Python Tutor

Curriculum Completo

Generado automáticamente

Tabla de Contenidos

Es Primo

N Números Primos

Primos en Rango

Suma de Dos Índices

Módulo 04: Automatización y Scripts

Avatar Básico

Avatar Personalizado

Avatar Personalizado II

Avatar Aleatorio

Reloj Digital Básico

Reloj Digital Avanzado

Reloj Digital CLI

Reloj Mundial

Captura de Cámara (OpenCV - Headless)

Imagen a Caricatura (Real)

Caricatura con GUI (Web)

Generador QR (API Real)

Info de Imagen (PIL)

Análisis de Texto (Frecuencia)

Análisis de Sentimiento (Simple)

Validador de Email (Regex)

Generador PDF (Reporte)

Texto a Voz Simple

Espiral Arcoíris (SVG Turtle)

Triángulo de Sierpinski (SVG)

Arte Geométrico (SVG)

Cuadrado y Círculo (SVG)

Cuadrado Simple (SVG)

Colores en Terminal (Colorama)

Descargar Archivos

Generar OTP

Generador de Contraseñas

Módulo 00: Bienvenida y Guía

Guía de Inicio y Créditos

🚀 Módulo 00 — Bienvenida y Guía del Curso
(Versión Hacker Futurista)

👋 Bienvenido al Curso de Python orientado a Ciberseguridad
Autor: Carlos Domínguez
Asistente Tecnológico: IA Generativa

Bienvenido, operad@r.
Has accedido al entorno de entrenamiento donde aprenderás Python desde cero hasta un nivel avanzado, con un enfoque real en:

* 🛠 Automatización
* 🔐 Ciberseguridad y criptografía
* 🧠 Análisis técnico
* ⚙️ Creación de herramientas y scripts operativos

Aquí no solo aprenderás a programar...
Aprenderás a pensar como un desarrollador, como un analista, como un hacker ético.

🎯 ¿Qué encontrarás en este curso?
Cada módulo contiene:

* Explicaciones claras
* Ejemplos reales
* Ejercicios interactivos
* Código “escribiéndose en tiempo real” estilo hacking
* Retos prácticos

* Mini-proyectos

Al finalizar serás capaz de:

* ✅ Dominar la sintaxis esencial
* ✅ Automatizar procesos
* ✅ Crear herramientas funcionales
* ✅ Escribir scripts de seguridad
* ✅ Analizar y resolver problemas
* ✅ Construir proyectos reales

🧬 Filosofía del Curso
> *“Este curso es totalmente gratuito y nació con el propósito de compartir conocimiento y ayudarte a desarrollar habilidades tecnológicas reales.”*

Tú marcas el ritmo.
Tu progreso depende de experimentar, no de memorizar.

🛠️ Recomendaciones
1. **Usa herramientas reales** (VSCode, PyCharm, Thonny)
2. **Experimenta sin miedo**: rompe, repara, aprende
3. **Sé constante**: 20 min/día cambian tu nivel

📚 Esquema del Plan de Estudios
* 🔹 **Módulo 00** — Bienvenida y Guía del Curso
* 🔹 **Módulo 01** — Introducción
* 🔹 **Módulo 02** — Conceptos Fundamentales
* 🔹 **Módulo 03** — Práctica de Algoritmos
* 🔹 **Módulo 04** — Automatización y Scripts
* 🔹 **Módulo 05** — Ciberseguridad y Criptografía
* 🔹 **Módulo 06** — Proyectos Finales

* 🔹 **Módulo 07** — Certificación

🌐 Conexión establecida...
```bash
Iniciando entorno PythonSecOS v1.0...
Cargando módulos...
Autenticando alumno...
ACCESO CONCEDIDO
```

Bienvenido operad@.
Aquí cada línea de código construye tu habilidad.
Prepárate para escribir, ejecutar y romper código en un entorno estilo laboratorio de ciberseguridad.

Código de Ejemplo

```
# 🧬 Módulo 00 — Bienvenida y Guía del Curso (Versión Hacker Futurista)
# ┌──────────────────────────────────────────────┐
# │   PYTHONSEC OS v1.0 – ENTORNO ACTIVADO     │
# └──────────────────────────────────────────────┘
print(">> Inicializando protocolo de bienvenida...")
print(">> Cargando parámetros del curso...")
print(">> Autenticando operad@...")
print("✔ ACCESO CONCEDIDO")

# 👋 Bienvenido/a al Curso de Python orientado a Ciberseguridad
# Autor: Carlos Domínguez
# Asistente Tecnológico: IA Generativa
# Créditos: Plataforma diseñada y desarrollada por Carlos Domínguez con la asistencia de Inteligencia Artificial.

# Bienvenido, operad@r.
# Has accedido al entorno de entrenamiento donde desarrollarás habilidades reales en Python,
# desde nivel cero hasta un nivel avanzado, con un enfoque
```

```
especializado en:

# 🛠 Automatización
# 🔐 Ciberseguridad y Criptografía
# 🧠 Análisis técnico
# ⚙️ Creación de herramientas y scripts operativos

# Aquí no solo vas a programar:
# vas a aprender a pensar como un analista, como un desarrollador… y
como un hacker ético.

# 🎯 ¿Qué encontrarás en este curso?
# Cada módulo incluye:
# - Explicaciones claras
# - Ejemplos reales
# - Ejercicios interactivos
# - Código “escribiéndose en tiempo real” estilo terminal hacker
# - Retos prácticos
# - Mini-proyectos progresivos

# Cuando completes este protocolo de entrenamiento serás capaz de:
# ✔ Dominar la sintaxis esencial de Python
# ✔ Automatizar procesos reales
# ✔ Crear herramientas y utilidades funcionales
# ✔ Escribir scripts de seguridad
# ✔ Analizar código y resolver problemas
# ✔ Construir proyectos completos

# 🧬 Filosofía del Curso
# “Este curso es 100% gratuito. Nace con el propósito de compartir
conocimiento
# y ayudarte a desarrollar habilidades tecnológicas reales.”

# Tú controlas el ritmo.
# Tu progreso no depende de memorizar, sino de experimentar.

# 🛠️ Recomendaciones
# 1️⃣ Usa herramientas reales (VSCode, PyCharm, Thonny)
# 2️⃣ Experimenta sin miedo: rompe → repara → rompe → aprende
# 3️⃣ Construye constancia: 20 min/día te llevarán lejos
```

```
# 📚 Plan de Estudios (Definitivo)
#  🔹 Módulo 00 — Bienvenida y Guía del Curso
#  🔹 Módulo 01 — Introducción
#  🔹 Módulo 02 — Conceptos Fundamentales
#  🔹 Módulo 03 — Práctica de Algoritmos
#  🔹 Módulo 04 — Automatización y Scripts
#  🔹 Módulo 05 — Ciberseguridad y Criptografía
#  🔹 Módulo 06 — Proyectos Finales
#  🔹 Módulo 07 — Certificación

print(">> Conexión establecida.")
print(">> Cargando módulos…")
print(">> Iniciando entorno PythonSecOS v1.0…")
print("✔ Sistema operativo de aprendizaje listo.")

print("\nBienvenido, operad@.")
print("Cada línea de código será un nivel más en tu evolución.")
print("Prepárate para escribir, ejecutar y romper código como en un
auténtico laboratorio de ciberseguridad.")
```

Módulo 01: Introducción

Hola Mundo y Print

Estudia el siguiente código: Hola Mundo y Print

Código de Ejemplo

```
###
# 01 - print()
# El módulo print() es el módulo que nos permite imprimir en consola
# Sirve para mostrar información en consola y te va a acompañar
# TODA TU VIDA. Desde hoy hasta el fin de los tiempos
###

# Este es un ejemplo básico de cómo imprimir un texto en consola
print("¡Hola, Twitch!")

# También puedes usar comillas simples para imprimir texto
print('Esto también funciona con una comilla')

# Puedes imprimir múltiples elementos separados por un espacio
print("Python", "es", "genial")

# El parámetro 'sep' permite definir cómo se separan los elementos
impresos
print("Python", "es", "brutal", sep = "-")

# El parámetro 'end' define lo que se imprime al final de la línea
print("Esto se imprime", end = "\n") # Aquí, el 'end' tiene un salto
de línea explícito
print("en una línea") # Esto se imprime en la línea siguiente

# También se pueden imprimir números directamente
print(42)

# Ejemplo de cómo imprimir el símbolo de pulgadas (")
```

```
# Si usamos comillas dobles dentro de un string con comillas dobles,
se produce un error:
# print("Esto es una "pulgada"")  # ❌ Esto generaría un error de
sintaxis

# ✅ Solución 1: Usar comillas simples para encerrar la cadena
print('Esto es una "pulgada" dentro de un string con comillas
simples')

# ✅ Solución 2: Usar el carácter de escape \ para incluir comillas
dobles dentro de un string con comillas dobles
print("Esto es una \"pulgada\" dentro de un string con comillas
dobles")

# ✅ Solución 3: Usar triple comillas para definir el string
print("""Esto es una "pulgada" dentro de un string con triple
comillas""")

# --- EJERCICIO INTERACTIVO ---
# 1. Ejecuta el código tal como está para ver el resultado.
# 2. Modifica el código para cambiar el comportamiento.
#    (Pista: Intenta cambiar los valores de las variables o la lógica)
# 3. Vuelve a ejecutar para ver tus cambios.
```

Tipos de Datos

Estudia el siguiente código: Tipos de Datos

Código de Ejemplo

```
###
# 02 - types()
# Python tiene varios tipos de datos
# int, float, complex, str, bool, NoneType, list, tuple, dict, range,
set...
###
```

```
"""
El comando `type()` devuelve el tipo de un objeto en Python
"""
print("int:")  # Enteros (números sin parte decimal)
print(type(10))  # Número entero positivo
print(type(0))  # El número cero también es un entero
print(type(-5))  # Número entero negativo
print(type(723842472378427893478923987 4))  # Python permite enteros de
gran tamaño

print("float:")  # Números decimales (de punto flotante)
print(type(3.14))  # Número con punto decimal
print(type(1.0))  # También es considerado un float, aunque sea un
número entero con punto decimal
print(type(1e3))  # Notación científica (equivalente a 1000.0)

print("complex:")  # Números complejos (con parte real e imaginaria)
print(type(1 + 2j))  # Un número complejo en Python (1 es la parte
real, 2j es la parte imaginaria)

print("str:")  # Cadenas de texto (strings)
print(type("Hola"))  # Un string con texto
print(type(""))  # Un string vacío
print(type("123"))  # Aunque parezca un número, está entre comillas,
por lo que es un string
print(type("""
  Multilinea
"""))  # Un string que abarca varias líneas usando triple comillas

print("bool:")  # Valores booleanos (True o False)
print(type(True))  # Valor booleano verdadero
print(type(False))  # Valor booleano falso
print(type(1 < 2))  # Comparación que devuelve un booleano (True)

print("NoneType:")  # Representa la ausencia de valor
print(type(None))  # `None` es un tipo especial en Python que
representa "sin valor" o "nulo"

# --- EJERCICIO INTERACTIVO ---
# 1. Ejecuta el código tal como está para ver el resultado.
```

```
# 2. Modifica el código para cambiar el comportamiento.
#    (Pista: Intenta cambiar los valores de las variables o la lógica)
# 3. Vuelve a ejecutar para ver tus cambios.
```

Conversión de Tipos (Casting)

Estudia el siguiente código: Conversión de Tipos (Casting)

Código de Ejemplo

```
###
# 03 - casting de types
# Transformar un tipo de un valor a otro
###



print("Conversión de tipos")

# Convertir una cadena que contiene un número a un entero y sumarlo
con otro entero
print(2 + int("100"))  # Convierte "100" a entero y suma 2. Resultado:
102

# Convertir un entero a cadena para concatenarlo con otra cadena
print("100" + str(2))  # Convierte el número 2 a cadena y lo
concatena. Resultado: "1002"

# Convertir una cadena con un número decimal a tipo float
print(type(float("3.1416")))  # Convierte "3.1416" a float y muestra
su tipo. Resultado: <class 'float'>

# Convertir un número decimal a entero (se trunca la parte decimal)
print(int(3.1416))  # Convierte 3.1416 a 3 eliminando la parte
decimal. Resultado: 3

# Evaluar valores numéricos como booleanos
```

```
print(bool(3))  # Cualquier número distinto de 0 es True. Resultado:
True
print(bool(0))  # 0 es False. Resultado: False
print(bool(-1))  # Números negativos también son True. Resultado: True

# Evaluar cadenas como booleanos
print(bool(""))  # Una cadena vacía es False. Resultado: False
print(bool(" "))  # Una cadena con espacios es True. Resultado: True
print(bool("False"))  # Una cadena con texto, aunque sea "False", es
True. Resultado: True

# Redondear un número decimal
print(round(2.51))  # Redondea 2.51 al entero más cercano. Resultado:
3

# Este genera un error y se comenta para evitar conflicto en la
ejecución
# print(int("Hola mundo"))  # ❌ Esto generaría un ValueError porque
"Hola mundo" no es un número

# --- EJERCICIO INTERACTIVO ---
# 1. Ejecuta el código tal como está para ver el resultado.
# 2. Modifica el código para cambiar el comportamiento.
#    (Pista: Intenta cambiar los valores de las variables o la lógica)
# 3. Vuelve a ejecutar para ver tus cambios.
```

Variables

Estudia el siguiente código: Variables

Código de Ejemplo

```
##
# 04 - Variables
# Las variables sirven para guardar datos en memoria.
# Python es un lenguaje de tipado dinámico y de tipado fuerte.
###
```

```
# Para asignar una variable solo hace falta poner el nombre de la
variable y asignarle un valor
my_name = "midudev"
print(my_name)  # Imprime el valor de la variable my_name

age = 32
print(age)  # Imprime el valor de la variable age)

# Reasignar un nuevo valor a una variable existente
age = 39
print(age)  # Ahora la variable age tiene el valor 39

# Tipado dinámico: el tipo de dato se determine en tiempo de ejecución
# No es necesario declarar explícitamente el tipo de variable
name = "midudev"
print(type(name))  # Muestra el tipo de dato de la variable name (str)

name = 32
print(type(name))  # Ahora la variable tiene un número entero (int)

# Tipado fuerte: Python no realiza conversione de tipo automáticas
# Esto generará un error porque no se puede sumar un número con una
cadena
# print(10 + "2")  # ❌ TypeError: unsupported operand type(s) for +:
'int' and 'str'

# f-string (literal de cadena de formato)
# desde la versión Python 3.6
print(f"Hola {my_name}, tengo {age + 5} años")

# No recomendada forma de asignar variables
name, age, city = "midudev", 32, "Bogotá"

# Convenciones de nombres de variables
mi_nombre_de_variable = "ok" # snake_case
nombre = "ok"

miNombreDeVariable = "no-recomendado" # camelCase
MiNombreDeVariable = "no-recomendado" # PascalCase
```

```
minombredevariable = "no-recomendado" # todojunto

mi_nombre_de_variable_123 = "ok"

MI_CONSTANTE = 3.14 # UPPER_CASE -> constantes

# Nombres NO válidos de variables (esto generaría errores)
# 123123_variable = "ko"  # ❌ No puede comenzar con un número
# mi-variable = "ko"  # ❌ No puede contener guiones (-), usa guion bajo (_)
# mi variable = "ko"  # ❌ No puede contener espacios
# True = False  # ❌ No puedes sobrescribir palabras reservadas

# Palabras reservadas en Python (no se pueden usar como nombres de variables)

# ['False', 'None', 'True', 'and', 'as', 'assert',
# 'async', 'await', 'break', 'class', 'continue',
# 'def', 'del', 'elif', 'else', 'except', 'finally',
# 'for', 'from', 'global', 'if', 'import', 'in', 'is',
# 'lambda', 'nonlocal', 'not', 'or', 'pass', 'raise',
# 'return', 'try', 'while', 'with', 'yield']

# Anotaciones de tipo (opcional, para mayor claridad en el código)
is_user_logged_in: bool = True # Indica que la variable es un booleano
print(is_user_logged_in)

name: str = "midudev" # Indica que la variable es una cadena de texto
print(name)



# --- EJERCICIO INTERACTIVO ---
# 1. Ejecuta el código tal como está para ver el resultado.
# 2. Modifica el código para cambiar el comportamiento.
#    (Pista: Intenta cambiar los valores de las variables o la lógica)
# 3. Vuelve a ejecutar para ver tus cambios.
```

Entrada de Datos (Input)

Estudia el siguiente código: Entrada de Datos (Input)

Código de Ejemplo

```
###
# 05 - Entrada de usuario (input()) - Versión simplificada
# La función input() permite obtener datos del usuario a través de la
consola.
###



# Para obtener datos del usuario se usa la función input()
# La función input() recibe un mensaje que se muestra al usuario
# y devuelve el valor introducido por el usuario
# nombre = input("Hola, ¿cómo te llamas?\n")
nombre = "Carlos" # Valor hardcodeado para demo
print(f"Hola {nombre}, encantado de conocerte")

# Ten en cuenta que la función input() devuelve un string
# Así que si queremos obtener un número se debe convertir el string a
un número
# age = input("¿Cuántos años tienes?\n")
age = "25" # Valor hardcodeado para demo
age = int(age)
print(f"Tienes {age} años")

# La función input() también puede devolver múltiples valores
# Para hacerlo, el usuario debe separar los valores con una coma
print("Obtener múltiples valores a la vez")
# country, city = input("¿En qué país y ciudad vives?\n").split()
country, city = "España Madrid".split()

print(f"Vives en {country}, {city}")

# --- EJERCICIO INTERACTIVO ---
# 1. Ejecuta el código tal como está para ver el resultado.
# 2. Modifica el código para cambiar el comportamiento.
```

```
#    (Pista: Intenta cambiar los valores de las variables o la lógica)
# 3. Vuelve a ejecutar para ver tus cambios.
```

Ejercicios Básicos

Estudia el siguiente código: Ejercicios Básicos

Código de Ejemplo

```
###
# exercises.py
# Ejercicios para practicar los conceptos aprendidos en las lecciones.
###



print("\nEjercicio 1: Imprimir mensajes")
print("Escribe un programa que imprima tu nombre y tu ciudad en líneas
separadas.")

### Completa aquí

print("--------------")

print("\nEjercicio 2: Muestra los tipos de datos de las siguientes
variables:")
print("Usa el comando 'type()' para determinar el tipo de datos de
cada variable.")
a = 15
b = 3.14159
c = "Hola mundo"
d = True
e = None

### Completa aquí

print("--------------")
```

```
print("\nEjercicio 3: Casting de tipos")
print("Convierte la cadena \"12345\" a un entero y luego a un float.")
print("Convierte el float 3.99 a un entero. ¿Qué ocurre?")

### Completa aquí

print("--------------")

print("\nEjercicio 4: Variables")
print("Crea variables para tu nombre, edad y altura.")
print("Usa f-strings para imprimir una presentación.")

# "Hola! Me llamo midudev y tengo 39 años, mido 1.70 metros"

### Completa aquí

print("--------------")

print("\nEjercicio 5: Números")
print("1. Crea una variable con el número PI (sin asignar una
variable)")
print("2. Redondea el número con round()")
print("3. Haz la división entera entre el número que te salió y el
número 2")
print("4. El resultado debería ser 1")

# --- EJERCICIO INTERACTIVO ---
# 1. Ejecuta el código tal como está para ver el resultado.
# 2. Modifica el código para cambiar el comportamiento.
#    (Pista: Intenta cambiar los valores de las variables o la lógica)
# 3. Vuelve a ejecutar para ver tus cambios.
```

