

Reporte de Evaluación - Fork de GitHub

Información General

Estudiante: Brahian Angel Ospina
Repositorio: Bangel0/act_ntp_s3
Fecha de evaluación: 25/7/2025, 20:01:22
Evaluado por: Sistema (Re-evaluación individual)

Resumen de Calificaciones

Calificación general: 4.2/5.0
Actividades completadas: 17/20
Porcentaje de completitud: 85.0%

Detalle de Actividades

#	Descripción	Archivo	Encontrado	Calificación
1	Usando un ciclo for, imprime los números...	src/ejercicio_01.py	Sí	4.0
2	Mediante un ciclo while, imprime los núm...	src/ejercicio_02.py	Sí	2.0
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	4.0
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	4.0
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	3.0
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	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	5.0
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	5.0
17	Con un ciclo for, solicita al usuario qu...	src/ejercicio_17.py	No	0.0
18	Mediante un ciclo while, genera y muestr...	src/ejercicio_18.py	No	0.0
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	No	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: 4.0/5.0

Retroalimentación:

El código resuelve el problema aunque empieza en 1 en vez de 0. La variable `numerosAscendentes` es descriptiva, pero el ciclo debería iniciar en 0 para cumplir con la descripción del problema.

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: 2.0/5.0

Retroalimentación:

La solución imprime una lista en lugar de los números individualmente en líneas separadas. Se podría mejorar la lógica del bucle while para imprimir cada número directamente y usar un solo entero como contador en lugar de una lista.

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, pero tiene un error en el rango del ciclo `for`. El ciclo itera hasta `len(suma_res)+1`, lo cual hace que `n` tome el valor 101 en la última iteración, sumando un número fuera del rango solicitado. Además, la lista `suma_res` no es necesaria, el ciclo `for` puede iterar directamente sobre el rango 1-101.

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:

El código resuelve el problema, pero incluye el 0 en la suma final. Podrías excluir el 0 del cálculo de la suma para una solución más precisa. Además, la función podría ser más reutilizable si devolviera la suma en lugar de imprimirla directamente.

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. Sin embargo, usar una comprensión de lista solo para imprimir no es la forma más legible. Preferiría un bucle for explícito para mayor claridad.

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 lógica es correcta y genera los múltiplos de 3. Sin embargo, la condición `multiplos != 16` no es la forma más clara de expresar que quieres 15 múltiplos. Podrías considerar usar un contador o ajustar la condición del while para ser más explícito sobre el número de iteraciones.

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 resuelve el problema, pero usa nombres de variables poco descriptivos, lo que dificulta la lectura. Se podría mejorar la legibilidad usando nombres más significativos y la función debería recibir el texto como argumento en lugar de tenerlo hardcoded.

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: 3.0/5.0

Retroalimentación:

La solución funciona, pero no utiliza un ciclo `while` de la manera más eficiente. Se podría simplificar el código utilizando un contador en lugar de un iterador y una lista predefinida.

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. Se puede mejorar la legibilidad al usar un ciclo for tradicional en lugar de una lista de comprensión para imprimir los números.

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()` hace que el programa sea más robusto al aceptar 'Fin', 'FIN', etc.

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 concisa. Cumple con el objetivo de la actividad utilizando un ciclo for de manera eficiente para imprimir cada carácter de la palabra 'python' en una línea separada.

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: 5.0/5.0

Retroalimentación:

La solución es correcta y funcional. El código es limpio, legible y cumple con los requisitos del problema utilizando un ciclo while para calcular el factorial.

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

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 funcional, claro y conciso, resolviendo el problema planteado de manera eficiente. 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:

La solución es correcta y concisa. Utiliza un ciclo for de forma eficiente para generar el triángulo rectángulo.

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: 5.0/5.0

Retroalimentación:

Excelente solución. El código es conciso, funcional y utiliza el formateo correcto para la presentación de la hora. Cumple a la perfección con los requisitos del ejercicio.

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 no encontrado

Calificación: 0.0/5.0

Retroalimentación:

Error al evaluar: got status: 429 . {"error":{"code":429,"message":"You exceeded your current quota, please check your plan and billing details. For more information on this error, head to: <https://ai.google.dev/gemini-api/docs/rate-limits>."},"status":"RESOURCE_EXHAUSTED","details":{"@type":"type.googleapis.com/google.rpc.QuotaFailure","violations":[{"quotaMetric":"generativelanguage.googleapis.com/generate_content_free_tier_requests","quotaId":"GenerateRequestsPerMinutePerProjectPerModel-FreeTier","quotaDimensions":{"location":"global","model":"gemini-2.0-fla sh"},"quotaValue":"15"}]},{"@type":"type.googleapis.com/google.rpc.Help","links":[{"description":"Learn more about Gemini API quotas","url":"https://ai.google.dev/gemini-api/docs/rate-limits"}]},{"@type":"type.googleapis.com/google.rpc.RetryInfo","retryDelay":"8s"}]}}

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 no encontrado

Calificación: 0.0/5.0

Retroalimentación:

Error al evaluar: got status: 429 . {"error":{"code":429,"message":"You exceeded your current quota, please check your plan and billing details. For more information on this error, head to: <https://ai.google.dev/gemini-api/docs/rate-limits>."},"status":"RESOURCE_EXHAUSTED","details":[{"@type":"type.googleapis.com/google.rpc.QuotaFailure","violations":[{"quotaMetric":"generativelanguage.googleapis.com/generate_content_free_tier_requests","quotaId":"GenerateRequestsPerMinutePerProjectPerModel-FreeTier","quotaDimensions":{"location":"global","model":"gemini-2.0-flan"},"quotaValue":15}]}]},{"@type":"type.googleapis.com/google.rpc.Help","links":[{"description":"Learn more about Gemini API quotas","url":"https://ai.google.dev/gemini-api/docs/rate-limits"}]}]},{"@type":"type.googleapis.com/google.rpc.RetryInfo","retryDelay":7s}}}

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 concisa. El código cumple con el objetivo propuesto y es fácil de entender.

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 no encontrado

Calificación: 0.0/5.0

Retroalimentación:

Error al evaluar: got status: 429 . {"error":{"code":429,"message":"You exceeded your current quota, please check your plan and billing details. For more information on this error, head to: <https://ai.google.dev/gemini-api/docs/rate-limits>."},"status":"RESOURCE_EXHAUSTED","details":[{"@type":"type.googleapis.com/google.rpc.QuotaFailure","violations":[{"quotaMetric":"generativelanguage.googleapis.com/generate_content_free_tier_requests","quotaId":"GenerateRequestsPerMinutePerProjectPerModel-FreeTier","quotaDimensions":{"location":"global","model":"gemini-2.0-flan"},"quotaValue":15}]}]},{"@type":"type.googleapis.com/google.rpc.Help","links":[{"description":"Learn more about Gemini API quotas","url":"https://ai.google.dev/gemini-api/docs/rate-limits"}]}]},{"@type":"type.googleapis.com/google.rpc.RetryInfo","retryDelay":4s}}}

Resumen General

Excelente trabajo. Completó 17/20 actividades (85%) con una calificación promedio de 4.2/5. Demuestra buen dominio de los conceptos.

Recomendaciones

- Completar los archivos faltantes: src/ejercicio_17.py, src/ejercicio_18.py, src/ejercicio_20.py
- Revisar y mejorar las actividades con calificación baja