

# Reporte de Evaluación - Fork de GitHub

## Información General

Estudiante: Samuel Bernal Soto  
Repositorio: samuprogramador333/act\_ntp\_s3  
Fecha de evaluación: 26/7/2025, 11:42:50  
Evaluado por: Sistema de Evaluación

## Resumen de Calificaciones

Calificación general: 3.2/5.0  
Actividades completadas: 14/20  
Porcentaje de completitud: 70.0%

## 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	Sí	3.0
4	Utilizando un ciclo while, solicita al u...	src/ejercicio_04.py	Sí	5.0
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	5.0
7	Con un ciclo for, cuenta cuántas letras ...	src/ejercicio_07.py	Sí	3.0
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	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	0.0
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	0.0
17	Con un ciclo for, solicita al usuario qu...	src/ejercicio_17.py	Sí	0.0
18	Mediante un ciclo while, genera y muestr...	src/ejercicio_18.py	Sí	0.0
19	Con un ciclo for, cuenta cuántas vocales...	src/ejercicio_19.py	Sí	0.0
20	Utilizando un ciclo while, solicita al u...	src/ejercicio_20.py	Sí	0.0

## 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 y cumple con todos los requisitos. El código es limpio y fácil de entender.

### 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, funcional y cumple con los requerimientos. El código es legible y 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: 3.0/5.0

Retroalimentación:

La solución calcula la suma correctamente, pero imprime la suma parcial en cada iteración en lugar de la suma total al final. Sería mejor mover el `print(suma)` fuera del bucle `for`.

### 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: 5.0/5.0

Retroalimentación:

La solución es correcta, concisa y bien estructurada. El código es legible y cumple con la descripción de la actividad.

### 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 y eficiente. El código es legible y cumple con los requisitos de la actividad. ¡Excelente trabajo!

### 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 claro, fácil de entender y cumple con los requisitos 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: 3.0/5.0

Retroalimentación:

La solución funciona pero es sensible a mayúsculas. Considera usar ``lower()`` para hacer el conteo sin importar si la 'a' es mayúscula o minúscula. También, el nombre de la variable 'palabra' tiene un error ortográfico.

**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, clara y eficiente. El código es fácil de leer y cumple con los requisitos de la actividad.

**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:

¡Excelente solución! El código es conciso, legible y cumple perfectamente con los requisitos de la actividad. Buen uso del 'step' en la función range.

**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:

El código resuelve correctamente el problema planteado y es fácil de entender. Se recomienda usar nombres de variables más descriptivos para mejorar la legibilidad, por ejemplo, ``contador_palabras`` en lugar de ``sumar_palabras``.

**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, concisa y cumple con los requisitos de la actividad. El código es limpio y fácil de entender.

**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 calcula correctamente el factorial, pero imprime el resultado en cada iteración del bucle, no solo al final. Deberías mover el ``print`` fuera del ``while`` para mostrar el resultado final una sola vez.

**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 código es conciso, funcional y resuelve correctamente el problema planteado. La elección del nombre de la variable es descriptiva.

**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: 0.0/5.0

**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, legible y resuelve el problema planteado de manera eficiente. Bien hecho.

**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: 0.0/5.0

**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: 0.0/5.0

**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: 0.0/5.0

**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: 0.0/5.0

**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: 0.0/5.0

## **Resumen General**

Buen trabajo general. Completó 14/20 actividades (70%) con una calificación promedio de 3.2/5. Hay oportunidades de mejora en algunos aspectos.

## **Recomendaciones**

- Revisar y mejorar las actividades con calificación baja