Condicionales (If)

Estudia el siguiente código: Condicionales (If)

Código de Ejemplo

```
###
# 01 - Sentencias condicionales (if, elif, else)
# Permiten ejecutar bloques de código solo si se cumplen ciertas
condiciones.
###



print("\n Sentencia simple condicional")
# Podemos usar la palabra clave "if" para ejecutar un bloque de código
# solo si se cumple una condición.
edad = 18
if edad >= 18:
  print("Eres mayor de edad")
  print("¡Felicidades!")

# Si no se cumple la condición, no se ejecuta el bloque de código
edad = 15
if edad >= 18:
  print("Eres mayor de edad")
  print("¡Felicidades!")

# Podemos usar el comando "else" para ejecutar un bloque de código
# si no se cumple la condición anterior del if
print("\n Sentencia condicional con else")
edad = 15
if edad >= 18:
  print("Eres mayor de edad")
else:
  print("Eres menor de edad")

print("\n Sentencia condicional con elif")
nota = 5

# Además de usar "if" y "else", podemos usar "elif" para determinar
# múltiples condiciones, ten en cuenta que sólo se ejecutará el primer
bloque
# de código que cumpla la condición (o la del else, si está presente)
if nota >= 9:
```

```
  print("¡Sobresaliente!")
elif nota >= 7:
  print("Notable!")
elif nota >= 5:
  print("¡Aprobado!")
else:
  print("¡No está calificado!")

print("\n Condiciones múltiples")
edad = 16
tiene_carnet = True

# Los operadores lógicos en Python son:
# and: True si ambos operandos son verdaderos
# or: True si al menos uno de los operandos es verdadero
# En JavaScript:
# && sería and
# || sería or

# En el caso que seas mayor de edad y tengas carnet...
# entonces podrás conducir
if edad >= 18 and tiene_carnet:
  print("Puedes conducir 🚗")
else:
  print("POLICIA 🚔!!!1!!!")

# En un pueblo de Isla Margarita son más laxos y
# te dejan conducir si eres mayor de edad O tienes carnet
if edad >= 18 or tiene_carnet:
  print("Puedes conducir en la Isla Margarita 🚗")
else:
  print("Paga al policía y te deja conducir!!!")

# También tenemos el operador lógico "not"
# que nos permite negar una condición
es_fin_de_semana = False
# JavaScript -> !
if not es_fin_de_semana:
  print("¡midu, venga que hay que streamear!")

# Podemos anidar condicionales, uno dentro del otro
# para determinar múltiples condiciones aunque
```

```
# siempre intentaremos evitar esto para simplificar
print("\n Anidar condicionales")
edad = 20
tiene_dinero = True

if edad >= 18:
  if tiene_dinero:
    print("Puedes ir a la discoteca")
  else:
    print("Quédate en casa")
else:
  print("No puedes entrar a la disco")

# Más fácil sería:
# if edad < 18:
#   print("No puedes entrar a la disco")
# elif tiene_dinero:
#   print("Puedes ir a la discoteca")
# else:
#   print("Quédate en casa")

# Ten en cuenta que hay valores que al usarlos como condiciones
# en Python son evaluados como verdaderos o falsos
# por ejemplo, el número 5, es True
numero = 5
if numero: # True
  print("El número no es cero")

# Pero el número 0 se evalúa como False
numero = 0
if numero: # False
  print("Aquí no entrará nunca")

# También el valor vacío "" se evalúa como False
nombre = ""
if nombre:
  print("El nombre no es vacío")

# ¡Ten cuidado con no confundir la asignación = con la comparación ==!
numero = 3 # asignación
es_el_tres = numero == 3 # comparación
```

```
if es_el_tres:
  print("El número es 3")

# A veces podemos crear condicionales en una sola línea usando
# las ternarias, es una forma concisa de un if-else en una línea de
código
print("\nLa condición ternaria:")
# [código si cumple la condición] if [condicion] else [codigo si no
cumple]

edad = 17
mensaje = "Es mayor de edad" if edad >= 18 else "Es menor de edad"
print(mensaje)

###
# EJERCICIOS
###

# Ejercicio 1: Determinar el mayor de dos números
# Pide al usuario que introduzca dos números y muestra un mensaje
# indicando cuál es mayor o si son iguales

# Ejercicio 2: Calculadora simple
# Pide al usuario dos números y una operación (+, -, *, /)
# Realiza la operación y muestra el resultado (maneja la división
entre zero)

# Ejercicio 3: Año bisiesto
# Pide al usuario que introduzca un año y determina si es bisiesto.
# Un año es bisiesto si es divisible por 4, excepto si es divisible
por 100 pero no por 400.

# Ejercicio 4: Categorizar edades
# Pide al usuario que introduzca una edad y la clasifique en:
# - Bebé (0-2 años)
# - Niño (3-12 años)
# - Adolescente (13-17 años)
# - Adulto (18-64 años)
# - Adulto mayor (65 años o más)

# --- EJERCICIO INTERACTIVO ---
# 1. Ejecuta el código tal como está para ver el resultado.
```

```
# 2. Modifica el código para cambiar el comportamiento.
#    (Pista: Intenta cambiar los valores de las variables o la lógica)
# 3. Vuelve a ejecutar para ver tus cambios.
```

Booleanos

Estudia el siguiente código: Booleanos

Código de Ejemplo

```
###
# 02 - Booleanos
# Valores lógicos: True (verdadero) y False (falso).
# Fundamentales para el control de flujo y la lógica en programación.
###



# Los booleanos representan valores de verdad: True o False.
print("\nValores booleanos básicos:")
print(True)
print(False)

# Operadores de comparación: devuelven un valor booleano.
print("\nOperadores de comparación:")
print("5 > 3:", 5 > 3)        # True
print("5 < 3:", 5 < 3)        # False
print("5 == 5:", 5 == 5)      # True (igualdad)
print("5 != 3:", 5 != 3)      # True (desigualdad)
print("5 >= 5:", 5 >= 5)      # True (mayor o igual que)
print("5 <= 3:", 5 <= 3)      # False (menor o igual que)

print("\nComparación de cadenas:")
print("'manzana' < 'pera':", "manzana" < "pera") # True
print("'Hola' == 'hola'", "Hola" == "hola") # False

# Operadores lógicos: and, or, not
```

```
print("\nOperadores lógicos:")
print("True and True:", True and True)   # True
print("True and False:", True and False)  # False
print("True or False:", True or False)    # True
print("False or False:", False or False)  # False
print("not True:", not True)             # False
print("not False:", not False)            # True

# Tablas de verdad (para referencia):
print("\nTablas de verdad:")
print("\nand:")
print("A     B     A and B")
print("True  True ", True and True)
print("True  False", True and False)
print("False True ", False and True)
print("False False", False and False)

print("\n or:")
print("A     B     A or B")
print("True  True ", True or True)
print("True  False", True or False)
print("False True ", False or True)
print("False False", False or False)

print("\n not:")
print("A     not A")
print("True ", not True)
print("False", not False)

# --- EJERCICIO INTERACTIVO ---
# 1. Ejecuta el código tal como está para ver el resultado.
# 2. Modifica el código para cambiar el comportamiento.
#    (Pista: Intenta cambiar los valores de las variables o la lógica)
# 3. Vuelve a ejecutar para ver tus cambios.
```

Listas

Estudia el siguiente código: Listas

Código de Ejemplo

```
###
# 03 - Listas
# Secuencias mutables de elementos.
# Pueden contener elementos de diferentes tipos.
###



# Creación de listas
print("\nCrear listas")
lista1 = [1, 2, 3, 4, 5] # lista de enteros
lista2 = ["manzanas", "peras", "plátanos"] # lista de cadenas
lista3 = [1, "hola", 3.14, True] # lista de tipos mixtos

lista_vacia = []
lista_de_listas = [[1, 2], ['calcetin', 4]]
matrix = [[1, 2], [2, 3], [4, 5]]

print(lista1)
print(lista2)
print(lista3)
print(lista_vacia)
print(lista_de_listas)
print(matrix)

# Acceso a elementos por índice
print("\nAcceso a elementos por índice")
print(lista2[0])  # manzanas
print(lista2[1])  # peras
print(lista2[-1]) # plátanos
print(lista2[-2]) # peras

print(lista_de_listas[1][0])

# Slicing (rebanado) de listas
lista1 = [1, 2, 3, 4, 5]
print(lista1[1:4]) # [2, 3, 4]
print(lista1[:3]) # [1, 2, 3]
print(lista1[3:]) # [4, 5]
```

```
print(lista1[:]) # [1, 2, 3, 4, 5]

# El tercer parámetro es el paso (step)
lista1 = [1, 2, 3, 4, 5, 6, 7, 8]
print(lista1[::2]) # para devolver índices pares
print(lista1[::-1]) # para devolver índices inversos

# Modificar una lista
lista1[0] = 20
print(lista1)

# Añadir elementos a una lista
lista1 = [1, 2, 3]

# forma larga y menos eficiente
lista1 = lista1 + [4, 5, 6]
print(lista1)

# forma corta y más eficiente
lista1 += [7, 8, 9]
print(lista1)

# Recuperar longitud de una lista
print("Longitud de la lista", len(lista1))

###
# EJERCICIOS
###

# Ejercicio 1: El mensaje secreto
# Dada la siguiente lista:
# mensaje = ["C", "o", "d", "i", "g", "o", " ", "s", "e", "c", "r",
"e", "t", "o"]
# Utilizando slicing y concatenación, crea una nueva lista que
contenga solo el mensaje "secreto".

# Ejercicio 2: Intercambio de posiciones
# Dada la siguiente lista:
# numeros = [10, 20, 30, 40, 50]
# Intercambia la primera y la última posición utilizando solo
asignación por índice.
```

```
# Ejercicio 3: El sándwich de listas
# Dadas las siguientes listas:
# pan = ["pan arriba"]
# ingredientes = ["jamón", "queso", "tomate"]
# pan_abajo = ["pan abajo"]
# Crea una lista llamada sandwich que contenga el pan de arriba, los
ingredientes y el pan de abajo, en ese orden.

# Ejercicio 4: Duplicando la lista
# Dada una lista:
# lista = [1, 2, 3]
# Crea una nueva lista que contenga los elementos de la lista original
duplicados.
# Ejemplo: [1, 2, 3] -> [1, 2, 3, 1, 2, 3]

# Ejercicio 5: Extrayendo el centro
# Dada una lista con un número impar de elementos, extrae el elemento
que se encuentra en el centro de la lista utilizando slicing.
# Ejemplo: lista = [10, 20, 30, 40, 50] -> El centro es 30

# Ejercicio 6: Reversa parcial
# Dada una lista, invierte solo la primera mitad de la lista
(utilizando slicing y concatenación).
# Ejemplo: lista = [1, 2, 3, 4, 5, 6] -> Resultado: [3, 2, 1, 4, 5, 6]


# --- EJERCICIO INTERACTIVO ---
# 1. Ejecuta el código tal como está para ver el resultado.
# 2. Modifica el código para cambiar el comportamiento.
#    (Pista: Intenta cambiar los valores de las variables o la lógica)
# 3. Vuelve a ejecutar para ver tus cambios.
```

Métodos de Listas

Estudia el siguiente código: Métodos de Listas

Código de Ejemplo

```
###
# 04 - Listas Métodos
# Los métodos más importantes para trabajar con listas
###



# Creamos una lista con valores
lista1 = ['a', 'b', 'c', 'd']

# Añadir o insertar elementos a la lista
lista1.append('e') # Añade un elemento al final
print(lista1)

lista1.insert(1, '@') # Inserta un elemento en la posición que le
indiquemos como primer argumento
print(lista1)

lista1.extend(['😃', '😍']) # Agrega elementos al final de la lista
print(lista1)

# Eliminar elementos de la lista
lista1.remove('@') # Eliminar la primera aparición de la cadena de
texto @
print(lista1)

ultimo = lista1.pop() # Eliminar el último elemento de la lista y
además te lo devuelve
print(ultimo)
print(lista1)

lista1.pop(1) # Eliminar el segundo elemento de la lista (es el índice
1)
print(lista1)

# Eliminar por lo bestia un índice
del lista1[-1]
print(lista1)
```

```
lista1.clear() # Eliminar todos los elementos de la lista
print(lista1)

# Eliminar un rango de elementos
lista1 = ['🐼', '🐨', '🐶', '😿', '🐹']
del lista1[1:3] # eliminamos los elementos del índice 1 al 3 (no
incluye el índice 3)
print(lista1)

# Más métodos útiles
print('Ordenar listas modificando la original')
numbers = [3, 10, 2, 8, 99, 101]
numbers.sort()
print(numbers)

print('Ordenar listas creando una nueva lista')
numbers = [3, 10, 2, 8, 99, 101]
sorted_numbers = sorted(numbers)
print(sorted_numbers)

print("Ordenar una lista de cadenas de texto (todo minúscula)")
frutas = ['manzana', 'pera', 'limón', 'manzana', 'pera', 'limón']
sorted_frutas = sorted(frutas)
print(sorted_frutas)

print("Ordenar una lista de cadenas de texto (mezclas mayúscula y
minúscula)")
frutas = ['manzana', 'Pera', 'Limón', 'manzana', 'pera', 'limón']
frutas.sort(key=str.lower)
print(frutas)

# Más cositas útiles
animals = ['🐶', '🐼', '🐨', '🐶']
print(len(animals)) # Tamaño de la listas -> 4
print(animals.count('🐶')) # Cuantas veces aparece el elemento '🐶' ->
2
print('🐼' in animals) # Comprueba si hay un '🐼' en la lista -> True
print('🐹' in animals) # -> False

###
# EJERCICIOS
# Usa siempre que puedas los métodos que has aprendido
```

```
###

# Ejercicio 1: Añadir y modificar elementos
# Crea una lista con los números del 1 al 5.
# Añade el número 6 al final usando append().
# Inserta el número 10 en la posición 2 usando insert().
# Modifica el primer elemento de la lista para que sea 0.

# Ejercicio 2: Combinar y limpiar listas
# Crea dos listas:
# lista_a = [1, 2, 3]
# lista_b = [4, 5, 6, 1, 2]
# Extiende lista_a con lista_b usando extend().
# Elimina la primera aparición del número 1 en lista_a usando
remove().
# Elimina el elemento en el índice 3 de lista_a usando pop(). Imprime
el elemento eliminado.
# Limpia completamente lista_b usando clear().

# Ejercicio 3: Slicing y eliminación con del
# Crea una lista con los números del 1 al 10.
# Utiliza slicing y del para eliminar los elementos desde el índice 2
hasta el 5 (sin incluir el 5).
# Imprime la lista resultante.

# Ejercicio 4: Ordenar y contar
# Crea una lista con los siguientes números: [5, 2, 8, 1, 9, 4, 2].
# Ordena la lista de forma ascendente usando sort().
# Cuenta cuántas veces aparece el número 2 en la lista usando count().
# Comprueba si el número 7 está en la lista usando in.

# Ejercicio 5: Copia vs. Referencia
# Crea una lista llamada original con los números [1, 2, 3].
# Crea una copia de la lista original llamada copia_1 usando slicing.
# Crea otra copia llamada copia_2 usando copy().
# Crea una referencia a la lista original llamada referencia.
# Modifica el primer elemento de la lista referencia a 10.
# Imprime las cuatro listas (original, copia_1, copia_2, referencia) y
observa los cambios.

# Ejercicio 6: Ordenar strings sin diferenciar mayúsculas y
minúsculas.
```

```
# Crea una lista con las siguientes cadenas: ["Manzana", "pera",
"BANANA", "naranja"].
# Ordena la lista sin diferenciar entre mayúsculas y minúsculas.

# --- EJERCICIO INTERACTIVO ---
# 1. Ejecuta el código tal como está para ver el resultado.
# 2. Modifica el código para cambiar el comportamiento.
#    (Pista: Intenta cambiar los valores de las variables o la lógica)
# 3. Vuelve a ejecutar para ver tus cambios.
```

Bucle While

Estudia el siguiente código: Bucle While

Código de Ejemplo

```
###
# 01 - Bucles (while)
# Permiten ejecutar un bloque de código repetidamente mientras se
cumpla una condición
###



print("\n Bucle while:")

# Bucle con una simple condición
contador = 0

while contador <= 5:
  print(contador)
  contador += 1 # es super importante para evitar un bucle infinito

# utilizando la palabra break, para romper el bucle
print("\n Bucle while con break:")
contador = 0
```

```
while True:
  print(contador)
  contador += 1
  if contador == 5:
    break # sale del bucle

# continue, que lo hace es saltar esa iteración en concreto
# y continuar con el bucle
print("\n Bucle con continue")
contador = 0
while contador < 10:
  contador += 1

  if contador % 2 == 0:
    continue

  print(contador)

# else, esta condición cuando se ejecuta?
print("\n Bucle while con else:")
contador = 0
while contador < 5:
  print(contador)
  contador += 1
else:
  print("El bucle ha terminado")

# else, esta condición cuando se ejecuta?
print("\n Bucle while con else:")
contador = 0
while contador < 5:
  print(contador)
  contador += 1
else:
  print("El bucle ha terminado")

# pedirle al usuario un número que tiene
# que ser positivo si no, no le dejamos en paz
numero = -1
# while numero < 0:
#   numero = int(input("Escribe un número positivo: "))
#   if numero < 0:
```

```
#     print("El número debe ser positivo. Intenta otra vez, majo o
maja.")
numero = 10 # Valor hardcodeado para evitar bucle infinito
print(f"El número que has introducido es {numero}")

numero = -1
# while numero < 0:
#   try:
#     numero = int(input("Escribe un número positivo: "))
#     if numero < 0:
#       print("El número debe ser positivo. Intenta otra vez, majo o
maja.")
#   except:
#     print("Lo que introduces debe ser un número, que si no peta!")
numero = 5 # Valor hardcodeado
print(f"El número que has introducido es {numero}")

###
# EJERCICIOS (while)
###

# Ejercicio 1: Cuenta atrás
# Imprime los números del 10 al 1 usando un bucle while.
print("\nEjercicio 1:")

# Ejercicio 2: Suma de números pares (while)
# Calcula la suma de los números pares entre 1 y 20 (inclusive) usando
un bucle while.
print("\nEjercicio 2:")

# Ejercicio 3: Factorial de un número
# Pide al usuario que introduzca un número entero positivo.
# Calcula su factorial usando un bucle while.
# El factorial de un número entero positivo es el producto de todos
los números del 1 al ese número. Por ejemplo, el factorial de 5
# 5! = 5 x 4 x 3 x 2 x 1 = 120.
print("\nEjercicio 3:")

# Ejercicio 4: Validación de contraseña
# Pide al usuario que introduzca una contraseña.
# La contraseña debe tener al menos 8 caracteres.
# Usa un bucle while para seguir pidiendo la contraseña hasta que
```

```
cumpla con los requisitos.
# Si la contraseña es válida, imprime "Contraseña válida".
print("\nEjercicio 4:")

# Ejercicio 5: Tabla de multiplicar
# Pide al usuario que introduzca un número.
# Imprime la tabla de multiplicar de ese número (del 1 al 10) usando
un bucle while.
print("\nEjercicio 5:")

# Ejercicio 6: Números primos hasta N
# Pide al usuario que introduzca un número entero positivo N.
# Imprime todos los números primos menores o iguales que N usando un
bucle while.
print("\nEjercicio 6:")

# --- EJERCICIO INTERACTIVO ---
# 1. Ejecuta el código tal como está para ver el resultado.
# 2. Modifica el código para cambiar el comportamiento.
#    (Pista: Intenta cambiar los valores de las variables o la lógica)
# 3. Vuelve a ejecutar para ver tus cambios.
```

