una buena práctica.

Actividad 8: Usando un ciclo while, calcula y muestra los cuadrados de los números del 1 al 20 (1^2, 2^2, …, 20^2), cada resultado en una línea.

Archivo esperado: src/ejercicio_08.py

Estado: Archivo encontrado

Calificación: 5.0/5.0

Retroalimentación:

La solución es correcta y eficiente. El código es limpio y fácil de entender. ¡Buen trabajo!

Actividad 9: Con un ciclo for, imprime todos los números pares del 2 al 50 (ambos inclusive), cada número en una línea.

Archivo esperado: src/ejercicio_09.py

Estado: Archivo encontrado

Calificación: 4.0/5.0

Retroalimentación:

La solución es correcta y concisa utilizando una comprensión de lista. Sin embargo, el uso de `print` dentro de la comprensión de lista se considera menos legible que un bucle for explícito para este caso. Considera usar un bucle for tradicional para mayor claridad.

Actividad 10: Mediante un ciclo while, solicita al usuario que escriba palabras. El proceso termina cuando el usuario escriba la palabra "fin". Al final, muestra cuántas palabras se leyeron (sin contar "fin").

Archivo esperado: src/ejercicio_10.py

Estado: Archivo encontrado

Calificación: 4.0/5.0

Retroalimentación:

La solución resuelve el problema correctamente. Se puede mejorar la claridad del código eliminando la variable `palabra_lower` ya que se utiliza directamente en el `if` dentro del ciclo. Considera usar `len(contenedor)` para obtener la cantidad de palabras en lugar de mantener un contador.

Actividad 11: Con un ciclo for, imprime cada carácter de la palabra "python" en una línea separada.

Archivo esperado: src/ejercicio_11.py

Estado: Archivo encontrado

Calificación: 4.0/5.0

Retroalimentación:

La solución es correcta y concisa, utilizando una comprensión de lista para imprimir cada carácter. Aunque funcional, el uso de una comprensión de lista solo para imprimir puede ser menos legible que un ciclo for tradicional en este caso.

Actividad 12: Utilizando un ciclo while, calcula el factorial de un número entero n introducido por el usuario y muestra el resultado.

Archivo esperado: src/ejercicio_12.py

Estado: Archivo encontrado

Calificación: 0.0/5.0

Retroalimentación:

Contenido insuficiente - solo comentarios o código trivial

Actividad 13: Con un ciclo for, imprime los números del 1 al 30 saltando de 3 en 3 (1, 4, 7, …, 28), cada número en una línea.

Archivo esperado: src/ejercicio_13.py

Estado: Archivo encontrado

Calificación: 5.0/5.0

Retroalimentación:

Excelente solución. El uso de la comprensión de listas para imprimir los números es conciso y eficiente.

Actividad 14: Mediante un ciclo while, implementa un juego de adivinanza: el programa genera un número aleatorio del 1 al 10 y solicita al usuario que lo adivine. El proceso se repite hasta que el usuario acierte. Muestra un mensaje de felicitación al final.

Archivo esperado: src/ejercicio_14.py

Estado: Archivo encontrado

Calificación: 4.0/5.0

Retroalimentación:

La solución es funcional y correcta. Sin embargo, la generación del número aleatorio debería estar fuera del ciclo `while` para que el número a adivinar sea el mismo durante toda la partida. Se podría mejorar la legibilidad añadiendo un mensaje de error si el usuario no acierta.

Actividad 15: Con un ciclo for, imprime un triángulo rectángulo de 5 filas usando el carácter '*'.

Archivo esperado: src/ejercicio_15.py

Estado: Archivo encontrado

Calificación: 4.0/5.0

Retroalimentación:

La solución es correcta y concisa utilizando una comprensión de lista. Podrías mejorar la legibilidad utilizando un bucle for tradicional en lugar de una comprensión de lista con efecto secundario (impresión).

Actividad 16: Utilizando un ciclo while, simula un reloj digital que muestre cada segundo desde 00:00 hasta 00:59 en formato MM:SS, cada valor en una línea.

Archivo esperado: src/ejercicio_16.py

Estado: Archivo encontrado

Calificación: 3.0/5.0

Retroalimentación:

El código cumple parcialmente con el objetivo, pero tiene errores. La inicialización del contador de minutos es correcta, pero la lógica del ciclo while y el manejo de segundos no son del todo correctos, impidiendo que llegue al segundo 59 y que se formatee correctamente los segundos menores a 10 (ej. 01, 02).

Actividad 17: Con un ciclo for, solicita al usuario que ingrese un número entero positivo y calcula la suma de sus dígitos, mostrando el resultado final.

Archivo esperado: src/ejercicio_17.py

Estado: Archivo encontrado

Calificación: 4.0/5.0

Retroalimentación:

La solución funciona correctamente. Se podría mejorar la legibilidad incluyendo validación de entrada para asegurar que el usuario ingrese un número y agregando comentarios explicando el propósito del código.

Actividad 18: Mediante un ciclo while, genera y muestra la secuencia de Fibonacci empezando por 1, 1, 2, 3, 5, … y termina cuando se alcance el primer valor mayor que 1000.

Archivo esperado: src/ejercicio_18.py

Estado: Archivo encontrado

Calificación: 3.0/5.0

Retroalimentación:

El código genera la secuencia de Fibonacci, pero imprime los números de forma incorrecta (imprime n1, n2 y luego n3 en cada iteración). Debería imprimir la secuencia completa dentro del ciclo while para mostrar la secuencia correcta hasta el primer número mayor que 1000.

Actividad 19: Con un ciclo for, cuenta cuántas vocales (sin distinción de mayúsculas/minúsculas) hay en la frase frase = "programacion es divertida" y muestra el total.

Archivo esperado: src/ejercicio_19.py

Estado: Archivo encontrado

Calificación: 4.0/5.0

Retroalimentación:

La solución es correcta y funcional. Podrías mejorar la legibilidad usando lower() para evitar tener que definir vocales en minúsculas y podrías usar un comentario al inicio para explicar el propósito del script.

Actividad 20: Utilizando un ciclo while, solicita al usuario que ingrese edades una a una. El proceso termina cuando se introduzca -1. Al final, muestra la edad mayor que se haya ingresado.

Archivo esperado: src/ejercicio_20.py

Estado: Archivo encontrado

Calificación: 4.0/5.0

Retroalimentación:

La solución funciona correctamente y cumple con los requisitos. Podrías mejorar la legibilidad evitando el `continue` dentro del `if` y usando una variable más descriptiva para `control`.

Resumen General

Buen trabajo general. Completó 19/20 actividades (95%) con una calificación promedio de 3.9/5. Hay oportunidades de mejora en algunos aspectos.

Recomendaciones

- Revisar y mejorar las actividades con calificación baja