

# Reporte de Evaluación - Fork de GitHub

## Información General

Estudiante: Harrison Olvany Rengifo Marin  
Repositorio: HarrisonRM/act\_ntp\_s3  
Fecha de evaluación: 21/8/2025, 7:34:49  
Evaluado por: Sistema de Evaluación de No Calificados

## Resumen de Calificaciones

Calificación general: 4.4/5.0  
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	5.0
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	No	0.0
4	Utilizando un ciclo while, solicita al u...	src/ejercicio_04.py	Sí	4.0
5	Con un ciclo for, imprime la tabla de mu...	src/ejercicio_05.py	Sí	5.0
6	Mediante un ciclo while, genera y muestr...	src/ejercicio_06.py	Sí	4.0
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	4.0
9	Con un ciclo for, imprime todos los núme...	src/ejercicio_09.py	Sí	5.0
10	Mediante un ciclo while, solicita al usu...	src/ejercicio_10.py	Sí	5.0
11	Con un ciclo for, imprime cada carácter ...	src/ejercicio_11.py	Sí	5.0
12	Utilizando un ciclo while, calcula el fa...	src/ejercicio_12.py	Sí	3.0
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	5.0
15	Con un ciclo for, imprime un triángulo r...	src/ejercicio_15.py	Sí	5.0
16	Utilizando un ciclo while, simula un rel...	src/ejercicio_16.py	Sí	4.0
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	5.0
19	Con un ciclo for, cuenta cuántas vocales...	src/ejercicio_19.py	Sí	5.0
20	Utilizando un ciclo while, solicita al u...	src/ejercicio_20.py	Sí	5.0

## 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: 5.0/5.0

Retroalimentación:

La solución es correcta, concisa y cumple con todos los requisitos. El código es legible y funcional.

### 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:

Excelente solución. El código es claro, conciso y cumple con todos los requisitos del ejercicio. Bien hecho.

### 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 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 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: 4.0/5.0

Retroalimentación:

La solución es correcta y funcional, pero la suma del último número (0) es innecesaria. Se podría mejorar evitando la suma del cero antes de imprimir el resultado final.

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

Archivo esperado: src/ejercicio\_05.py

Estado: Archivo encontrado

Calificación: 5.0/5.0

Retroalimentación:

La solución es correcta, clara y concisa. El código es legible y cumple con el objetivo del problema sin problemas.

### 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: 4.0/5.0

Retroalimentación:

La solución es correcta pero el ciclo `while` podría ser más explícito en cuanto a la cantidad de múltiplos a generar, en lugar de depender del valor del contador. El cálculo del múltiplo dentro del loop es redundante, ya que `contador` ya es el múltiplo.

**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:

Solución correcta y eficiente. El código es legible y cumple con el objetivo planteado utilizando un ciclo for y una condición if de manera clara y concisa.

**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: 4.0/5.0

Retroalimentación:

El código resuelve correctamente el problema, pero se solicitaba usar un ciclo `while` en lugar de `for`. Considera usar `while` para cumplir estrictamente con los requisitos.

**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: 5.0/5.0

Retroalimentación:

La solución es correcta y eficiente. El código es conciso y cumple con los requisitos del problema.

**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: 5.0/5.0

Retroalimentación:

La solución es correcta y eficiente. El uso de `.lower()` para la comparación con 'fin' es una buena práctica para hacer el programa más robusto.

**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: 5.0/5.0

Retroalimentación:

La solución es correcta y eficiente. El código es limpio y fácil de entender. Cumple con todos los requisitos de la actividad.

**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: 3.0/5.0

Retroalimentación:

El código está incompleto. Falta el ciclo `while` para calcular el factorial y la impresión del resultado. Completa el código con la lógica faltante y considera añadir validación para números negativos.

**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:

La solución es correcta y concisa. Utiliza el ciclo `for` con un paso adecuado para imprimir los números solicitados de manera 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: 5.0/5.0

Retroalimentación:

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

**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: 5.0/5.0

Retroalimentación:

Excelente solución. El código es conciso, claro y resuelve el problema planteado de manera eficiente utilizando un ciclo for y la multiplicación de strings. No hay necesidad de mejoras significativas.

**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: 4.0/5.0

Retroalimentación:

La solución funciona correctamente y cumple con los requisitos. Sin embargo, la descripción solicitaba un ciclo `while` en lugar de ciclos `for`. Considera usar `while` para ajustarse completamente a la descripción.

**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 es correcta y funcional. Sin embargo, la descripción indica el uso de un ciclo `for`, pero el código utiliza un ciclo `while`. Considera usar un ciclo `for` como se especifica en la descripción para mejorar la adherencia a los requisitos.

**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: 5.0/5.0

Retroalimentación:

La solución es correcta y eficiente. El código es legible y sigue las buenas prácticas. La pausa `time.sleep()` es un detalle interesante para visualizar la secuencia.

**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: 5.0/5.0

Retroalimentación:

La solución es correcta, clara y eficiente. El código es legible y utiliza buenas prácticas al convertir la frase a minúsculas para la comparación.

**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: 5.0/5.0

Retroalimentación:

La solución es correcta, clara y eficiente. El código es bien estructurado y fácil de entender, cumpliendo con todos los requisitos.

## Resumen General

Excelente trabajo. Completó 19/20 actividades (95%) con una calificación promedio de 4.4/5. Demuestra buen dominio de los conceptos.

## Recomendaciones

- Completar los archivos faltantes: src/ejercicio\_03.py