Bucle For

Estudia el siguiente código: Bucle For

Código de Ejemplo

```
###
# 02 - Bucles (for)
# Permiten ejecutar un bloque de código repetidamente mientras ITERA
un iterable o una lista
###



print("\nBucle for:")
```

```
# Iterar una lista
frutas = ["manzana", "pera", "mandarina"]
for fruta in frutas:
  print(fruta)

# Iterar sobre cualquier cosa que sea iterable
cadena = "midudev"
for caracter in cadena:
  print(caracter)

# enumerate()
frutas = ["manzana", "pera", "mandarina"]
for idx, value in enumerate(frutas):
  print(f"El índice es {idx} y la fruta es {value}")

# bucles anidados
letras = ["A", "B", "C"]
numeros = [1, 2, 3]

for letra in letras:
  for numero in numeros:
    print(f"{letra}{numero}")



# break
print("\nbreak:")
animales = ["perro", "gato", "raton", "loro", "pez", "canario"]
for idx, animal in enumerate(animales):
  print(animal)
  if animal == "loro":
    print(f"El loro está escondido en el índice {idx}")
    break

# continue
print("\ncontinue:")
animales = ["perro", "gato", "raton", "loro", "pez", "canario"]
for idx, animal in enumerate(animales):
  if animal == "loro": continue

  print(animal)
```

```
# Comprensión de listas (list comprehension)
animales = ["perro", "gato", "raton", "loro", "pez", "canario"]
animales_mayus = [animal.upper() for animal in animales]
print(animales_mayus)

# Muestra los números pares de una lista
pares = [num for num in [1, 2, 3, 4, 5, 6] if num % 2 == 0]
print(pares)

###
# EJERCICIOS (for)
###

# Ejercicio 1: Imprimir números pares
# Imprime todos los números pares del 2 al 20 (inclusive) usando un
bucle for.
print("\nEjercicio 1:")

# Ejercicio 2: Calcular la media de una lista
# Dada la siguiente lista de números:
# numeros = [10, 20, 30, 40, 50]
# Calcula la media de los números usando un bucle for.
print("\nEjercicio 2:")

# Ejercicio 3: Buscar el máximo de una lista
# Dada la siguiente lista de números:
# numeros = [15, 5, 25, 10, 20]
# Encuentra el número máximo en la lista usando un bucle for.
print("\nEjercicio 3:")

# Ejercicio 4: Filtrar cadenas por longitud
# Dada la siguiente lista de palabras:
# palabras = ["casa", "arbol", "sol", "elefante", "luna"]
# Crea una nueva lista que contenga solo las palabras con más de 5
letras
# usando un bucle for y list comprehension.
print("\nEjercicio 4:")

# Ejercicio 5: Contar palabras que empiezan con una letra
# Dada la siguiente lista de palabras:
# palabras = ["casa", "arbol", "sol", "elefante", "luna", "coche"]
# Pide al usuario que introduzca una letra.
```

```
# Cuenta cuántas palabras en la lista empiezan con esa letra (sin
diferenciar mayúsculas/minúsculas).
print("\nEjercicio 5:")

# --- EJERCICIO INTERACTIVO ---
# 1. Ejecuta el código tal como está para ver el resultado.
# 2. Modifica el código para cambiar el comportamiento.
#    (Pista: Intenta cambiar los valores de las variables o la lógica)
# 3. Vuelve a ejecutar para ver tus cambios.
```

Rango (Range)

Estudia el siguiente código: Rango (Range)

Código de Ejemplo

```
###
# 03 - range()
# Permite crear una secuencia de números. Puede ser útil para for,
pero no solo para eso
###



print("\nrange():")

# Generado una secuencia de números del 0 al 9
for num in range(10):
  print(num)

# range(inicio, fin)
for num in range(5, 10):
  print(num)

# range(inicio, fin, paso)
for num in range(0, 1000, 5):
  print(num)
```

```
for num in range(-5, 0):
  print(num)

for num in range(10, 0, -1):
  print(num)

for num in range(0, 444):
  print(num)

nums = range(10)
list_of_nums = list(nums)
print(list_of_nums)

# seria para hacerlo cinco veces
for _ in range(5):
  print("hacer cinco veces algo")

###
# EJERCICIOS (range)
###

# Ejercicio 1: Imprimir números del 1 al 10
# Imprime los números del 1 al 10 (inclusive) usando un bucle for y
range().
print("\nEjercicio 1:")

# Ejercicio 2: Imprimir números impares del 1 al 20
# Imprime todos los números impares entre 1 y 20 (inclusive) usando un
bucle for y range().
print("\nEjercicio 2:")

# Ejercicio 3: Imprimir múltiplos de 5
# Imprime los múltiplos de 5 desde 5 hasta 50 (inclusive) usando un
bucle for y range().
print("\nEjercicio 3:")

# Ejercicio 4: Imprimir números en orden inverso
# Imprime los números del 10 al 1 (inclusive) en orden inverso usando
un bucle for y range().
print("\nEjercicio 4:")
```

```
# Ejercicio 5: Suma de números en un rango
# Calcula la suma de los números del 1 al 100 (inclusive) usando un
bucle for y range().
print("\nEjercicio 5:")

# Ejercicio 6: Tabla de multiplicar
# Pide al usuario que introduzca un número.
# Imprime la tabla de multiplicar de ese número (del 1 al 10) usando
un bucle for y range().
print("\nEjercicio 6:")

# --- EJERCICIO INTERACTIVO ---
# 1. Ejecuta el código tal como está para ver el resultado.
# 2. Modifica el código para cambiar el comportamiento.
#    (Pista: Intenta cambiar los valores de las variables o la lógica)
# 3. Vuelve a ejecutar para ver tus cambios.
```

Funciones

Estudia el siguiente código: Funciones

Código de Ejemplo

```
###
# 04 - Funciones
# Bloques de código reutilizables y parametrizables para hacer tareas
especificas
###



# """ Definición de una función

# def nombre_de_la_funcion(parametro1, parametro2, ...):
#   # docstring
#   # cuerpo de la función
#   return valor_de_retorno # opcional
```

```
# """

# # Ejemplo de una función para imprimir algo en consola
# def saludar():
#   print("¡Hola!")

# # Ejemplo de una función con parámetro
# def saludar_a(nombre):
#   print(f"¡Hola {nombre}!")

# saludar_a("midudev")
# saludar_a("madeval")
# saludar_a("pheralb")
# saludar_a("felixicaza")
# saludar_a("Carmen Ansio")

# # Funciones con más parámetros
# def sumar(a, b):
#   suma = a + b
#   return suma

# result = sumar(2, 3)
# print(result)

# # Documentar las funciones con docstring
# def restar(a, b):
#   """Resta dos números y devuelve el resultado"""
#   return a - b

# parámetros por defecto
# def multiplicar(a, b = 2):
#   return a * b

# print(multiplicar(2))
# print(multiplicar(2, 3))

# Argumentos por posición
def describir_persona(nombre: str, edad: int, sexo: str):
  print(f"Soy {nombre}, tengo {edad} años y me identifico como
{sexo}")
```

```
# parámetros son posicionales
describir_persona(1, 25, "gato")
describir_persona("midudev", 25, "gato")
describir_persona("hombre", "madeval", 39)

# Argumentos por clave
# parámetros nombrados
describir_persona(sexo="gato", nombre="midudev", edad=25)
describir_persona(sexo="hombre", nombre="madeval", edad=21)

# Argumentos de longitud de variable (*args):
def sumar_numeros(*args):
  suma = 0
  for numero in args:
    suma += numero
  return suma

print(sumar_numeros(1, 2, 3, 4, 5))
print(sumar_numeros(1, 2))
print(sumar_numeros(1, 2,3 ,4, 5, 6, 7, 8, 9, 10))

# Argumentos de clave-valor variable (**kwargs):
def mostrar_informacion_de(**kwargs):
  for clave, valor in kwargs.items():
    print(f"{clave}: {valor}")

mostrar_informacion_de(nombre="midudev", edad=25, sexo="gato")
print("\n")
mostrar_informacion_de(name="madeval", edad=21, country="Uruguay")
print("\n")
mostrar_informacion_de(nick="pheralb", es_sub=True, is_rich=True)
print("\n")
mostrar_informacion_de(super_name="felixicaza", es_modo=True,
gatos=40)

# Ejercicios
# Volver a los ejercicios anteriores
# y convertirlos en funciones
# e intentar utilizar todos los casos y conceptos
# que hemos visto hasta ahora

# --- EJERCICIO INTERACTIVO ---
```

```
# 1. Ejecuta el código tal como está para ver el resultado.
# 2. Modifica el código para cambiar el comportamiento.
#    (Pista: Intenta cambiar los valores de las variables o la lógica)
# 3. Vuelve a ejecutar para ver tus cambios.
```

Reto: Los 4 Fantásticos

Estudia el siguiente código: Reto: Los 4 Fantásticos

Código de Ejemplo

```
"""
¿Está en Equilibrio la Alianza entre Reed Richards y Johnny Storm?

Objetivo:
Crea una función en Python que reciba una cadena de texto. Esta
función debe contar cuántas veces aparece la letra R (para Reed
Richards) y cuántas veces aparece la letra J (para Johnny Storm) en la
cadena.

- Si la cantidad de R y la cantidad de J son iguales, retorna True.
- Si no, retorna False.
- Si no hay ni R ni J, retorna True.
"""

def check_is_balanced(text):
    # --- TU CÓDIGO AQUÍ ---
    # 1. Convierte el texto a mayúsculas para facilitar el conteo
    # 2. Cuenta las 'R' y las 'J'
    # 3. Compara y retorna el resultado
    pass

# Casos de prueba
print(check_is_balanced("RRJJ"))       # Debería ser True
print(check_is_balanced("RRRRJJ"))     # Debería ser False
print(check_is_balanced("RRJJJJJJ"))   # Debería ser False
print(check_is_balanced("awwwaqAQAQA")) # Debería ser True (0 == 0)
```

```
# --- EJERCICIO INTERACTIVO ---
# 1. Implementa la función check_is_balanced.
# 2. Ejecuta el código para verificar tus resultados.
```

Reto: Jurassic Park

Estudia el siguiente código: Reto: Jurassic Park

Código de Ejemplo

```
"""
Objetivo:
Escribe una función en Python que reciba una lista de números enteros
y devuelva la suma total de los huevos que pertenecen a los
dinosaurios carnívoros (es decir, la suma de todos los números pares
en la lista).
"""

def count_carnivore_dinosaur_eggs(egg_list) -> int:
  """
  Devuelve la suma de los números pares en la lista.
  """
  total_carnivore_eggs = 0

  # --- TU CÓDIGO AQUÍ ---
  # Itera sobre la lista y suma solo los números pares

  return total_carnivore_eggs

egg_list = [3, 4, 7, 5, 8]
print(f"Lista de huevos: {egg_list}")
print(f"Total huevos carnívoros (pares):
{count_carnivore_dinosaur_eggs(egg_list)}") # Debería ser 12

# --- EJERCICIO INTERACTIVO ---
```

```
# 1. Implementa la lógica dentro de la función.
# 2. Ejecuta para verificar.
```

Reto: Primera Suma

Estudia el siguiente código: Reto: Primera Suma

Código de Ejemplo

```
"""
Dado un array de números y un número goal, encuentra los dos primeros
números del array que sumen el número goal y devuelve sus índices. Si
no existe tal combinación, devuelve None.

nums = [4, 5, 6, 2]
goal = 8

find_first_sum(nums, goal)  # [2, 3] (porque 6 + 2 = 8)
"""

def find_first_sum(nums, goal):
  # --- TU CÓDIGO AQUÍ ---
  # Encuentra dos números que sumen 'goal'
  # Devuelve una lista con sus índices [i, j]
  pass

nums = [4, 5, 6, 2]
goal = 8
result = find_first_sum(nums, goal)
print(f"Resultado: {result}") # Debería ser [2, 3]

# --- EJERCICIO INTERACTIVO ---
# 1. Implementa la función.
# 2. Pista: Puedes usar dos bucles anidados (fuerza bruta) o un
diccionario para hacerlo más eficiente.
```

Diccionarios

Estudia el siguiente código: Diccionarios

Código de Ejemplo

```
###
# 04 - Dictionaries
# Los diccionarios son colecciones de pares clave-valor.
# Sirven para almacenar datos relacionados.
###



# ejemplo tipico de diccionario
persona = {
  "nombre": "midudev",
  "edad": 25,
  "es_estudiante": True,
  "calificaciones": [7, 8, 9],
  "socials": {
    "twitter": "@midudev",
    "instagram": "@midudev",
    "facebook": "midudev"
  }
}

# para acceder a los valores
print(persona["nombre"])
print(persona["calificaciones"][2])
print(persona["socials"]["twitter"])

# cambiar valores al acceder
persona["nombre"] = "madeval"
persona["calificaciones"][2] = 10

# eliminar completamente una propiedad
del persona["edad"]
# print(persona)
```

```
es_estudiante = persona.pop("es_estudiante")
print(f"es_estudiante: {es_estudiante}")
print(persona)

# sobreescribir un diccionario con otro diccionario
a = { "name": "miduev", "age": 25 }
b = { "name": "madeval", "es_estudiante": True }

a.update(b)
print(a)

# comprobar si existe una propiedad
print("name" in persona) # False
print("nombre" in persona) # True

# obtener todas las claves
print("\nkeys:")
print(persona.keys())

# obtener todas los valores
print("\nvalues:")
print(persona.values())

# obtener tanto clave como valor
print("\nitems:")
print(persona.items())

for key, value in persona.items():
  print(f"{key}: {value}")

# --- EJERCICIO INTERACTIVO ---
# 1. Ejecuta el código tal como está para ver el resultado.
# 2. Modifica el código para cambiar el comportamiento.
#    (Pista: Intenta cambiar los valores de las variables o la lógica)
# 3. Vuelve a ejecutar para ver tus cambios.
```

Reto: Batalla Pokémon

Estudia el siguiente código: Reto: Batalla Pokémon

Código de Ejemplo

```
"""
Simula una batalla entre dos listas de números.
- Si lista_a[i] > lista_b[i], suma la diferencia al siguiente de A.
- Si lista_b[i] > lista_a[i], suma la diferencia al siguiente de B.
- Si son iguales, ambos se eliminan.

Devuelve el ganador ("Xa" o "Xb") o "x" si hay empate.
"""

def battle(lista_a, lista_b):
    # --- TU CÓDIGO AQUÍ ---
    # Implementa la lógica de la batalla
    # Pista: Puedes simplificarlo sumando todos los elementos de cada
lista y comparando los totales.
    pass

lista_a = [4, 4, 4]
lista_b = [2, 8, 2]
winner = battle(lista_a, lista_b)
print(f"Ganador: {winner}")

# --- EJERCICIO INTERACTIVO ---
# 1. Implementa la función battle.
```

Módulo 02: Conceptos Fundamentales

Hola Mundo (Repaso)

Estudia el siguiente código: Hola Mundo (Repaso)

Código de Ejemplo

```
#!/usr/bin/python3
""" Primer Programa en Python
    Solo para verificar si podemos ejecutar python3 correctamente
"""

name = "Sanjeev"
print("hello "+name+"\n")


# --- EJERCICIO INTERACTIVO ---
# 1. Ejecuta el código tal como está para ver el resultado.
# 2. Modifica el código para cambiar el comportamiento.
#    (Pista: Intenta cambiar los valores de las variables o la lógica)
# 3. Vuelve a ejecutar para ver tus cambios.
```

Números

Estudia el siguiente código: Números

Código de Ejemplo

```
#!/usr/bin/python # Configurando el shebang

# Este tutorial cubrirá el concepto de tipos numéricos en Python3.x
```

```
# Hay tres tipos numéricos en Python
# 1. Int (Entero)
# 2. Float (Flotante/Decimal)
# 3. Complex (Complejo)

# Int (Entero)
positive_int = 55
negative_int = -1039
zero = 0
print(positive_int)
print(negative_int)
print(zero)
print(type(negative_int))

# Float (Decimal)
positive_float = 1.497
negative_float = -2.9987654
exponent_float = 3e8 # e para indicar la potencia de 10

print(positive_float)
print(negative_float)
print(exponent_float)
print(type(exponent_float))

# Complex (Complejo)
# Número como 3 + 5j donde j representa una parte imaginaria
complex_num = -5 + 2j
print(complex_num)
print(type(complex_num))



# --- EJERCICIO INTERACTIVO ---
# 1. Ejecuta el código tal como está para ver el resultado.
# 2. Modifica el código para cambiar el comportamiento.
#    (Pista: Intenta cambiar los valores de las variables o la lógica)
# 3. Vuelve a ejecutar para ver tus cambios.
```

Manejo de Errores (Try/Except)

Estudia el siguiente código: Manejo de Errores (Try/Except)

Código de Ejemplo

```
import requests as req

base_url="https://github.com/"
username = "deepraj1729"

url = base_url+username

# Petición GET a github para el nombre de usuario
try:
    res = req.get(url)
    if res.status_code == 404:
        print("Error 404. Página no encontrada")
    elif res.status_code == 200:
        print("Estado: OK")

except Exception as e:
    print("No se pudo establecer conexión")
    print(e)

# --- EJERCICIO INTERACTIVO ---
# 1. Ejecuta el código tal como está para ver el resultado.
# 2. Modifica el código para cambiar el comportamiento.
#    (Pista: Intenta cambiar los valores de las variables o la lógica)
# 3. Vuelve a ejecutar para ver tus cambios.
```

Comentarios y Docstrings

Estudia el siguiente código: Comentarios y Docstrings

Código de Ejemplo

```
#!/usr/bin/python3

# Los comentarios son líneas que el intérprete de Python ignora.
# Sirven para explicar el código a otros humanos (o a tu "yo" del
futuro).

# 1. Comentarios de una sola línea (usan #)
variable = 10 # Esto es una variable

# 2. Comentarios multilínea (no existen oficialmente, pero se usan
strings)
"""
Esto es un string multilínea que no se asigna a ninguna variable.
Python lo ignora, por lo que funciona como un comentario de bloque.
"""

# 3. Docstrings (Cadenas de documentación)
# Se usan para documentar funciones, clases y módulos.

def saludar(nombre):
    """
    Esta función recibe un nombre y saluda.

    Parámetros:
        nombre (str): El nombre de la persona.
    """
    print(f"Hola, {nombre}")

# Podemos acceder al docstring usando el atributo __doc__
print(saludar.__doc__)

saludar("Pythonista")

# --- EJERCICIO INTERACTIVO ---
# 1. Ejecuta el código tal como está para ver el resultado.
# 2. Añade tus propios comentarios al código.
# 3. Modifica el docstring de la función saludar.
```

