100 Ejercicios

Tema 1: Introducción a Python

Temas Cubiertos:

1. Instalación y Configuración del Entorno de Desarrollo
2. Sintaxis Básica
3. Variables y Tipos de Datos
4. Estructuras de Control
5. Funciones y Módulos
6. Manejo de Excepciones

1ra Parte

- **50 Ejercicios para Practicar y Afianzar los Conocimientos del Tema 1**: Estos ejercicios están diseñados para ayudar a los estudiantes a comprender y practicar los conceptos básicos de Python, como la instalación, la sintaxis básica, las variables y tipos de datos, las estructuras de control, las funciones y módulos, y el manejo de excepciones.

2da Parte

- **50 Ejercicios Avanzados para Llevar al Límite la Programación en Python sobre el Tema 1**: Estos ejercicios están diseñados para llevar a los estudiantes más allá de los conceptos básicos, explorando características avanzadas y técnicas de programación en Python que desafían su comprensión y habilidades, proporcionando una experiencia de aprendizaje más profunda y rica.

1era Parte

1. Instalación y Configuración

1. **Instalación de Python**: Instala Python en tu sistema operativo y verifica la instalación ejecutando python --version en la terminal.

2. **Configuración del Entorno de Desarrollo:** Configura Visual Studio Code o Jupyter Notebook como tu entorno de desarrollo preferido.

2. Sintaxis Básica

3. **Hola, Mundo:** Escribe un programa que imprima "Hola, Mundo".
4. **Comentarios:** Escribe un programa que incluya comentarios de una línea y de múltiples líneas.
5. **Entrada de Usuario:** Escribe un programa que solicite al usuario su nombre y lo salude.
6. **Cálculo Simple:** Escribe un programa que calcule la suma de dos números proporcionados por el usuario.
7. **Condicionales:** Escribe un programa que verifique si un número proporcionado por el usuario es positivo, negativo o cero.

3. Variables y Tipos de Datos

8. **Declaración de Variables:** Declara variables de diferentes tipos de datos (entero, flotante, cadena, booleano) y imprímelas.
9. **Conversión de Tipos:** Escribe un programa que convierta un número flotante a entero y viceversa.
10. **Operaciones Matemáticas:** Realiza operaciones matemáticas básicas (suma, resta, multiplicación, división) con variables.
11. **Formateo de Cadenas:** Escribe un programa que utilice f-strings para formatear cadenas.
12. **Longitud de Cadena:** Escribe un programa que calcule la longitud de una cadena proporcionada por el usuario.

4. Estructuras de Control

13. **Condicional Simple:** Escribe un programa que verifique si un número proporcionado por el usuario es par o impar.

14. **Condicional Anidado:** Escribe un programa que verifique si un número está en un rango específico (por ejemplo, entre 1 y 100).

15. **Bucle for:** Escribe un programa que imprima los números del 1 al 10.

16. **Bucle while:** Escribe un programa que imprima los números del 10 al 1 en orden descendente.

17. **Bucle con break:** Escribe un programa que imprima los números del 1 al 10 y se detenga si encuentra un número divisible por 4.

18. **Bucle con continue:** Escribe un programa que imprima los números del 1 al 10, pero omita los números pares.

19. **Bucle Anidado:** Escribe un programa que imprima una tabla de multiplicar del 1 al 5.

20. **List Comprehensions:** Escribe un programa que cree una lista de cuadrados de números del 1 al 10 utilizando list comprehensions.

5. Funciones y Módulos

21. **Función Simple:** Escribe una función que tome un número y devuelva su cuadrado.

22. **Función con Múltiples Parámetros:** Escribe una función que tome dos números y devuelva su suma.

23. **Función con Valor por Defecto:** Escribe una función que tome dos números y devuelva su suma, con un valor por defecto para el segundo número.

24. **Función Recursiva:** Escribe una función recursiva que calcule el factorial de un número.

25. **Importación de Módulos:** Escribe un programa que importe el módulo math y utilice la función sqrt.

26. **Módulo Propio:** Crea un módulo propio con una función que salude al usuario.

6. Manejo de Excepciones

27. **Manejo Básico de Excepciones:** Escribe un programa que maneje una excepción de división por cero.

28. **Excepción Específica:** Escribe un programa que maneje una excepción ValueError cuando el usuario introduce un valor no numérico.