Conversión Numérica

Estudia el siguiente código: Conversión Numérica

Código de Ejemplo

```
#!/usr/bin/python

# Este tutorial cubrirá el concepto de conversión de tipos numéricos
en Python3.x

# Int
positive_int = 55

# Float
negative_float = -2.9987654

# Complex
complex_num = 1j

# convertir de int a float:
positive_float = float(positive_int)

# convertir de float a int:
negative_int = int(negative_float)

# convertir de int a complex:
complex_from_int = complex(positive_int)

print(positive_float)
print(negative_int)
print(complex_from_int)

print(type(positive_float))
print(type(negative_int))
print(type(complex_from_int))
```

```
# --- EJERCICIO INTERACTIVO ---
# 1. Ejecuta el código tal como está para ver el resultado.
# 2. Modifica el código para cambiar el comportamiento.
#    (Pista: Intenta cambiar los valores de las variables o la lógica)
# 3. Vuelve a ejecutar para ver tus cambios.
```

Operaciones Numéricas Extra

Estudia el siguiente código: Operaciones Numéricas Extra

Código de Ejemplo

```
#!/usr/bin/python
from decimal import Decimal as D

# En este ejemplo, cubriremos
# 1. Cómo funcionan los números float y por qué su comparación te
sorprende
# 2. Cómo usar el formato Decimal como en la escuela usando el módulo
decimal
# 3. Representación binaria, octal y hexadecimal
# 4. Cualquier operación matemática de entero y float resultará en
float

# Los enteros pueden ser de cualquier longitud, un número de punto
flotante es preciso solo hasta 15 decimales
print((1.1 + 2.2) == 3.3)
print(1.1 + 2.2)

# Salida: Decimal('3.3')
print(D('1.1') + D('2.2'))

# Salida: Decimal('3.000')
print(D('1.2') * D('2.50'))

#  ___________________________________
# | Sistema Numérico | Prefijo   |
```

```
# | Binario          | '0b' o '0B' |
# | Octal            | '0o' o '0O' |
# | Hexadecimal      | '0x' o '0X' |
#  ------------------------------

# Salida: 121
print(0b1111001)
print(bin(121))

# Salida: 257 (252 + 5)
print(0xFC + 0b101)
print(hex(252), bin(5))

# Salida: 23
print(0o27)
print(oct(23))

integer_num = 3
float_num = 1.7
sum_result = integer_num + float_num

# Salida: sum_result = 4.7 y clase float
print(sum_result)
print(type(sum_result))


# --- EJERCICIO INTERACTIVO ---
# 1. Ejecuta el código tal como está para ver el resultado.
# 2. Modifica el código para cambiar el comportamiento.
#    (Pista: Intenta cambiar los valores de las variables o la lógica)
# 3. Vuelve a ejecutar para ver tus cambios.
```

Cadenas de Texto (Strings)

Estudia el siguiente código: Cadenas de Texto (Strings)

Código de Ejemplo

```
#!/usr/bin/python

# Python tiene la clase str para representar y manejar cadenas

first_name = "Sanjeev"
last_name  = 'Jaiswal'
nick_name  = '''Jassi'''
address    = """ Dirección de correo, ¿verdad?
    si es así, es Hyderabad, Madhapur.
    Pin: 500081"""

mobile_num = 9618961800

print("Nombre:", first_name)
print("Nombre: " + first_name) # Concatenación de cadenas
print("Dirección multilínea: " + address)

greetings = 'Hola'
print("La longitud de la cadena es " + str(len(greetings))) # len()
para la longitud de la cadena

print(greetings + nick_name)  ## Hola Jassi. Concatenación de
cadenas

pi = 3.14   # text = 'El valor de pi es ' + pi      ## NO, no
funciona
text = 'El valor de pi es '  + str(pi)  ## necesitamos convertir
específicamente el número a tipo str para imprimir


# --- EJERCICIO INTERACTIVO ---
# 1. Ejecuta el código tal como está para ver el resultado.
# 2. Modifica el código para cambiar el comportamiento.
#    (Pista: Intenta cambiar los valores de las variables o la lógica)
# 3. Vuelve a ejecutar para ver tus cambios.
```

Formato de Strings

Estudia el siguiente código: Formato de Strings

Código de Ejemplo

```
#!/usr/bin/python
# Python tiene la clase str para representar y manejar cadenas

first_name = "Sanjeev"
last_name  = 'Jaiswal'
nick_name  = '''Jassi'''
address    = """ Dirección de correo, ¿verdad?
    si es así, es Hyderabad, Madhapur.
    Pin: 500081"""

mobile_num = 9618961800

text = ("Ejemplo del operador " + chr(37) + ":  %d es mi número %s es
mi apodo. Tengo %.2f grandes para %s" % (mobile_num, nick_name, 4.0,
last_name))
print(text)

# Argumentos por posición
print("======== Argumentos por posición ========")
print("Nombre: {}".format(first_name))
print("Nombre: {0}".format(first_name))
print(f"Nombre: {first_name}")
print("Nombre Completo: {} {}".format(first_name, last_name))
print("Nombre Completo: {0} {1}".format(first_name, last_name))
print("Nombre Completo: {1} {0}".format(first_name, last_name))
print(f"Nombre Completo: {first_name} {last_name}")

# Argumentos por parámetro
print("======= Argumentos por parámetro =======")
print("Apodo: {nick_name}".format(nick_name = "Jassi"))

# Salida: 'Coordenadas: 37.24N, -115.81W'
print('Coordenadas: {latitude}, {longitude}'.format(latitude='37.24N',
longitude='-115.81W'))
```

```
full_name = {'first_name': 'Alicia', 'last_name': 'Gearcia'}
# Salida 'Nombre Completo: Gearcia Alicia'
print('Nombre Completo: {last_name}
{first_name}'.format(**full_name))


# --- EJERCICIO INTERACTIVO ---
# 1. Ejecuta el código tal como está para ver el resultado.
# 2. Modifica el código para cambiar el comportamiento.
#    (Pista: Intenta cambiar los valores de las variables o la lógica)
# 3. Vuelve a ejecutar para ver tus cambios.
```

Métodos de Strings

Estudia el siguiente código: Métodos de Strings

Código de Ejemplo

```
#!/usr/bin/python
# Python tiene la clase str para representar y manejar cadenas

first_name = "Sanjeev"
last_name  = 'jaiswal'
nick_name  = '''Jassi'''
address    = """ Dirección de correo, ¿verdad?
    si es así, es Hyderabad, Madhapur.
    Pin: 500081"""

mobile_num = 9618961800

greetings = 'Hola'
print("La longitud de la cadena 'Hola' es: " + str(len(greetings))) #
len() para la longitud de la cadena

# ejemplos de funciones lower(), upper() y capitalize()
## hola Jassi. Concatenación de cadenas
```

```
print(greetings.lower(), nick_name)

# Hagamos los nombres todo en MAYÚSCULAS
print(first_name.upper(), last_name.upper())

# Capitalizar last_name
print(last_name.capitalize())

#-------------------------------------------------------------------
---------------------------------#
# Método                                  | True
(si)                                                          #
# str.isalnum()                           | La cadena consta solo de
caracteres alfanuméricos (sin símbolos) #
# str.isalpha()                           | La cadena consta solo de
caracteres alfabéticos (sin símbolos)   #
# str.islower()                           | Los caracteres alfabéticos
de la cadena están todos en minúsculas            #
# str.isupper()                           | Los caracteres alfabéticos
de la cadena están todos en mayúsculas            #
# str.isnumeric()                         | La cadena consta solo de
caracteres numéricos                   #
#-------------------------------------------------------------------
---------------------------------#

print(str(mobile_num).isnumeric())
print(first_name.isalpha())
print(last_name.isalnum())
print(nick_name.isupper())

# funciones join() y split()
reversed_first_name =''.join(reversed(first_name))
print("Inverso de {} es {}". format(first_name, reversed_first_name))

# Práctica para split(), replace(), strip(), find()
# Revisa nuestras Preguntas de Práctica de Laboratorio



# --- EJERCICIO INTERACTIVO ---
# 1. Ejecuta el código tal como está para ver el resultado.
# 2. Modifica el código para cambiar el comportamiento.
```

```
#    (Pista: Intenta cambiar los valores de las variables o la lógica)
# 3. Vuelve a ejecutar para ver tus cambios.
```

Estructuras Condicionales

Estudia el siguiente código: Estructuras Condicionales

Código de Ejemplo

```
#!/usr/bin/python

# Las sentencias de control son uno de los bloques fundamentales de
Python
# Cubriremos
# 1. if elif else
# 2. while, for, range
# 3. break, continue, pass

# 1. Ejemplo de if, elif, else
# Puede haber cero o más sentencias elif
# else es opcional

# url = input("Introduce tu sitio web: ")
url = "https://cybercloud.guru" # Valor hardcodeado para demo

if 'http' in url:
    print('url no segura')
elif 'https' in url:
    print('algo más segura')
elif url == 'cybercloud.guru':
    print('ajá cybercloud guru encontrado')
elif url == '':
    print('¿Cómo puede estar vacía una url? ¿olvidaste escribir?')
else:
    print('aquí está tu url: ' + url)
```

```
# --- EJERCICIO INTERACTIVO ---
# 1. Ejecuta el código tal como está para ver el resultado.
# 2. Modifica el código para cambiar el comportamiento.
#    (Pista: Intenta cambiar los valores de las variables o la lógica)
# 3. Vuelve a ejecutar para ver tus cambios.
```

Bucle While Avanzado

Estudia el siguiente código: Bucle While Avanzado

Código de Ejemplo

```
#!/usr/bin/python
import random

# Sintaxis del bucle While
# while expresion:
#   sentencia(s)
# Puedes usar el bloque else con while también
# while expresion:
#   sentencia(s)
# else:
#   sentencia(s)

guess_num_range = 20
num_to_be_guessed = int(guess_num_range * random.random()) + 1
guess = 0

# while guess != num_to_be_guessed:
#     guess = int(input("Adivina el número: "))
#     if guess > 0:
#         if guess > num_to_be_guessed:
#             print("El número es demasiado grande")
#         elif guess < num_to_be_guessed:
#             print("El número es demasiado pequeño")
```

```
#     else:
#         print("¡Lamento que te rindas!")
#         break
# else:
#     print("¡Felicitaciones. Lo lograste!")

print(f"Simulando adivinanza... El número era {num_to_be_guessed}")
print("¡Felicitaciones. Lo lograste!")


# --- EJERCICIO INTERACTIVO ---
# 1. Ejecuta el código tal como está para ver el resultado.
# 2. Modifica el código para cambiar el comportamiento.
#    (Pista: Intenta cambiar los valores de las variables o la lógica)
# 3. Vuelve a ejecutar para ver tus cambios.
```

Bucle For Avanzado

Estudia el siguiente código: Bucle For Avanzado

Código de Ejemplo

```
#!/usr/bin/python

# sintaxis del bucle for
# for variable_iteradora in secuencia:
#   sentencia(s)
# Puedes usar el bloque else con el bucle for igual que usamos con
while en el ejemplo anterior
# for <variable> in <secuencia>:
#       <sentencias>
# else:
#       <sentencias>

port_details = {
    '22': 'ssh',
    '21': 'ftp',
```

```
    '23': 'telnet',
    '80': 'http',
    '443': 'https'
}

print("====== Detalles de Puertos ======")
for port in port_details:
    print("{} -> {}".format(port, port_details[port]))

print("============================")




# --- EJERCICIO INTERACTIVO ---
# 1. Ejecuta el código tal como está para ver el resultado.
# 2. Modifica el código para cambiar el comportamiento.
#    (Pista: Intenta cambiar los valores de las variables o la lógica)
# 3. Vuelve a ejecutar para ver tus cambios.
```

Uso de Range

Estudia el siguiente código: Uso de Range

Código de Ejemplo

```
#!/usr/bin/python

# La función integrada de Python range() genera números enteros entre
el entero de inicio dado y el entero de parada, es decir, range()
devuelve un objeto de rango.
# Usando el bucle for, podemos iterar sobre una secuencia de números
producida por la función range().
# Solo permite números de tipo entero como argumentos.
# No podemos proporcionar un parámetro de tipo string o float dentro
de la función range().
# Los argumentos pueden ser positivos (+ve) o negativos (-ve).
```

```
# No acepta ‘0’ como valor de paso. Si el paso es ‘0’, la función
lanza un ValueError.

for step in range(10, 100, 10):
    print(step)


print("\nOtro ejemplo para iterar sobre una lista usando range")
port_lists = [21, 22, 23, 25, 53, 80, 443, 3306, 8080, 9002, 27017]

for port in range(len(port_lists)):
    print(port_lists[port])


# --- EJERCICIO INTERACTIVO ---
# 1. Ejecuta el código tal como está para ver el resultado.
# 2. Modifica el código para cambiar el comportamiento.
#    (Pista: Intenta cambiar los valores de las variables o la lógica)
# 3. Vuelve a ejecutar para ver tus cambios.
```

Control de Bucles (Break/Continue)

Estudia el siguiente código: Control de Bucles (Break/Continue)

Código de Ejemplo

```
#!/usr/bin/python

# Ejemplo de pass, break, continue
# pass: no hace nada. útil cuando se prueba algo o cuando ese bloque
de código no es necesario
# break: termina el bucle actual y reanuda la ejecución en la
siguiente sentencia
# continue: devuelve el control al principio del bucle


port_details = {
```

```
    '21': 'ftp',
    '23': 'telnet',
    '80': 'http',
    '443': 'https',
    '3306': 'mysql'
}

print("====== Detalles de Puertos ======")
for port in port_details:
    if port == '80' or port == '443':
        print("el puerto {} es un puerto web permitido".format(port))
        continue
    elif port == '22':
        print("el acceso ssh parece permitido aquí")
        break
    else:
        pass
        print("Solo pasó y mostrando los detalles del puerto abajo")
        print("{} -> {}".format(port, port_details[port]))
else:
    print("Llegó aquí significa que terminó todo en el bucle for")
print("============================")




# --- EJERCICIO INTERACTIVO ---
# 1. Ejecuta el código tal como está para ver el resultado.
# 2. Modifica el código para cambiar el comportamiento.
#    (Pista: Intenta cambiar los valores de las variables o la lógica)
# 3. Vuelve a ejecutar para ver tus cambios.
```

Condicionales en Una Línea

Estudia el siguiente código: Condicionales en Una Línea

Código de Ejemplo

```
#!/usr/bin/python

# Nota: Esto es opcional pero bueno saberlo.
# En la situación donde solo tenemos un if y un else, y el cuerpo de
cada rama contiene solo una expresión, entonces podemos usar una
expresión condicional. Las expresiones condicionales se pueden usar
para expresar sucintamente un condicional simple

# name = input("¿Cuál es tu primer nombre? ")
name = "Christopher" # Valor hardcodeado para demo

# 1) Llamar a `print` con una cadena diferente usando una sola
expresión condicional
print(
    "Tu nombre es tan largo o más largo que el nombre promedio en los
Estados Unidos"
) if len(name) >= 6 else print (
    "Tu nombre es más corto que el nombre promedio en los Estados
Unidos"
)

# 2) Establecer `message` usando una sola expresión condicional
message = (
    "La primera letra de tu nombre está entre las cinco más comunes"
    if name[0].lower() in ["a", "j", "m", "e", "l"]
    else "La primera letra de tu nombre no está entre las cinco más
comunes"
)

print(message)

# 3) Cambiar la cadena pasada a la función `print` usando una
expresión condicional
for letter in name:
    print(
        f"{letter} {'es una vocal' if letter.lower() in ['a', 'e',
'i', 'o', 'u'] else 'es una consonante'}"
    )
```

```
# --- EJERCICIO INTERACTIVO ---
# 1. Ejecuta el código tal como está para ver el resultado.
# 2. Modifica el código para cambiar el comportamiento.
#    (Pista: Intenta cambiar los valores de las variables o la lógica)
# 3. Vuelve a ejecutar para ver tus cambios.
```

Operadores Aritméticos

Estudia el siguiente código: Operadores Aritméticos

Código de Ejemplo

```
#!/usr/bin/python

x = 21
y = 5

# Salida: x + y = 26
# Suma
print('x + y =',x+y)

# Salida: x - y = 16
# Resta
print('x - y =',x-y)

# Salida: x * y = 105
# Multiplicación
print('x * y =',x*y)

# Salida: x / y = 4.2
# División, siempre resulta en un flotante (decimal)
print('x / y =',x/y)

# Salida x % y = 1
# Módulo (residuo de la división)
```

```
print('x % y =', x%y)

# Salida: x // y = 4
# División Entera (cociente sin decimales)
print('x // y =',x//y)

# Salida: x ** y = 4084101
# Potencia (x elevado a la y)
print('x ** y =',x**y)


# --- EJERCICIO INTERACTIVO ---
# 1. Ejecuta el código tal como está para ver el resultado.
# 2. Modifica el código para cambiar el comportamiento.
#    (Pista: Intenta cambiar los valores de las variables o la lógica)
# 3. Vuelve a ejecutar para ver tus cambios.
```

Operadores de Comparación

Estudia el siguiente código: Operadores de Comparación

Código de Ejemplo

```
#!/usr/bin/python

# Operadores de comparación con números
x = 19
y = 91

print('{} > {} es'.format(x,y),x>y)
print('{} < {} es'.format(x, y),x<y)
print('{} == {} es'.format(x,y),x==y)
print('{} != {} es'.format(x,y),x!=y)
print('{} >= {} es'.format(x,y),x>=y)
print('{} <= {} es'.format(x,y),x<=y)

# Veamos cómo funciona con cadenas de texto (strings)
```

```
name_title = 'Jassi'
name_lowercase = 'jassi'

print('{} > {} es'.format(name_title,
name_lowercase),name_title>name_lowercase)
print('{} < {} es'.format(name_title,
name_lowercase),name_title<name_lowercase)
print('{} == {} es'.format(name_title,
name_lowercase),name_title==name_lowercase)
print('{} != {} es'.format(name_title, name_lowercase),name_title!
=name_lowercase)
print('{} >= {} es'.format(name_title,
name_lowercase),name_title>=name_lowercase)
print('{} <= {} es'.format(name_title,
name_lowercase),name_title<=name_lowercase)


# --- EJERCICIO INTERACTIVO ---
# 1. Ejecuta el código tal como está para ver el resultado.
# 2. Modifica el código para cambiar el comportamiento.
#    (Pista: Intenta cambiar los valores de las variables o la lógica)
# 3. Vuelve a ejecutar para ver tus cambios.
```

Operadores de Asignación

Estudia el siguiente código: Operadores de Asignación

Código de Ejemplo

```
#!/usr/bin/python

# El operador de asignación se usa para asignar un valor a la variable
del lado izquierdo
# Ejemplos de operadores de asignación en Python
# Operador      Ejemplo Equivalente a
# =            x = 5    x = 5
# +=       x += 5       x = x + 5
```

```
# -=       x -= 5       x = x - 5
# *=       x *= 5       x = x * 5
# /=       x /= 5       x = x / 5
# %=       x %= 5       x = x % 5
# //=      x //= 5      x = x // 5
# **=      x **= 5      x = x ** 5
# &=       x &= 5       x = x & 5
# |=       x |= 5       x = x | 5
# ^=       x ^= 5       x = x ^ 5
# >>=     x >>= 5       x = x >> 5
# <<=     x <<= 5       x = x << 5

x = 5
print('x =', x)

x += 5
print('x =', x)

x *= 5
print('x = {}'.format(x))

# Puedes probar con los otros operadores


# --- EJERCICIO INTERACTIVO ---
# 1. Ejecuta el código tal como está para ver el resultado.
# 2. Modifica el código para cambiar el comportamiento.
#    (Pista: Intenta cambiar los valores de las variables o la lógica)
# 3. Vuelve a ejecutar para ver tus cambios.
```

Operadores Lógicos

Estudia el siguiente código: Operadores Lógicos

Código de Ejemplo

```
#!/usr/bin/python

# Operadores Lógicos en Python
# Operador
Significado
Ejemplo
# and      True (Verdadero) si ambos operandos son verdaderos        x
and y
# or       True (Verdadero) si alguno de los operandos es verdadero
x or y
# not      True (Verdadero) si el operando es falso (invierte el
valor)   not x

x = True
y = False

# Salida: x and y es False
print('x and y es',x and y)

# Salida: x or y es True
print('x or y es',x or y)

# Salida: not x es False
print('not x es',not x)


# --- EJERCICIO INTERACTIVO ---
# 1. Ejecuta el código tal como está para ver el resultado.
# 2. Modifica el código para cambiar el comportamiento.
#    (Pista: Intenta cambiar los valores de las variables o la lógica)
# 3. Vuelve a ejecutar para ver tus cambios.
```

Operadores de Identidad

Estudia el siguiente código: Operadores de Identidad

Código de Ejemplo

```
#!/usr/bin/python

# Operadores de Identidad
# 'is' y 'is not' son los operadores de identidad en Python.
# Se usan para verificar si dos valores (o variables) están ubicados en la misma parte de la memoria.
# Que dos variables sean iguales (==) no implica que sean idénticas (is).

# Ejemplo tomado de programiz.com
x1 = 5
y1 = 5
x2 = 'Hola'
y2 = 'Hola'
x3 = [1,2,3]
y3 = [1,2,3]

# Salida: False (porque x1 y y1 SON idénticos, así que 'is not' es falso)
print(x1 is not y1)

# Salida: True (porque x2 y y2 apuntan al mismo string en memoria)
print(x2 is y2)

# Salida: False (porque x3 y y3 son listas diferentes en memoria, aunque tengan el mismo contenido)
print(x3 is y3)

# Aquí vemos que x1 y y1 son enteros con el mismo valor, por lo que son iguales e idénticos (Python optimiza enteros pequeños).
# Lo mismo ocurre con x2 y y2 (strings cortos).
# Pero x3 y y3 son listas. Son iguales (==) pero NO idénticas (is).
# Esto es porque el intérprete las ubica en lugares separados de la memoria aunque sean iguales.



# --- EJERCICIO INTERACTIVO ---
# 1. Ejecuta el código tal como está para ver el resultado.
# 2. Modifica el código para cambiar el comportamiento.
```

```
#    (Pista: Intenta cambiar los valores de las variables o la lógica)
# 3. Vuelve a ejecutar para ver tus cambios.
```

Estructuras de Datos

Estudia el siguiente código: Estructuras de Datos

Código de Ejemplo

```
# No es algo nuevo. En cada lenguaje de programación escucharás sobre sus estructuras de datos.
# Una Estructura de Datos no es más que cómo organizas tus datos, cómo los muestras, trabajas con ellos, etc.
# Python tiene muchas estructuras de datos integradas que ya has visto como números y cadenas.
# Discutiremos las Estructuras de Datos (DS) más importantes que usarás más a menudo en Python.
# Ellas son:
# 1. Listas (Lists)
# 2. Tuplas (Tuples)
# 3. Diccionarios (Dictionaries)
# 4. Conjuntos (Sets)

# Ten en cuenta que hay muchas otras con diferentes bibliotecas de python como array.array pero no las discutiremos en los fundamentos de Python

# La Lista es una estructura de datos mutable y se implementa como un array dinámico.
proto_list = ["http", "https", "ftp", "ssh"] # Puedes tener una lista definida o crear una lista vacía
print(proto_list)

# Tuplas
# Las tuplas son objetos inmutables, por lo demás se ven similares a las listas.
# Inmutable significa que los elementos no se pueden agregar o
```

```
eliminar dinámicamente y todos los elementos en una tupla deben
definirse en el momento de la creación.
proto_tuple = ("http", "https", "ftp", "ssh") # ¿Observaste [] y ()?
[] significa lista y () significa tupla aquí. Bastante interesante,
¿verdad?
print(proto_tuple)

# Diccionarios
# Los diccionarios, en resumen Dict, almacenan un número arbitrario de
objetos, cada uno identificado por una clave única.
# La clave suele ser una cadena y el valor puede ser de cualquier tipo
de dato de Python.
emp_id = {
    "sid": 657387,
    "daniel": 603719,
    "jassi": 770521,
}
print(emp_id)

# Conjuntos (Sets)
# Un conjunto es una colección de objetos que no permiten elementos
duplicados.
# Los conjuntos no están ordenados como los diccionarios, por lo que
no puedes predecir cuál se imprimirá primero.
# Los conjuntos son inmutables en cuanto a sus elementos (hashables),
pero el conjunto en sí es mutable (puedes agregar/quitar).
proto_set = {"tcp", "icmp", "ssh", "icmp", "ftp"} # ¿Observaste cómo
se crea? con {}. Ahora recuerda [], () y {} al crear estas estructuras
de datos.
print(proto_set) # ¡SIN valores duplicados! ;)

# Además, {} significa diccionario vacío. Entonces, ¿qué usar para un
conjunto vacío? (Pista: set())

# --- EJERCICIO INTERACTIVO ---
# 1. Ejecuta el código tal como está para ver el resultado.
# 2. Modifica el código para cambiar el comportamiento.
#    (Pista: Intenta cambiar los valores de las variables o la lógica)
# 3. Vuelve a ejecutar para ver tus cambios.
```

Listas a Fondo

Estudia el siguiente código: Listas a Fondo

Código de Ejemplo

```
#!/usr/bin/python

# Lista Vacía
allowed_ports = []

# lista predefinida
allowed_ports = [22, 23, 25, 53, 80, 69, 443, 3306, 8000, 8080, 5439,
8081, 9001,27017]

# Iterar a través de la lista
print("Iterar a través de la lista")
for port in allowed_ports:
    print(port)

# Verificar si un elemento existe en la lista
print("\nVerificando si el puerto 5432 está presente en la lista
aprobada o no")
if 5432 in allowed_ports:
    print("El puerto predeterminado de Postgres 5432 está permitido")
else:
    print("¡No Aprobado!\n\tNecesitas obtener aprobación del
administrador para habilitar el puerto 5432 para postgres")

# Acceder a la lista
print("\nMostrando algunas formas de acceder al contenido de los
elementos de la lista")
print("Primer elemento de la lista {}".format(allowed_ports[0])) # El
primer índice de la lista es 0
print("Último elemento de la lista {}".format(allowed_ports[-1])) # -1
en la lista muestra el último elemento
print("Rango del 3er al 6to elemento de la lista
{}".format(allowed_ports[2:6])) # La búsqueda comenzará en el índice 2
(incluido) ya que contiene el 3er elemento y terminará en el índice 6
(no incluido) lo que significa el 7mo elemento.
```

```
print("Elementos desde el principio hasta el 8vo elemento, es decir,
puerto 3306 en nuestro caso {}".format(allowed_ports[:8]))
print("Elementos desde el 5to elemento hasta el final
{}".format(allowed_ports[4:]))
print("Veamos si entiendes el slicing aquí
{}".format(allowed_ports[-7:-2]))

# Cambiar el valor de un elemento existente
print("\nCambiar el valor del puerto 69 a 690, verifica el 6to
elemento")
print(allowed_ports)
allowed_ports[5] = 690
print(allowed_ports)

# Manipulación de listas
# función len()
print("El número de puertos permitidos en la lista son: ",
len(allowed_ports))
print("Los puertos permitidos son: ", allowed_ports)

# función count() para contar cuántas entradas de este tipo hay
print("Nº de puerto 80 en la lista: ", allowed_ports.count(80))
print("Nº de puerto 8001 en la lista: ", allowed_ports.count(8001))

# función index() para encontrar el número de índice del contenido
coincidente en la lista
print("Índice del puerto 3306 en la lista: ",
allowed_ports.index(3306))

# Ver la diferencia entre la función reversed() y reverse()
print("\nFormas de invertir la lista")
reverse_port_list = list(reversed(allowed_ports))
print("Inverso de la lista de puertos usando el método reversed(): ",
reverse_port_list)

allowed_ports.reverse()
print("Inverso de la lista de puertos usando el método reverse():",
allowed_ports)

# invirtiéndolo de nuevo para ponerlo en orden ascendente :D
allowed_ports.reverse()
```

```
# función sorted() para ordenar una lista rápidamente
print("\nFormas de ordenar la lista")
sorted_port_list = sorted(allowed_ports)
print("Lista de puertos ordenada usando el método sorted()",
sorted_port_list)

# Añadir/Eliminar elementos
print("\nFormas de añadir elementos")
# añade los elementos al final de la lista
allowed_ports.append(27018)
print(allowed_ports)

# Añadiendo elementos en la 7ma posición, así que el índice es 6
allowed_ports.insert(6,110)
print(allowed_ports)

print("\nFormas de eliminar elementos")
# el método remove() elimina el elemento especificado
allowed_ports.remove(27018) # eliminar puerto 27018
print(allowed_ports)

# el método pop() elimina el índice especificado, (o el último
elemento si no se especifica índice)
allowed_ports.pop(6) # eliminar 7mo índice
print(allowed_ports)

# la palabra clave del elimina en el índice especificado o incluso
toda la lista
del allowed_ports[0]
print(allowed_ports)
# Comentado a propósito `del allowed_ports`

# el método clear vacía la lista
allowed_ports.clear()
print(allowed_ports)

# eliminemos toda la lista ahora.
del allowed_ports
try:
    print(allowed_ports)
except Exception as e:
    print("¿Parece que allowed_ports no existe ahora? Error:", e)
```

```
# Puedes intentar las 2 tareas siguientes
# 1. Unir las listas
# 2. Copiar las listas
# También, intenta entender Shallow copy vs Deep copy




# --- EJERCICIO INTERACTIVO ---
# 1. Ejecuta el código tal como está para ver el resultado.
# 2. Modifica el código para cambiar el comportamiento.
#    (Pista: Intenta cambiar los valores de las variables o la lógica)
# 3. Vuelve a ejecutar para ver tus cambios.
```

Diccionarios a Fondo

Estudia el siguiente código: Diccionarios a Fondo

Código de Ejemplo

```
#!/usr/bin/python3

# Ejemplos para diccionario
# Diccionario Vacío
emp_dict = {}

# diccionario con claves enteras
emp_dict = {100: 'Sanjeev', 101: 'Jassi'}
print(emp_dict)

# diccionario con claves mixtas
emp_dict = {100: 'Sanjeev', 'skills': ['Python', 'AWS']}
print(emp_dict)

# usando la función dict()
emp_dict = dict({100: 'Sanjeev', 101: 'Jassi'})
```

```
print(emp_dict)

# Creemos emp_dict de este formato
'''
{
        'Employee ID':
        {       Name: 'string'
                Joined: 'yyyy-mm-dd'
                Title: 'string'
                Skills: [‘list’, ‘of’, ‘skills’]
                Project: {‘project_name’: ‘project description’}
        }
}
'''

# Inicializando Empleado
emp_dict = {
    100:
        {
            'name': "Sanjeev",
            'joined': "2017-08-14",
            'title': "Cloud Security Engineer",
            'skills': ['Python', 'AWS', 'AppSec'],
            'projects': {
                'CSPM implementation': 'Implement Cloud Security
Posture for AWS'
            }
        },
    101:
        {
            'name': "Jassi",
            'joined': "2017-10-27",
            'title': "Cloud Security Manager",
            'skills': ['Python', 'AWS', 'AWS Security'],
            'projects': {
                'CSPM implementation': 'Implement Cloud Security
Posture for AWS and Azure'
            }
        }
}

print(emp_dict)
```

```
# Obtener el tipo de emp_dict
print(type(emp_dict))

# obtener claves de un diccionario usando keys()
emp_ids = emp_dict.keys()
print(emp_ids)

# Obtener valores de un diccionario usando values()
emp_details = emp_dict.values()
print(emp_details)

# obtener clave y valor ambos usando items()
emps = emp_dict.items()
print(emps)

# Longitud de un diccionario (número de elementos) usando len()
print(len(emp_dict))

# Iterar a través de un diccionario
for id in emp_dict.keys():
    print(f"ID Empleado: {id}")
    print(f"\tDetalles Empleado: {emp_dict[id]}")

# Accediendo a elementos del diccionario
# get vs [] para recuperar elementos
# Sanjeev
print(emp_dict[100]['name'])

# ['Python', 'AWS', 'AppSec']
print(emp_dict[101].get('skills'))

# Intentar acceder a claves que no existen
# None
print(emp_dict[100].get('mailid'))

# KeyError: 'location' Comenta la línea de abajo para ejecutar otras
líneas abajo
# print(emp_dict[101]['location'])

# Añadir un empleado más a emp_dict usando update()
new_emp = {
```

```
    102:
        {
            'name': "Rakesh",
            'joined': "2018-01-07",
            'title': "Business Analyst",
            'skills': ['Power BI', 'MBA', 'Marketing Expert'],
            'projects': {
                'Flexmind Marketing': 'Increase the membership my
targeted marketing'
            }
        }
}

emp_dict.update(new_emp)
print(emp_dict)

# Actualizar el valor existente. Actualizar título de emp_id: 100 como
"Sr. Cloud Security Engineer"
emp_dict[100]['title'] = "Sr. Cloud Security Engineer"
print(emp_dict[100])

# Aprender a eliminar
# pop, clear, del
# Pop empleado id 101
emp_dict.pop(101, "No encontrado")
print(len(emp_dict))

# Eliminar el empleado 102
del emp_dict[102]
print(len(emp_dict))

# Limpiar el diccionario
emp_dict.clear()
print(emp_dict)

# Eliminar el diccionario en sí e imprimirlo lanzaría un error
del emp_dict
# print(emp_dict)
```

```
# --- EJERCICIO INTERACTIVO ---
# 1. Ejecuta el código tal como está para ver el resultado.
# 2. Modifica el código para cambiar el comportamiento.
#    (Pista: Intenta cambiar los valores de las variables o la lógica)
# 3. Vuelve a ejecutar para ver tus cambios.
```

Tuplas

Estudia el siguiente código: Tuplas

Código de Ejemplo

```
# Las tuplas son un objeto inmutable en Python. Significa que una vez
que se establecen los datos, no puedes cambiarlos.
# Se pueden usar para datos constantes o diccionario sin clave (tuplas
anidadas)

# Inicialización de Tupla Vacía
tup = ()
print(tup)

# Inicialización de Tupla con datos
# No me gustó esta forma sin embargo ;)
tup1 = 'python', 'aws', 'security'
print(tup1)

# Otra para hacer lo mismo
tup2 = ('python', 'django', 'linux')
print(tup2)

# Concatenación
tup3 = tup1 + tup2
print(tup3)

# Anidamiento de tuplas
tup4 = (tup1, tup2)
```

```
print(tup4)

# Longitud de una tupla
print(len(tup3))
print(len(tup4))

# Indexación y slicing de Tupla
print(tup3[2])
print(tup2[1:])

# Eliminando una tupla, eliminar un elemento individual de la tupla no
es posible. Elimina toda la tupla
del tup4

# Convierte una lista en tupla
tup5 = tuple(["Sanjeev", '2021', "Flexmind"])
print(tup5)

# prueba tuple() a una cadena
tup6 = tuple('Python')
print(tup6)

# Iteración de Tupla
for tup in tup5:
    print(tup)

# Método Max y min
max_elem = max(tup1)
print("elemento max: ", max_elem)
print("elemento min: ", min(tup5))




# --- EJERCICIO INTERACTIVO ---
# 1. Ejecuta el código tal como está para ver el resultado.
# 2. Modifica el código para cambiar el comportamiento.
#    (Pista: Intenta cambiar los valores de las variables o la lógica)
# 3. Vuelve a ejecutar para ver tus cambios.
```

Conjuntos (Sets)

Estudia el siguiente código: Conjuntos (Sets)

Código de Ejemplo

```
# Un Conjunto es una colección de datos que no está ordenada, ni
indexada y es única. Es uno de los 4 tipos de datos nativos en Python
# Esto se basa en un concepto de estructura de datos tabla hash.
# No podemos acceder a sus elementos por índice como en una lista
# Los conjuntos no pueden tener elementos mutables, de lo contrario
pueden contener datos mixtos

# Inicialización de conjunto vacío
# usa el método set(). Usar {} creará un diccionario vacío
test = {}
# Salida <class 'dict'>
print(type(test))
sets = set()
# Salida <class 'set'>
print(type(sets))

# inicialización de conjuntos
my_set = {1,2,3}
# Salida {1, 2, 3}
print(my_set)

# u otra forma es usando el método set()
my_another_set = set([4,5,6])
# Salida {4, 5, 6}
print(my_another_set)

# Añadir elementos
# usa el método set.add() para añadir elementos
for num in range(6):
    my_set.add(num)
# Salida {0, 1, 2, 3, 4, 5}
print(my_set)
```

```
# Eliminar elementos
# eliminar elementos del conjunto usando el método remove() o
discard()
# si el elemento no existe remove() lanzará un error, pero discard()
no lo hará
my_set.remove(4)
# Salida {0, 1, 2, 3, 5}
print(my_set)
my_set.discard(7)
# Salida {0, 1, 2, 3, 5}
print(my_set)


# Unión
# fusionando 2 conjuntos usando el método union() o el operador '|'
# devolverá un nuevo conjunto
final_set = my_set.union(my_another_set)
# Salida {0, 1, 2, 3, 4, 5, 6}
print(final_set)
same_set = my_set | my_another_set
# Salida {0, 1, 2, 3, 4, 5, 6}
print(same_set)

# Actualizar
# uniendo 2 conjuntos usando el método update()
# actualizará el conjunto con los datos de otro conjunto
my_set.update(my_another_set)
# Salida {0, 1, 2, 3, 5, 4, 6}
print(my_set)


# Intersección
# Intersección de 2 conjuntos usando el método intersection() o el
operador '&'
intersect = my_set.intersection(my_another_set)
# Salida {4, 5, 6}
print(intersect)
intersect2 = my_set & my_another_set
# Salida {4, 5, 6}
print(intersect2)
```

```
# Diferencia
# Diferencia de 2 conjuntos usando el método difference() o el
operador '-'
diff = my_set.difference(my_another_set)
# salida {0, 1, 2, 3}
print(diff)
diff2 = my_set - my_another_set
# Salida {0, 1, 2, 3}
print(diff2)

# Diferencia Simétrica
# usando el método symmetric_difference() o el operador '^'
# Salida {0, 1, 2, 3}
print(my_set ^ my_another_set)
# Salida {0, 1, 2, 3}
print(my_set.symmetric_difference(my_another_set))
# Salida {0, 1, 2, 3}
print(my_another_set.symmetric_difference(my_set))

# Limpiar el conjunto
intersect2.clear()
# Salida set()
print(intersect2)

# Iterando a través de conjuntos
for num in my_set:
    print(num)

# la palabra clave del eliminará el conjunto completamente
# del my_set

# UNA COSA MÁS
# Al igual que las tuplas son una lista inmutable, frozenset es un
conjunto inmutable
# imm_set = frozenset([1, 2, 3, 4])

# --- EJERCICIO INTERACTIVO ---
# 1. Ejecuta el código tal como está para ver el resultado.
# 2. Modifica el código para cambiar el comportamiento.
#    (Pista: Intenta cambiar los valores de las variables o la lógica)
# 3. Vuelve a ejecutar para ver tus cambios.
```