29. **Bloques try y finally:** Escribe un programa que utilice los bloques try, except y finally.

30. **Multiples Except:** Escribe un programa que maneje múltiples tipos de excepciones (ValueError, ZeroDivisionError).

Ejercicios de Repaso

31. **Sumar Dígitos:** Escribe un programa que sume los dígitos de un número proporcionado por el usuario.

32. **Número Primo:** Escribe un programa que verifique si un número es primo.

33. **Invertir Cadena:** Escribe un programa que invierta una cadena proporcionada por el usuario.

34. **Contar Vocales:** Escribe un programa que cuente el número de vocales en una cadena.

35. **Palíndromo:** Escribe un programa que verifique si una cadena es un palíndromo.

36. **Máximo de Tres:** Escribe una función que devuelva el máximo de tres números.

37. **Tabla de Multiplicar:** Escribe un programa que imprima la tabla de multiplicar de un número proporcionado por el usuario.

38. **Sumar Lista:** Escribe una función que sume todos los elementos de una lista.

39. **Producto de Lista:** Escribe una función que calcule el producto de todos los elementos de una lista.

40. **Encontrar el Mínimo:** Escribe una función que encuentre el mínimo de una lista de números.

41. **Contar Elementos**: Escribe un programa que cuente el número de elementos en una lista.
42. **Remover Duplicados**: Escribe una función que remueva los elementos duplicados de una lista.
43. **Ordenar Lista**: Escribe un programa que ordene una lista de números.
44. **Merge Sorted Lists**: Escribe una función que combine dos listas ordenadas en una sola lista ordenada.
45. **Anagramas**: Escribe un programa que verifique si dos cadenas son anagramas.
46. **Generador de Contraseñas**: Escribe un programa que genere una contraseña aleatoria de una longitud especificada.
47. **Calculadora Simple**: Escribe un programa que actúe como una calculadora simple.
48. **Conversión de Temperatura**: Escribe un programa que convierta la temperatura de grados Celsius a Fahrenheit y viceversa.
49. **Contador de Palabras**: Escribe un programa que cuente el número de palabras en una cadena.
50. **Suma de Pares e Impares**: Escribe un programa que sume por separado los números pares e impares de una lista de números.

2da Parte

Instalación y Configuración

1. **Configuración del Entorno**: Configura un entorno virtual utilizando venv o virtualenv y explica los beneficios de utilizar entornos virtuales.
2. **Docker**: Crea un archivo Dockerfile para un entorno Python y ejecuta un contenedor con Python instalado.

Sintaxis Básica

3. **Documentación**: Documenta un script utilizando docstrings y genera documentación HTML utilizando Sphinx.

4. **Tipos Anotados:** Utiliza anotaciones de tipo para una función que suma dos números y devuelve el resultado.
5. **Linting:** Configura pylint para un proyecto y corrige los errores de estilo en un script.
6. **Formateo Automático:** Utiliza black para formatear automáticamente un script de Python.

Variables y Tipos de Datos

7. **Variables de Entorno:** Escribe un script que lea variables de entorno y las utilice para la configuración del programa.
8. **Serialización JSON:** Escribe un programa que serialice y deserialice datos en formato JSON utilizando el módulo json.
9. **Expresiones Regulares:** Escribe un programa que utilice expresiones regulares para validar una dirección de correo electrónico.
10. **Conversiones Avanzadas:** Convierte una cadena JSON compleja a un diccionario y viceversa.
11. **Manejo de Fechas:** Utiliza el módulo datetime para calcular la diferencia en días entre dos fechas proporcionadas por el usuario.

Estructuras de Control

12. **Compresión de Lista Avanzada:** Genera una lista de números primos utilizando list comprehensions y una función generadora.
13. **Condicionales en Comprehensions:** Crea un diccionario de cuadrados de números del 1 al 10, pero solo incluye números pares.
14. **Iteradores Personalizados:** Implementa un iterador personalizado que devuelva los primeros n números cuadrados.
15. **Recursión:** Escribe una función recursiva para resolver el problema de la Torre de Hanoi.

16. **Itertools:** Utiliza itertools para generar todas las permutaciones de una lista de elementos.
17. **Generadores:** Implementa un generador que produzca una secuencia infinita de números de Fibonacci.
18. **Bucles Eficientes:** Escribe un script que procese un archivo de texto grande línea por línea utilizando generadores.