Funciones Avanzadas

Estudia el siguiente código: Funciones Avanzadas

Código de Ejemplo

```
#!/usr/bin/python

# Ejemplos de función en Python 3.x

# ¿Cuándo necesitas una función?
#   Cuando quieres realizar un conjunto de tareas específicas y
quieres reutilizar ese código siempre que sea necesario
#   También, para una mejor modularidad, legibilidad y solución de
problemas

# ¿Cómo escribir una función (Sintaxis)?
'''def nombre_funcion():
    {
        # algo de código aquí
    }
'''

# Diferentes formas de pasar parámetros
# Qué devolver a través de la función

# Una función básica de sumar dos números
def sumar_numeros(num1, num2):
    return num1+num2

# ¿Cómo llamar a una función?
# la forma más básica de llamar a una función es `nombre_funcion()`
# Llamando a la función de arriba
print(sumar_numeros(5,4))

# Función para encontrar si un número es par
def es_par(num):
    if num%2 == 0:
```

```
        return True
    return False

num = 12
resultado = es_par(num)
if resultado:
    print(f'{num} es par')
else:
    print(f'{num} no es par')



# --- EJERCICIO INTERACTIVO ---
# 1. Ejecuta el código tal como está para ver el resultado.
# 2. Modifica el código para cambiar el comportamiento.
#    (Pista: Intenta cambiar los valores de las variables o la lógica)
# 3. Vuelve a ejecutar para ver tus cambios.
```

Funciones de Colección

Estudia el siguiente código: Funciones de Colección

Código de Ejemplo

```
#!/usr/bin/python
# Ejemplo de lambda, map, filter

# 1) Ordenar la lista de diccionarios `people` alfabéticamente
basándose en la
# clave 'name' de cada diccionario usando la función `sorted` y
almacenar
# la nueva lista como `sorted_by_name`

people = [
    {"name": "Kevin Bacon", "age": 61},
    {"name": "Fred Ward", "age": 77},
    {"name": "finn Carter", "age": 59},
```

```
    {"name": "Ariana Richards", "age": 40},
    {"name": "Victor Wong", "age": 74},
]

# sorted_by_name = None
sorted_by_name = sorted(people, key=lambda d: d['name'].lower())

assert sorted_by_name == [
    {"name": "Ariana Richards", "age": 40},
    {"name": "finn Carter", "age": 59},
    {"name": "Fred Ward", "age": 77},
    {"name": "Kevin Bacon", "age": 61},
    {"name": "Victor Wong", "age": 74},
], f"Se esperaba que sorted_by_name fuera igual a '{sorted_by_name}'
igual a '{[{'name': 'Ariana Richards', 'age': 40}, {'name': 'finn
Carter', 'age': 59}, {'name': 'Fred Ward', 'age': 77}, {'name': 'Kevin
Bacon', 'age': 61}, {'name': 'Victor Wong', 'age': 74}]}'"

# 2) Usa la función `map` para iterar sobre `sorted_by_name` para
generar una
# nueva lista llamada `name_declarations` donde cada valor es una
cadena con
# `<NOMBRE> tiene <EDAD> años.` donde los valores `<NOMBRE>` y
`<EDAD>` son de
# los diccionarios.

# name_declarations = None
name_declarations = list(
    map(lambda d: f"{d['name']} tiene {d['age']} años",
sorted_by_name)
)

assert name_declarations == [
    "Ariana Richards tiene 40 años",
    "finn Carter tiene 59 años",
    "Fred Ward tiene 77 años",
    "Kevin Bacon tiene 61 años",
    "Victor Wong tiene 74 años",
], f"Se esperaba que name_declarations fuera igual a
'{name_declarations}' igual a '{['Ariana Richards tiene 40 años',
'finn Carter tiene 59 años', 'Fred Ward tiene 77 años', 'Kevin Bacon
tiene 61 años', 'Victor Wong tiene 74 años']}'"
```

```
# 3) Combina las funciones `filter` y `sorted` para iterar sobre
`sorted_by_name` para generar una
# nueva lista llamada `under_seventy` que solo contenga los
diccionarios donde la
# clave 'age' sea menor de 70, ordenando la lista por edad.

# under_seventy = None
under_seventy = sorted(
    filter(lambda d: d['age'] < 70, sorted_by_name), key=lambda d:
d['age']
)

assert under_seventy == [
    {"name": "Ariana Richards", "age": 40},
    {"name": "finn Carter", "age": 59},
    {"name": "Kevin Bacon", "age": 61},
], f"Se esperaba que under_seventy fuera igual a '{under_seventy}'
igual a '{[{'name': 'Ariana Richards', 'age': 40}, {'name': 'finn
Carter', 'age': 59}, {'name': 'Kevin Bacon', 'age': 61}]}'"



# --- EJERCICIO INTERACTIVO ---
# 1. Ejecuta el código tal como está para ver el resultado.
# 2. Modifica el código para cambiar el comportamiento.
#    (Pista: Intenta cambiar los valores de las variables o la lógica)
# 3. Vuelve a ejecutar para ver tus cambios.
```

Funciones Integradas

Estudia el siguiente código: Funciones Integradas

Código de Ejemplo

```
#Entrada en CLI
# input_text = input("Introduce algo aquí y luego presiona enter....
```

```
")
input_text = "Hola Python" # Valor hardcodeado
print("Ingresaste: ",input_text)

#Absoluto o Mod
n = abs(-12)
print("El valor absoluto de -12 es: ",n)

#Expresión Booleana
x=12>19
print("El valor booleano de la expresión (12<19) es ",bool(x))

data = {
    "id": 1,
    "name": "Ramesh",
    "designation":"SDE 1",
    "Hobbies": "Loves playing football"
    }

print(data,locals())

# --- EJERCICIO INTERACTIVO ---
# 1. Ejecuta el código tal como está para ver el resultado.
# 2. Modifica el código para cambiar el comportamiento.
#    (Pista: Intenta cambiar los valores de las variables o la lógica)
# 3. Vuelve a ejecutar para ver tus cambios.
```

Expresiones Regulares (Regex)

Estudia el siguiente código: Expresiones Regulares (Regex)

Código de Ejemplo

```
import re

urls = ["https://www.facebook.com","https://www.google.com","https://
www.amazon.in"]
```

```
def checkValidURL(url):
    url_reg_ex = r"^(([^:/?#]+):)?(//([^/?#]*))?([^?#]*)(\?([^#]*))?
(#(.*))?"
    data = re.search(url_reg_ex,url)
    if data is not None:
        return True
    return False

def parseDomain(url):
    domain = url.split("//")[1].split("www")[1].split(".")[1]
    print(domain)


if __name__ == "__main__":
    for url in urls:
        url_status = checkValidURL(url)
        if url_status:
            parseDomain(url)




# --- EJERCICIO INTERACTIVO ---
# 1. Ejecuta el código tal como está para ver el resultado.
# 2. Modifica el código para cambiar el comportamiento.
#    (Pista: Intenta cambiar los valores de las variables o la lógica)
# 3. Vuelve a ejecutar para ver tus cambios.
```

Introducción a Clases y Objetos

Estudia el siguiente código: Introducción a Clases y Objetos

Código de Ejemplo

```
"""
Las clases son "plantillas" para crear objetos.
```

```
Los objetos agrupan datos (atributos) y comportamientos (métodos).
"""

class Laptop:
    # El método __init__ es el constructor. Se ejecuta al crear un
objeto.
    def __init__(self, marca, modelo, ram):
        self.marca = marca
        self.modelo = modelo
        self.ram = ram
        self.encendida = False

    def encender(self):
        self.encendida = True
        print(f"{self.marca} {self.modelo} se está encendiendo...")

    def apagar(self):
        self.encendida = False
        print(f"{self.marca} {self.modelo} se está apagando...")

    def info(self):
        estado = "Encendida" if self.encendida else "Apagada"
        print(f"Laptop: {self.marca} {self.modelo} | RAM: {self.ram} |
Estado: {estado}")

# Crear objetos (instancias) de la clase Laptop
mi_laptop = Laptop("Dell", "XPS 13", "16GB")
tu_laptop = Laptop("Apple", "MacBook Pro", "32GB")

# Usar los objetos
mi_laptop.info()
tu_laptop.info()

mi_laptop.encender()
mi_laptop.info()

# --- EJERCICIO INTERACTIVO ---
# 1. Ejecuta el código tal como está para ver el resultado.
# 2. Crea una nueva instancia de Laptop con tus datos.
# 3. Llama a los métodos encender() y apagar() de tu nueva laptop.
```

Módulo 03: Práctica de Algoritmos

Clima de la Ciudad

Estudia el siguiente código: Clima de la Ciudad

Código de Ejemplo

```
#!/usr/bin/python3
# Script para obtener la temperatura y otra información de una ciudad
desde una app del clima
import json
from datetime import datetime

# Respuesta de API simulada para evitar bloqueos por clave API
inválida
city_name = "Madrid"

# Obtener la hora desde valores UTC y zona horaria proporcionados
# pasar el valor como utc + zona horaria (ambos son timestamp UTC)
def time_from_utc_with_timezone(utc_with_tz):
    local_time = datetime.utcfromtimestamp(utc_with_tz)
    return local_time.time()

# Datos del clima simulados (normalmente vendrían de una API)
weather_data = {
    "coord": {"lon": -3.7038, "lat": 40.4168},
    "weather": [{"id": 800, "main": "Clear", "description": "clear
sky", "icon": "01d"}],
    "base": "stations",
    "main": {"temp": 288.15, "feels_like": 287.5, "temp_min": 286.15,
"temp_max": 290.15, "pressure": 1013, "humidity": 65},
    "visibility": 10000,
    "wind": {"speed": 3.5, "deg": 180},
    "clouds": {"all": 0},
    "dt": 1700740800,
    "sys": {"type": 1, "id": 6443, "country": "ES", "sunrise":
```

```
1700724000, "sunset": 1700759400},
    "timezone": 3600,
    "id": 3117735,
    "name": "Madrid",
    "cod": 200
}

print(f"Simulando llamada a API del clima para {city_name}...")

# Asegurarse de obtener 200 como respuesta para proceder
if weather_data['cod'] == 200:
    kelvin = 273.15 # La temperatura aquí está en Kelvin y la mostraré
en Celsius
    temp = int(weather_data['main']['temp'] - kelvin)
    feels_like_temp = int(weather_data['main']['feels_like'] - kelvin)
    pressure = weather_data['main']['pressure']
    humidity = weather_data['main']['humidity']
    wind_speed = weather_data['wind']['speed'] * 3.6
    sunrise = weather_data['sys']['sunrise']
    sunset = weather_data['sys']['sunset']
    timezone = weather_data['timezone']
    cloudy = weather_data['clouds']['all']
    description = weather_data['weather'][0]['description']

    sunrise_time = time_from_utc_with_timezone(sunrise + timezone)
    sunset_time = time_from_utc_with_timezone(sunset + timezone)

    print(f"Información del Clima para la Ciudad: {city_name}")
    print(f"Temperatura (Celsius): {temp}")
    print(f"Sensación térmica (Celsius): {feels_like_temp}")
    print(f"Presión: {pressure} hPa")
    print(f"Humedad: {humidity}%")
    print("Velocidad del viento: {0:.2f} km/hr".format(wind_speed))
    print(f"Amanecer a las {sunrise_time} y Atardecer a las
{sunset_time}")
    print(f"Nubosidad: {cloudy}%")
    print(f"Info: {description}")
else:
    print(f"Nombre de Ciudad: {city_name} no encontrado!")

print("\nNota: Estos son datos simulados. Para usar una API real:")
print("1. Obtén una API key de openweathermap.org")
```

```
print("2. Reemplaza los datos simulados con: requests.get(weather_url,
timeout=5)")


# --- EJERCICIO INTERACTIVO ---
# 1. Ejecuta el código tal como está para ver el resultado.
# 2. Modifica el código para cambiar el comportamiento.
#    (Pista: Intenta cambiar los valores de las variables o la lógica)
# 3. Vuelve a ejecutar para ver tus cambios.
```

Conversor Fahrenheit a Celsius

Estudia el siguiente código: Conversor Fahrenheit a Celsius

Código de Ejemplo

```
#!/usr/bin/python3

def fahr_to_cel(fahrenheit):
    celsius = (fahrenheit - 32) / 1.8
    return celsius

try:
    # fahr = int(input('Introduce la temperatura en Fahrenheit por
favor: '))
    fahr = 100
    print(f"Calculando para {fahr} Fahrenheit...")
except:
    exit("Lo siento. Solo se permiten números reales")

print(fahr_to_cel(fahr))



# --- EJERCICIO INTERACTIVO ---
# 1. Ejecuta el código tal como está para ver el resultado.
# 2. Modifica el código para cambiar el comportamiento.
```

```
#    (Pista: Intenta cambiar los valores de las variables o la lógica)
# 3. Vuelve a ejecutar para ver tus cambios.
```

Base de Datos en Memoria

Estudia el siguiente código: Base de Datos en Memoria

Código de Ejemplo

```
# El problema requiere implementar un almacén clave-valor en memoria
# donde puedas establecer un par clave-valor y
# recuperar el valor de una clave.

# Ejemplo:
# db.set(101, ["Sanjeev", "ProdSec"])
# db.set(102, ["Deep", "DevSecOps"])
# db.get(102)
# Piensa en una línea de comandos para n entradas, establecer n
entradas, luego basado en el comando implementar get, delete
# Ejemplo:
# Elige opciones para operaciones:
#     1. establecer/crear entradas
#     Cuántas entradas: 2
#     id_emp: entrada
#     detalles_emp: entrada([])
#     2. obtener detalles id_emp
#     3. eliminar detalles id_emp
#     4. actualizar detalles id_emp

class DB:

    def __init__(self):
        self.dic = {}

    def set(self, key: int, value: list) -> None:
        if not isinstance(key, int):
            raise TypeError("La clave debe ser un entero")
```

```
        if not isinstance(value, list):
            raise TypeError("El valor debe ser una lista")
        if key in self.dic:
            raise ValueError(f"La clave {key} ya existe")
        self.dic[key] = value

    def get(self, key: int) -> list:
        if not isinstance(key, int):
            raise TypeError("La clave debe ser un entero")
        if key in self.dic:
            return self.dic[key]
        else:
            print(f"No se encontró registro para id emp: {key}")
            return None

    def delete(self, key: int) -> None:
        if not isinstance(key, int):
            raise TypeError("La clave debe ser un entero")
        if key in self.dic:
            print(f"Eliminando detalles id emp {key}:
{self.dic[key]}")
            del self.dic[key]
            print(f"Id emp {key} eliminado exitosamente.")
        else:
            print(f"No se encontró registro para id emp: {key}")

        print(f"Lista actual de empleados: {self.dic}")

# Ejemplo de uso
db = DB()

# Añadir empleados
try:
    db.set(101, ["Jassi", "ProdSec"])
    db.set(102, ["Deep", "DevSecOps"])
    db.set(103, ["Deepraj Barman", "Developer"])
    db.set(104, ["Himanshu", "AppSec"])
    # Intentando añadir un empleado con una clave existente
    db.set(101, ["Alex", "Security"])
except (TypeError, ValueError) as e:
    print(e)
```

```
# Obtener detalles emp
try:
    details = db.get(102)
    if details:
        print(f"Detalles id emp 102: {details}")
except TypeError as e:
    print(e)

# Eliminar detalles emp
try:
    db.delete(102)
except TypeError as e:
    print(e)




# --- EJERCICIO INTERACTIVO ---
# 1. Ejecuta el código tal como está para ver el resultado.
# 2. Modifica el código para cambiar el comportamiento.
#    (Pista: Intenta cambiar los valores de las variables o la lógica)
# 3. Vuelve a ejecutar para ver tus cambios.
```

DB en Memoria (CMD)

Estudia el siguiente código: DB en Memoria (CMD)

Código de Ejemplo

```
# El problema requiere implementar un almacén clave-valor en memoria
# donde puedas establecer un par clave-valor y
# recuperar el valor de una clave.

# Ejemplo:
# db.set(101, ["Sanjeev", "ProdSec"])
# db.set(102, ["Deep", "DevSecOps"])
# db.get(102)
```

```
class DB:

    def __init__(self):
        self.dic = {}

    def set(self, key: int, value: list) -> None:
        if not isinstance(key, int):
            raise TypeError("La clave debe ser un entero")
        if not isinstance(value, list):
            raise TypeError("El valor debe ser una lista")
        if key in self.dic:
            raise ValueError(f"La clave {key} ya existe")
        self.dic[key] = value

    def get(self, key: int) -> list:
        if not isinstance(key, int):
            raise TypeError("La clave debe ser un entero")
        if key in self.dic:
            return self.dic[key]
        else:
            print(f"No se encontró registro para id emp: {key}")
            return None

    def delete(self, key: int) -> None:
        if not isinstance(key, int):
            raise TypeError("La clave debe ser un entero")
        if key in self.dic:
            print(f"Eliminando detalles id emp {key}:
{self.dic[key]}")
            del self.dic[key]
            print(f"Id emp {key} eliminado exitosamente.")
        else:
            print(f"No se encontró registro para id emp: {key}")

        print(f"Lista actual de empleados: {self.dic}")

def main():
    db = DB()

    # Entradas precargadas de ejemplo
    print("--- Preloading Data ---")
```

```
    try:
        db.set(101, ["Jassi", "ProdSec"])
        db.set(102, ["Deep", "DevSecOps"])
        db.set(103, ["Deepraj Barman", "Developer"])
        db.set(104, ["Himanshu", "AppSec"])
    except (TypeError, ValueError) as e:
        print(e)

    # Simulando Operaciones de Línea de Comandos
    print("\n--- Simulando Operaciones ---")

    # Operación SET
    print("Ejecutando: SET 105 ['Alice', 'Manager']")
    try:
        db.set(105, ['Alice', 'Manager'])
        print(f"Clave 105 establecida exitosamente")
    except (TypeError, ValueError) as e:
        print(e)

    # Operación GET
    print("\nEjecutando: GET 102")
    try:
        details = db.get(102)
        if details:
            print(f"Detalles id emp 102: {details}")
    except TypeError as e:
        print(e)

    # Operación DELETE
    print("\nEjecutando: DELETE 101")
    try:
        db.delete(101)
    except TypeError as e:
        print(e)

if __name__ == "__main__":
    main()



# --- EJERCICIO INTERACTIVO ---
# 1. Ejecuta el código tal como está para ver el resultado.
```

```
# 2. Modifica el código para cambiar el comportamiento.
#    (Pista: Intenta cambiar los valores de las variables o la lógica)
# 3. Vuelve a ejecutar para ver tus cambios.
```

DB en Memoria (JSON)

Estudia el siguiente código: DB en Memoria (JSON)

Código de Ejemplo

```
# El problema requiere implementar un almacén clave-valor en memoria
# donde puedas establecer un par clave-valor y
# recuperar el valor de una clave.

import json
import os

class DB:
    def __init__(self, filename='db.json'):
        self.filename = filename
        self.dic = self.load()

    def load(self):
        if os.path.exists(self.filename):
            with open(self.filename, 'r') as f:
                return json.load(f)
        return {}

    def save(self):
        with open(self.filename, 'w') as f:
            json.dump(self.dic, f, indent=4)

    def set(self, key: int, value: list) -> None:
        if not isinstance(key, int):
            raise TypeError("La clave debe ser un entero")
        if not isinstance(value, list):
            raise TypeError("El valor debe ser una lista")
```

```
        if str(key) in self.dic: # Las claves JSON son siempre strings
             print(f"Advertencia: Sobrescribiendo clave existente
{key}")
        self.dic[str(key)] = value
        self.save()

    def get(self, key: int) -> list:
        if not isinstance(key, int):
            raise TypeError("La clave debe ser un entero")
        key_str = str(key)
        if key_str in self.dic:
            return self.dic[key_str]
        else:
            print(f"No se encontró registro para id emp: {key}")
            return None

    def delete(self, key: int) -> None:
        if not isinstance(key, int):
            raise TypeError("La clave debe ser un entero")
        key_str = str(key)
        if key_str in self.dic:
            print(f"Eliminando detalles id emp {key}:
{self.dic[key_str]}")
            del self.dic[key_str]
            print(f"Id emp {key} eliminado exitosamente.")
            self.save()
        else:
            print(f"No se encontró registro para id emp: {key}")
        print(f"Lista actual de empleados: {self.dic}")

def main():
    db = DB()

    print("--- Operaciones de Base de Datos con Persistencia JSON
---")

    # Demo set
    try:
        key = 999
        value = ["DemoUser", "DemoRole"]
        print(f"Estableciendo clave {key} con valor {value}")
        db.set(key, value)
```

```
    except (TypeError, ValueError) as e:
        print(e)

    # Demo get
    try:
        print(f"Obteniendo clave {key}")
        details = db.get(999)
        if details:
            print(f"Detalles id emp 999: {details}")
    except TypeError as e:
        print(e)

    # Demo verificación persistencia
    print("\n--- Verificando contenido del archivo ---")
    if os.path.exists('db.json'):
        with open('db.json', 'r') as f:
            print(f.read())

if __name__ == "__main__":
    main()


# --- EJERCICIO INTERACTIVO ---
# 1. Ejecuta el código tal como está para ver el resultado.
# 2. Modifica el código para cambiar el comportamiento.
#    (Pista: Intenta cambiar los valores de las variables o la lógica)
# 3. Vuelve a ejecutar para ver tus cambios.
```

DB Persistente

Estudia el siguiente código: DB Persistente

Código de Ejemplo

```
# Para hacer el programa interactivo y persistente,
# puedes usar un bucle para pedir operaciones al usuario continuamente
hasta que decida salir.
```

```
# El problema requiere implementar un almacén clave-valor en memoria
# donde puedas establecer un par clave-valor y
# recuperar el valor de una clave.

# Ejemplo:
# db.set(101, ["Sanjeev", "ProdSec"])
# db.set(102, ["Deep", "DevSecOps"])
# db.get(102)
# Piensa en una línea de comandos para n entradas, establecer n
entradas, luego basado en el comando implementar get, delete
# Ejemplo:
# Elige opciones para operaciones:
#     1. establecer/crear entradas
#     Cuántas entradas: 2
#     id_emp: entrada
#     detalles_emp: entrada([])
#     2. obtener detalles id_emp
#     3. eliminar detalles id_emp
#     4. actualizar detalles id_emp

class DB:
    def __init__(self):
        self.dic = {}

    def set(self, key: int, value: list) -> None:
        if not isinstance(key, int):
            raise TypeError("La clave debe ser un entero")
        if not isinstance(value, list):
            raise TypeError("El valor debe ser una lista")
        if key in self.dic:
            raise ValueError(f"La clave {key} ya existe")
        self.dic[key] = value

    def get(self, key: int) -> list:
        if not isinstance(key, int):
            raise TypeError("La clave debe ser un entero")
        if key in self.dic:
            return self.dic[key]
        else:
            print(f"No se encontró registro para id emp: {key}")
            return None
```

```
    def delete(self, key: int) -> None:
        if not isinstance(key, int):
            raise TypeError("La clave debe ser un entero")
        if key in self.dic:
            print(f"Eliminando detalles id emp {key}:
{self.dic[key]}")
            del self.dic[key]
            print(f"Id emp {key} eliminado exitosamente.")
        else:
            print(f"No se encontró registro para id emp: {key}")
        print(f"Lista actual de empleados: {self.dic}")

def main():
    db = DB()
    db.set(101, ["Jassi", "ProdSec"])
    db.set(102, ["Deep", "DevSecOps"])
    db.set(103, ["Deepraj Barman", "Developer"])
    db.set(104, ["Himanshu", "AppSec"])

    # while True:
    #    print("\nElige una operación:")
    #    ...

    print("Modo interactivo desactivado. Ejecutando demo...")
    # Demo set
    try:
        key = 999
        value = ["DemoUser", "DemoRole"]
        db.set(key, value)
        print(f"Establecida clave {key} con valor {value}")
    except (TypeError, ValueError) as e:
        print(e)

if __name__ == "__main__":
    main()


# --- EJERCICIO INTERACTIVO ---
# 1. Ejecuta el código tal como está para ver el resultado.
# 2. Modifica el código para cambiar el comportamiento.
#    (Pista: Intenta cambiar los valores de las variables o la lógica)
# 3. Vuelve a ejecutar para ver tus cambios.
```

Es Primo

Estudia el siguiente código: Es Primo

Código de Ejemplo

```
#!/usr/bin/python3

# Ejercicio para if else y bucle for
# Script para decir si un número es primo o no
# Si un número es divisible solo por 1 y por sí mismo, entonces es un
número primo

# Obtener entrada del usuario, como es string, necesitas convertirla a
int
try:
    # num = int(input('Introduce un número: '))
    num = 29
    print(f"Comprobando si {num} es primo...")
except:
    exit("¡Solo enteros por favor!")


# Verificar si el número es negativo
if num < 0:
    exit('El número debe ser positivo')
# Si el número es positivo, escribir la lógica de número primo
else:
    for prime in range(2, num):
        if (num % prime)== 0:
            print(num, 'no es un número primo')
            break
    else:
        print(num, 'es un número primo')



# --- EJERCICIO INTERACTIVO ---
```

```
# 1. Ejecuta el código tal como está para ver el resultado.
# 2. Modifica el código para cambiar el comportamiento.
#    (Pista: Intenta cambiar los valores de las variables o la lógica)
# 3. Vuelve a ejecutar para ver tus cambios.
```

N Números Primos

Estudia el siguiente código: N Números Primos

Código de Ejemplo

```
#!/usr/bin/python3

# Función para comprobar si un número es primo contra una lista de
primos dada
def prime(num, primes):
    # bucle para probar si num es primo contra la lista dada
    for prime in primes:
        if  (num % prime) == 0:
            return False
    # Tenemos un nuevo número primo, añadirlo a la lista primes[]
    primes.append(num)
    return True

def n_primes(n):
    primes = []
    start_num, prime_counter = 2, 0
    while True:
        if prime(start_num, primes):
            prime_counter += 1
            if prime_counter == n:
                return primes
        start_num += 1

try:
    # nprimes = int(input("Introduce cuántos números primos quieres
mostrar: "))
```

```
    nprimes = 10
    print(f"Generando los primeros {nprimes} números primos...")
except:
    exit("Debe ser solo un entero")


if nprimes < 0:
    exit("¡El número debe ser positivo!")
else:
    primes_list = n_primes(nprimes)
    print(primes_list)



# --- EJERCICIO INTERACTIVO ---
# 1. Ejecuta el código tal como está para ver el resultado.
# 2. Modifica el código para cambiar el comportamiento.
#    (Pista: Intenta cambiar los valores de las variables o la lógica)
# 3. Vuelve a ejecutar para ver tus cambios.
```

Primos en Rango

Estudia el siguiente código: Primos en Rango

Código de Ejemplo

```
#!/usr/bin/python

try:
    # lower = int(input('Introduce inicio del rango: '))
    # upper = int(input('Introduce fin del rango: '))
    lower = 10
    upper = 50
    print(f"Primos entre {lower} y {upper}...")
except:
    exit("Asegúrate de que los rangos sean solo enteros")
```

```
if( lower < 0 or upper < 0 ):
    exit("Los rangos deben ser números positivos")

print("Los números primos entre", lower, "y", upper, "son:")
for num in range(lower, upper+1):
    if(num > 1):
        for i in range(2, num):
            if (num % i) == 0:
                break
        else:
            print(num)



# --- EJERCICIO INTERACTIVO ---
# 1. Ejecuta el código tal como está para ver el resultado.
# 2. Modifica el código para cambiar el comportamiento.
#    (Pista: Intenta cambiar los valores de las variables o la lógica)
# 3. Vuelve a ejecutar para ver tus cambios.
```

Suma de Dos Índices

Estudia el siguiente código: Suma de Dos Índices

Código de Ejemplo

```
#!/usr/bin/python

# Hay un array lleno de enteros y un valor objetivo t, también entero
# Necesitas encontrar qué par de enteros suman el objetivo e imprimir
sus índices
# Puedes asumir que solo hay un par que resulta en la suma objetivo

num_list = [2, 1, 3, 5, 6, 11, 2, 13, 4, 15]
target = 12

def twoSum(arr, t):
```

```
    index_dict = {}
    length = len(arr)
    index = 0

    while index < length:
        if (t - arr[index]) in index_dict:
            return index_dict[t - arr[index]], index
        index_dict[arr[index]] = index
        index += 1

print(twoSum(num_list, target))



# --- EJERCICIO INTERACTIVO ---
# 1. Ejecuta el código tal como está para ver el resultado.
# 2. Modifica el código para cambiar el comportamiento.
#    (Pista: Intenta cambiar los valores de las variables o la lógica)
# 3. Vuelve a ejecutar para ver tus cambios.
```

Módulo 04: Automatización y Scripts

Avatar Básico

Estudia el siguiente código: Avatar Básico

Código de Ejemplo

```
# --- IMPORTANTE: EJECUCIÓN LOCAL REQUERIDA ---
# Este ejercicio requiere librerías externas o generar archivos.
# Por favor, ejecútalo en tu IDE local (VS Code, PyCharm, etc).
# ------------------------------------------------------
from py_avataaars import PyAvataaar
# Es posible que necesites instalar la librería cairo.
# Para mac: escribe `brew install cairo`
avatar = PyAvataaar()
avatar.render_png_file('basic_avatar.png')
avatar.render_svg_file('basic_avatar.svg')



# --- EJERCICIO INTERACTIVO ---
# 1. Ejecuta el código tal como está para ver el resultado.
# 2. Modifica el código para cambiar el comportamiento.
#    (Pista: Intenta cambiar los valores de las variables o la lógica)
# 3. Vuelve a ejecutar para ver tus cambios.
```

Avatar Personalizado

Estudia el siguiente código: Avatar Personalizado

Código de Ejemplo

```
# --- IMPORTANTE: EJECUCIÓN LOCAL REQUERIDA ---
# Este ejercicio requiere librerías externas o generar archivos.
# Por favor, ejecútalo en tu IDE local (VS Code, PyCharm, etc).
# ----------------------------------------------------
import py_avataaars as pa

avatar = pa.PyAvataaar(
    style=pa.AvatarStyle.TRANSPARENT,
    skin_color=pa.SkinColor.LIGHT,
    hair_color=pa.HairColor.BLACK,
    facial_hair_type=pa.FacialHairType.DEFAULT,
    facial_hair_color=pa.HairColor.BLACK,
    top_type=pa.TopType.SHORT_HAIR_SHORT_FLAT,
    hat_color=pa.Color.BLACK,
    mouth_type=pa.MouthType.SMILE,
    eye_type=pa.EyesType.DEFAULT,
    eyebrow_type=pa.EyebrowType.DEFAULT,
    nose_type=pa.NoseType.DEFAULT,
    accessories_type=pa.AccessoriesType.SUNGLASSES,
    clothe_type=pa.ClotheType.SHIRT_V_NECK,
    clothe_color=pa.Color.BLUE_03,
    clothe_graphic_type=pa.ClotheGraphicType.BAT,
)
avatar.render_png_file('avatar_custom.png')


# --- EJERCICIO INTERACTIVO ---
# 1. Ejecuta el código tal como está para ver el resultado.
# 2. Modifica el código para cambiar el comportamiento.
#    (Pista: Intenta cambiar los valores de las variables o la lógica)
# 3. Vuelve a ejecutar para ver tus cambios.
```

Avatar Personalizado II

Estudia el siguiente código: Avatar Personalizado II

Código de Ejemplo

```
# --- IMPORTANTE: EJECUCIÓN LOCAL REQUERIDA ---
# Este ejercicio requiere librerías externas o generar archivos.
# Por favor, ejecútalo en tu IDE local (VS Code, PyCharm, etc).
# ----------------------------------------------------
import python_avatars as pa

pa.Avatar(
    style=pa.AvatarStyle.TRANSPARENT,
    # background_color='#FF00FF',
    # Choose graphic shirt
    clothing=pa.ClothingType.GRAPHIC_SHIRT,
    clothing_color=pa.ClothingColor.BLUE_01,
    # Important to choose this as shirt_graphic, otherwise shirt_text
will be ignored
    shirt_graphic=pa.ClothingGraphic.CUSTOM_TEXT,
    shirt_text='Flexmind'
).render("avatar_text.svg")

# --- EJERCICIO INTERACTIVO ---
# 1. Ejecuta el código tal como está para ver el resultado.
# 2. Modifica el código para cambiar el comportamiento.
#    (Pista: Intenta cambiar los valores de las variables o la lógica)
# 3. Vuelve a ejecutar para ver tus cambios.
```

Avatar Aleatorio

Estudia el siguiente código: Avatar Aleatorio

Código de Ejemplo

```
# --- IMPORTANTE: EJECUCIÓN LOCAL REQUERIDA ---
# Este ejercicio requiere librerías externas o generar archivos.
# Por favor, ejecútalo en tu IDE local (VS Code, PyCharm, etc).
# ----------------------------------------------------
```

```
import python_avatars as pa

# Avatar completamente aleatorio
random_avatar_1 = pa.Avatar.random()

# Avatar aleatorio excepto el sombrero
random_avatar_2 = pa.Avatar.random(top=pa.HatType.HAT)  # Más
atributos pueden mantenerse fijos

# Avatar fijo pero ropa aleatoria
random_avatar_3 = pa.Avatar(
    style=pa.AvatarStyle.CIRCLE,
    hair_color=pa.HairColor.BLACK,
    accessory=pa.AccessoryType.NONE,
    clothing=pa.ClothingType.pick_random(), # La ropa se elige
aleatoriamente
)

# Renderizar salida
random_avatar_1.render("avatar_1.svg")
print("Guardado avatar_1.svg")
random_avatar_2.render("avatar_2.svg")
print("Guardado avatar_2.svg")
random_avatar_3.render("avatar_3.svg")
print("Guardado avatar_3.svg")

# --- EJERCICIO INTERACTIVO ---
# 1. Ejecuta el código tal como está para ver el resultado.
# 2. Modifica el código para cambiar el comportamiento.
#    (Pista: Intenta cambiar los valores de las variables o la lógica)
# 3. Vuelve a ejecutar para ver tus cambios.
```

Reloj Digital Básico

Estudia el siguiente código: Reloj Digital Básico

Código de Ejemplo

```
#!/usr/bin/python3
# --- IMPORTANTE: EJECUCIÓN LOCAL REQUERIDA ---
# Este ejercicio requiere librerías externas o generar archivos.
# Por favor, ejecútalo en tu IDE local (VS Code, PyCharm, etc).
# ----------------------------------------------------
# Reloj Digital Básico (Versión Web)
# Este script genera un archivo HTML con un reloj digital funcional.

html_content = """
<!DOCTYPE html>
<html>
<head>
<style>
body { display: flex; justify-content: center; align-items: center;
height: 100vh; background-color: #282c34; color: #61dafb; font-family:
monospace; }
.clock { font-size: 4rem; border: 4px solid #61dafb; padding: 20px;
border-radius: 10px; box-shadow: 0 0 20px rgba(97, 218, 251, 0.5); }
</style>
<script>
function updateClock() {
    const now = new Date();
    const timeString = now.toLocaleTimeString();
    document.getElementById("clock").innerText = timeString;
}
setInterval(updateClock, 1000);
window.onload = updateClock;
</script>
</head>
<body>
<div id="clock" class="clock">Loading...</div>
</body>
</html>
"""

filename = "reloj_digital.html"
with open(filename, "w") as f:
    f.write(html_content)
```

```
print(f"Generado archivo: {filename}")
print("El reloj debería aparecer en la sección de archivos
generados.")

# --- EJERCICIO INTERACTIVO ---
# 1. Ejecuta el código para generar el reloj.
# 2. Modifica el CSS en la variable html_content para cambiar el color
del reloj.
```

Reloj Digital Avanzado

Estudia el siguiente código: Reloj Digital Avanzado

Código de Ejemplo

```
#!/usr/bin/python3
# --- IMPORTANTE: EJECUCIÓN LOCAL REQUERIDA ---
# Este ejercicio requiere librerías externas o generar archivos.
# Por favor, ejecútalo en tu IDE local (VS Code, PyCharm, etc).
# ------------------------------------------------------
# Reloj Digital Avanzado (Temporizador Web)
# Genera un temporizador de cuenta regresiva en HTML.

seconds = 10 # Duración del temporizador

html_content = f"""
<!DOCTYPE html>
<html>
<head>
<style>
body {{ display: flex; justify-content: center; align-items: center;
height: 100vh; background-color: #222; color: #ff5555; font-family:
"Courier New", monospace; }}
.timer {{ font-size: 5rem; font-weight: bold; }}
.message {{ font-size: 2rem; color: #50fa7b; display: none; }}
</style>
<script>
```

```
let timeLeft = {seconds};
function updateTimer() {{
    if (timeLeft <= 0) {{
        document.getElementById("timer").style.display = "none";
        document.getElementById("message").style.display = "block";
        return;
    }}
    const mins = Math.floor(timeLeft / 60).toString().padStart(2,
"0");
    const secs = (timeLeft % 60).toString().padStart(2, "0");
    document.getElementById("timer").innerText = `${{mins}}:${{secs}}
`;
    timeLeft--;
}}
setInterval(updateTimer, 1000);
window.onload = updateTimer;
</script>
</head>
<body>
<div id="timer" class="timer">00:00</div>
<div id="message" class="message">¡TIEMPO TERMINADO!</div>
</body>
</html>
"""

filename = "temporizador.html"
with open(filename, "w") as f:
    f.write(html_content)

print(f"Generado archivo: {filename}")
print(f"Temporizador configurado para {seconds} segundos.")

# --- EJERCICIO INTERACTIVO ---
# 1. Ejecuta el código para generar el temporizador.
# 2. Cambia la variable 'seconds' para ajustar la duración.
```