Funciones y Módulos

19. **Decoradores:** Crea un decorador que registre la entrada y salida de una función en un archivo de log.
20. **Funciones de Orden Superior:** Escribe una función que acepte otra función como argumento y aplique esa función a cada elemento de una lista.
21. **Función Parcial:** Utiliza functools.partial para crear una versión específica de una función genérica.
22. **Módulos Personalizados:** Crea un módulo propio con funciones matemáticas avanzadas y documenta cada función con docstrings.
23. **Importación Dinámica:** Escribe un programa que importe dinámicamente un módulo en tiempo de ejecución basado en la entrada del usuario.
24. **Context Managers:** Implementa un contexto personalizado utilizando contextlib.contextmanager para manejar archivos de manera segura.

Manejo de Excepciones

25. **Excepciones Personalizadas:** Define una excepción personalizada y utilízala en un programa que valide la entrada del usuario.
26. **Propagación de Excepciones:** Escribe un programa que maneje una excepción en una función y la propague a la función de llamada.
27. **Reintento Automático:** Implementa un decorador que intente ejecutar una función varias veces en caso de una excepción específica.

28. **Manejo de Excepciones Anidadas:** Escribe un programa que maneje excepciones anidadas en varios niveles de llamadas de función.

29. **Bloque Try-Except-Finally:** Utiliza un bloque try-except-finally para garantizar la liberación de recursos en un programa que lee de un archivo y maneja posibles excepciones.

Ejercicios de Repaso

30. **Programación Defensiva:** Escribe una función que verifique la validez de los argumentos y maneje posibles errores antes de realizar operaciones.

31. **Optimización con Profiler:** Utiliza cProfile para analizar el rendimiento de una función que realiza cálculos complejos y optimiza el código.

32. **Memoización:** Implementa una función recursiva con memoización para calcular los números de Fibonacci de manera eficiente.

33. **Manejo de Archivos Grandes:** Escribe un programa que procese un archivo CSV grande utilizando generadores para manejar la memoria eficientemente.

34. **Programación Funcional:** Utiliza map(), filter() y reduce() para procesar una lista de números y encontrar la suma de los cuadrados de los números pares.

35. **Ejecución Condicional:** Escribe un script que ejecute diferentes funciones basadas en la entrada del usuario utilizando un diccionario de funciones.

36. **Validación de Datos:** Crea una función que valide la estructura de un objeto JSON contra un esquema definido.

37. **Decoradores Anidados:** Implementa decoradores anidados que agreguen múltiples capas de funcionalidad a una función.

38. **Operaciones con Bitwise:** Escribe funciones que realicen operaciones bit a bit y utilízalas en ejemplos prácticos.

39. **Serialización Avanzada:** Implementa la serialización y deserialización de objetos complejos utilizando el módulo pickle.

Desafíos de Código

40. **Compresión de Datos:** Implementa un algoritmo simple de compresión de datos y escribe un programa que lo utilice para comprimir y descomprimir archivos de texto.
41. **Automatización de Tareas:** Utiliza subprocess para escribir un script que automatice una tarea común en tu sistema operativo.
42. **Análisis de Texto:** Escribe un programa que realice un análisis de frecuencia de palabras en un documento de texto grande.
43. **Generación de Archivos PDF:** Utiliza ReportLab para generar un archivo PDF con contenido dinámico basado en la entrada del usuario.
44. **Cliente HTTP Personalizado:** Implementa un cliente HTTP básico utilizando socket para realizar solicitudes GET y POST.
45. **Web Scraping:** Utiliza BeautifulSoup para extraer datos estructurados de una página web y guardarlos en un archivo CSV.
46. **Automatización de Navegación Web:** Utiliza Selenium para automatizar la interacción con un sitio web y realizar una tarea repetitiva.
47. **Cifrado y Descifrado:** Escribe un programa que cifre y descifre textos utilizando un algoritmo de cifrado simple como el cifrado César.
48. **Generador de Código QR:** Utiliza qrcode para generar códigos QR a partir de cadenas de texto proporcionadas por el usuario.
49. **Procesamiento de Imágenes:** Utiliza Pillow para realizar operaciones básicas de procesamiento de imágenes, como redimensionar y recortar.

50. **Juego Simple:** Utiliza pygame para desarrollar un juego simple como el clásico "Pong".