Reloj Digital CLI

Estudia el siguiente código: Reloj Digital CLI

Código de Ejemplo

```
#!/usr/bin/python3
# --- IMPORTANTE: EJECUCIÓN LOCAL REQUERIDA ---
# Este ejercicio requiere librerías externas o generar archivos.
# Por favor, ejecútalo en tu IDE local (VS Code, PyCharm, etc).
# ----------------------------------------------------
# Reloj Digital con Fecha (Versión Web)
# Genera un reloj que muestra fecha y hora.

html_content = """
<!DOCTYPE html>
<html>
<head>
<style>
body { display: flex; flex-direction: column; justify-content: center;
align-items: center; height: 100vh; background-color: #000; color:
#0f0; font-family: "Consolas", monospace; }
.time { font-size: 6rem; text-shadow: 0 0 10px #0f0; }
.date { font-size: 2rem; color: #0a0; margin-top: 10px; }
</style>
<script>
function updateClock() {
    const now = new Date();
    document.getElementById("time").innerText =
now.toLocaleTimeString();
    document.getElementById("date").innerText =
now.toLocaleDateString(undefined, { weekday: "long", year: "numeric",
month: "long", day: "numeric" });
}
setInterval(updateClock, 1000);
window.onload = updateClock;
</script>
</head>
<body>
<div id="time" class="time">--:--:--</div>
```

```
<div id="date" class="date">Loading date...</div>
</body>
</html>
"""

filename = "reloj_fecha.html"
with open(filename, "w") as f:
    f.write(html_content)

print(f"Generado archivo: {filename}")

# --- EJERCICIO INTERACTIVO ---
# 1. Ejecuta el código.
# 2. Modifica el estilo CSS para cambiar el color del texto a rojo
(#f00).
```

Reloj Mundial

Estudia el siguiente código: Reloj Mundial

Código de Ejemplo

```
#!/usr/bin/python3
# --- IMPORTANTE: EJECUCIÓN LOCAL REQUERIDA ---
# Este ejercicio requiere librerías externas o generar archivos.
# Por favor, ejecútalo en tu IDE local (VS Code, PyCharm, etc).
# ------------------------------------------------------
# Reloj Mundial (Versión Web)
# Genera un tablero con múltiples zonas horarias.

html_content = """
<!DOCTYPE html>
<html>
<head>
<style>
body { font-family: sans-serif; background: #f0f2f5; display: flex;
justify-content: center; align-items: center; height: 100vh; margin:
```

```
0; }
.container { display: grid; grid-template-columns: repeat(auto-fit,
minmax(200px, 1fr)); gap: 20px; width: 80%; max-width: 1000px; }
.clock-card { background: white; padding: 20px; border-radius: 15px;
box-shadow: 0 4px 6px rgba(0,0,0,0.1); text-align: center; }
.city { font-size: 1.2rem; color: #666; margin-bottom: 10px; font-
weight: bold; }
.time { font-size: 2rem; color: #333; font-family: monospace; }
</style>
<script>
const zones = [
    { city: "New York", zone: "America/New_York" },
    { city: "London", zone: "Europe/London" },
    { city: "Tokyo", zone: "Asia/Tokyo" },
    { city: "Sydney", zone: "Australia/Sydney" },
    { city: "India", zone: "Asia/Kolkata" }
];

function updateClocks() {
    const now = new Date();
    zones.forEach(item => {
        const timeString = now.toLocaleTimeString("en-US", { timeZone:
item.zone });
        document.getElementById(item.city).innerText = timeString;
    });
}
setInterval(updateClocks, 1000);
window.onload = updateClocks;
</script>
</head>
<body>
<div class="container">
    <div class="clock-card"><div class="city">New York</div><div
id="New York" class="time">--:--:--</div></div>
    <div class="clock-card"><div class="city">London</div><div
id="London" class="time">--:--:--</div></div>
    <div class="clock-card"><div class="city">Tokyo</div><div
id="Tokyo" class="time">--:--:--</div></div>
    <div class="clock-card"><div class="city">Sydney</div><div
id="Sydney" class="time">--:--:--</div></div>
    <div class="clock-card"><div class="city">India</div><div
id="India" class="time">--:--:--</div></div>
```

```
</div>
</body>
</html>
"""

filename = "reloj_mundial.html"
with open(filename, "w") as f:
    f.write(html_content)

print(f"Generado archivo: {filename}")

# --- EJERCICIO INTERACTIVO ---
# 1. Ejecuta el código para ver el reloj mundial.
# 2. Intenta añadir una nueva tarjeta para otra ciudad en el HTML.
```

Captura de Cámara (OpenCV - Headless)

Estudia el siguiente código: Captura de Cámara (OpenCV - Headless)

Código de Ejemplo

```
# --- IMPORTANTE: EJECUCIÓN LOCAL REQUERIDA ---
# Este ejercicio requiere librerías externas o generar archivos.
# Por favor, ejecútalo en tu IDE local (VS Code, PyCharm, etc).
# ------------------------------------------------------
# capturar una sola imagen de la webcam usando python
# Nota: En un entorno servidor/web, no podemos abrir una ventana
(imshow).
# En su lugar, guardamos la imagen en un archivo.

import cv2 as cv
import time
import os

# inicializar la cámara
# Si no hay cámara física, esto podría fallar o devolver un frame
negro.
```

```
cam_port = 0

print("Inicializando cámara...")
try:
    cam = cv.VideoCapture(cam_port)

    # esperar a que la cámara se caliente
    time.sleep(1)

    # leyendo la entrada usando la cámara
    result, image = cam.read()

    # Si la imagen se detecta sin errores, mostrar resultado
    if result:
        # guardando imagen en almacenamiento local
        filename = "cam_capture.jpg"
        cv.imwrite(filename, image)
        print(f"¡Foto tomada! Guardada como {filename}")
    else:
        print("No se pudo acceder a la cámara. (Es normal en
servidores sin webcam)")

except Exception as e:
    print(f"Error de cámara: {e}")

# --- EJERCICIO INTERACTIVO ---
# 1. Ejecuta el código.
# 2. Si falla (sin webcam), intenta cargar una imagen existente con
cv.imread() y guardarla con otro nombre.
```

Imagen a Caricatura (Real)

Estudia el siguiente código: Imagen a Caricatura (Real)

Código de Ejemplo

```
#!/usr/bin/python
# --- IMPORTANTE: EJECUCIÓN LOCAL REQUERIDA ---
# Este ejercicio requiere librerías externas o generar archivos.
# Por favor, ejecútalo en tu IDE local (VS Code, PyCharm, etc).
# ---------------------------------------------------
# Procesamiento de Imagen Real con PIL
# Primero generamos una imagen base y luego aplicamos filtros.

import py_avataaars as pa
from PIL import Image, ImageFilter, ImageOps
import os

print("--- Generando imagen base ---")
# 1. Generar un avatar como imagen base
avatar = pa.PyAvataaar(
    style=pa.AvatarStyle.CIRCLE,
    skin_color=pa.SkinColor.LIGHT,
    hair_color=pa.HairColor.BROWN,
    facial_hair_type=pa.FacialHairType.DEFAULT,
    top_type=pa.TopType.SHORT_HAIR_SHORT_FLAT,
    mouth_type=pa.MouthType.SMILE,
    eye_type=pa.EyesType.DEFAULT
)
avatar.render_png_file('base_image.png')
print("Imagen base guardada: base_image.png")

# 2. Procesar con PIL (Pillow)
print("--- Aplicando efectos de caricatura ---")
try:
    img = Image.open('base_image.png').convert("RGB")

    # Aplicar filtro de bordes (Contour)
    edges = img.filter(ImageFilter.CONTOUR)

    # Aplicar suavizado
    smooth = img.filter(ImageFilter.SMOOTH_MORE)

    # Combinar para efecto "cartoon" simple
    cartoon = Image.blend(smooth, edges, 0.5)
```

```
    # Guardar resultado
    output_file = "cartoon_result.png"
    cartoon.save(output_file)
    print(f"¡Éxito! Caricatura guardada como: {output_file}")

except Exception as e:
    print(f"Error procesando imagen: {e}")

# --- EJERCICIO INTERACTIVO ---
# 1. Ejecuta el código para ver el resultado real.
# 2. Cambia ImageFilter.CONTOUR por ImageFilter.EMBOSS o
ImageFilter.FIND_EDGES.
```

Caricatura con GUI (Web)

Estudia el siguiente código: Caricatura con GUI (Web)

Código de Ejemplo

```
#!/usr/bin/python3
# --- IMPORTANTE: EJECUCIÓN LOCAL REQUERIDA ---
# Este ejercicio requiere librerías externas o generar archivos.
# Por favor, ejecútalo en tu IDE local (VS Code, PyCharm, etc).
# ------------------------------------------------------
# Editor de Caricaturas (Interfaz Web)
# Genera una interfaz HTML para aplicar filtros a una imagen.

html_content = """
<!DOCTYPE html>
<html>
<head>
<style>
body { font-family: sans-serif; text-align: center; background-color:
#222; color: white; padding: 20px; }
.container { background: #333; padding: 20px; border-radius: 10px;
display: inline-block; }
```

```
img { max-width: 100%; border-radius: 5px; transition: filter 0.3s;
margin-bottom: 20px; border: 2px solid #555; }
.controls { display: flex; gap: 10px; justify-content: center; flex-
wrap: wrap; }
button { padding: 10px 20px; border: none; border-radius: 5px; cursor:
pointer; background-color: #007bff; color: white; font-size: 16px; }
button:hover { background-color: #0056b3; }
</style>
<script>
function applyFilter(filter) {
    const img = document.getElementById('targetImage');
    img.style.filter = filter;
}
</script>
</head>
<body>
<div class="container">
    <h1>Editor de Caricaturas Web</h1>
    <!-- Usamos una imagen de ejemplo -->
    <img id="targetImage" src="https://avataaars.io/?
avatarStyle=Circle&topType=ShortHairShortFlat&accessoriesType=Blank&ha
irColor=Brown&facialHairType=Blank&clotheType=Hoodie&clotheColor=Blue0
3&eyeType=Happy&eyebrowType=Default&mouthType=Smile&skinColor=Light"
alt="Avatar">

    <div class="controls">
        <button onclick="applyFilter('none')">Normal</button>
        <button onclick="applyFilter('grayscale(100%)')">B/N</button>
        <button onclick="applyFilter('sepia(100%)')">Sepia</button>
        <button onclick="applyFilter('contrast(200%)')">Alto
Contraste</button>
        <button onclick="applyFilter('invert(100%)')">Invertir</
button>
        <button onclick="applyFilter('blur(5px)')">Desenfoque</button>
        <button onclick="applyFilter('saturate(300%)')">Saturado</
button>
        <button onclick="applyFilter('hue-rotate(90deg)')">Alien</
button>
    </div>
</div>
</body>
</html>
```

```
"""

filename = "editor_caricatura.html"
with open(filename, "w") as f:
    f.write(html_content)

print(f"Generado archivo: {filename}")
print("La interfaz gráfica aparecerá en la sección de archivos
generados.")

# --- EJERCICIO INTERACTIVO ---
# 1. Ejecuta el código para generar la interfaz.
# 2. Añade un nuevo botón en el HTML para un filtro diferente (ej.
brightness).
```

Generador QR (API Real)

Estudia el siguiente código: Generador QR (API Real)

Código de Ejemplo

```
#!/usr/bin/python
# --- IMPORTANTE: EJECUCIÓN LOCAL REQUERIDA ---
# Este ejercicio requiere librerías externas o generar archivos.
# Por favor, ejecútalo en tu IDE local (VS Code, PyCharm, etc).
# ------------------------------------------------------
# Generador de QR usando una API pública
# No requiere instalar librerías complejas.

import requests

data = "https://www.python.org"
size = "300x300"
api_url = f"https://api.qrserver.com/v1/create-qr-code/?size={size}
&data={data}"

print(f"Generando QR para: {data}")
```

```
print("Contactando API...")

try:
    response = requests.get(api_url)
    if response.status_code == 200:
        filename = "qr_code.png"
        with open(filename, "wb") as f:
            f.write(response.content)
        print(f"¡Éxito! QR guardado como: {filename}")
    else:
        print("Error en la API.")
except Exception as e:
    print(f"Error de conexión: {e}")

# --- EJERCICIO INTERACTIVO ---
# 1. Ejecuta el código.
# 2. Cambia la variable 'data' por tu propio texto o URL.
```

Info de Imagen (PIL)

Estudia el siguiente código: Info de Imagen (PIL)

Código de Ejemplo

```
#!/usr/bin/python
# --- IMPORTANTE: EJECUCIÓN LOCAL REQUERIDA ---
# Este ejercicio requiere librerías externas o generar archivos.
# Por favor, ejecútalo en tu IDE local (VS Code, PyCharm, etc).
# -------------------------------------------------
# Análisis de Imagen con Pillow (PIL)

from PIL import Image
import os

# Usamos una imagen generada previamente o descargada
filename = "qr_code.png"
```

```
if not os.path.exists(filename):
    print(f"El archivo {filename} no existe. Ejecuta la lección
anterior primero.")
else:
    try:
        with Image.open(filename) as img:
            print("--- Información de la Imagen ---")
            print(f"Archivo: {filename}")
            print(f"Formato: {img.format}")
            print(f"Modo: {img.mode}")
            print(f"Tamaño: {img.size} (Ancho: {img.width}, Alto:
{img.height})")

            # Operación simple: Convertir a escala de grises
            grayscale = img.convert("L")
            grayscale.save("qr_grayscale.png")
            print("Imagen convertida a escala de grises guardada como
'qr_grayscale.png'")

    except Exception as e:
        print(f"Error procesando imagen: {e}")

# --- EJERCICIO INTERACTIVO ---
# 1. Ejecuta el código.
# 2. Intenta rotar la imagen usando img.rotate(90).
```

Análisis de Texto (Frecuencia)

Estudia el siguiente código: Análisis de Texto (Frecuencia)

Código de Ejemplo

```
#!/usr/bin/python
# Análisis de Frecuencia de Palabras

from collections import Counter
import re
```

```
texto = """
Python es un lenguaje de programación interpretado cuya filosofía hace hincapié
en la legibilidad de su código. Se trata de un lenguaje de programación multiparadigma,
ya que soporta orientación a objetos, programación imperativa y, en menor medida,
programación funcional. Es un lenguaje interpretado, dinámico y multiplataforma.
"""

print("--- Texto Original ---")
print(texto.strip())
print("-" * 20)

# Limpieza básica: minúsculas y eliminar puntuación
limpio = re.sub(r"[^\w\s]", "", texto.lower())
palabras = limpio.split()

print(f"Total de palabras: {len(palabras)}")

# Contar frecuencia
conteo = Counter(palabras)

print("\n--- Palabras más comunes ---")
for palabra, cantidad in conteo.most_common(5):
    print(f"{palabra}: {cantidad}")

# --- EJERCICIO INTERACTIVO ---
# 1. Ejecuta el código.
# 2. Añade más texto a la variable 'texto' y observa cómo cambia el conteo.
```

Análisis de Sentimiento (Simple)

Estudia el siguiente código: Análisis de Sentimiento (Simple)

Código de Ejemplo

```
#!/usr/bin/python
# Analizador de Sentimiento Básico (Basado en palabras clave)

positivas = ["bueno", "excelente", "increíble", "feliz", "amor",
"gusta", "genial"]
negativas = ["malo", "terrible", "triste", "odio", "feo", "error",
"fallo"]

def analizar_sentimiento(texto):
    texto = texto.lower()
    score = 0

    for p in positivas:
        if p in texto:
            score += 1

    for n in negativas:
        if n in texto:
            score -= 1

    return score

frases = [
    "Python es increíble y genial",
    "Este error es terrible y feo",
    "Hoy es un día normal"
]

print("--- Análisis de Sentimiento ---")
for frase in frases:
    score = analizar_sentimiento(frase)
    veredicto = "Neutro"
    if score > 0: veredicto = "Positivo"
    if score < 0: veredicto = "Negativo"

    print(f"Frase: '{frase}' | Score: {score} ({veredicto})")

# --- EJERCICIO INTERACTIVO ---
```

```
# 1. Ejecuta el código.
# 2. Añade tus propias frases a la lista y ve cómo se clasifican.
```

Validador de Email (Regex)

Estudia el siguiente código: Validador de Email (Regex)

Código de Ejemplo

```
#!/usr/bin/python
# Validación de Email con Expresiones Regulares

import re

# Regex estándar para email
email_regex = r"^[a-zA-Z0-9._%+-]+@[a-zA-Z0-9.-]+\.[a-zA-Z]{2,}$"

emails = [
    "usuario@example.com",
    "nombre.apellido@empresa.co.uk",
    "invalido@com",
    "sin_arroba.com",
    "user@domain"
]

print("--- Validador de Emails ---")
for email in emails:
    if re.match(email_regex, email):
        print(f"[VALIDO]   {email}")
    else:
        print(f"[INVALIDO] {email}")

# --- EJERCICIO INTERACTIVO ---
# 1. Ejecuta el código.
# 2. Intenta mejorar la regex o probar con otros emails.
```

Generador PDF (Reporte)

Estudia el siguiente código: Generador PDF (Reporte)

Código de Ejemplo

```
#!/usr/bin/python
# --- IMPORTANTE: EJECUCIÓN LOCAL REQUERIDA ---
# Este ejercicio requiere librerías externas o generar archivos.
# Por favor, ejecútalo en tu IDE local (VS Code, PyCharm, etc).
# ------------------------------------------------
# Generación de un Reporte PDF Simple
# REQUISITO: pip install fpdf2

from fpdf import FPDF

print("Generando reporte PDF...")

pdf = FPDF()
pdf.add_page()

# Título
# Usamos Helvetica que es una fuente core estándar para evitar
advertencias
pdf.set_font("helvetica", "B", 16)
pdf.cell(40, 10, "Reporte de Actividad")
pdf.ln(20)

# Contenido
pdf.set_font("helvetica", "", 12)
# ln=1 está deprecado en fpdf2, usamos new_x="LMARGIN", new_y="NEXT"
pdf.cell(0, 10, "Este es un documento generado automáticamente con
Python.", new_x="LMARGIN", new_y="NEXT")
pdf.cell(0, 10, "Podemos añadir texto, líneas y números.",
new_x="LMARGIN", new_y="NEXT")

for i in range(1, 6):
    pdf.cell(0, 10, f"Línea de detalle número {i}", new_x="LMARGIN",
new_y="NEXT")
```

```
filename = "reporte_simple.pdf"
pdf.output(filename)
print(f"¡PDF guardado como {filename}!")

# --- EJERCICIO INTERACTIVO ---
# 1. Ejecuta el código en tu IDE local.
# 2. Añade una línea final que diga "Fin del Reporte" en negrita.
```

Texto a Voz Simple

Estudia el siguiente código: Texto a Voz Simple

Código de Ejemplo

```
#!/usr/bin/python
# --- IMPORTANTE: EJECUCIÓN LOCAL REQUERIDA ---
# Este ejercicio requiere librerías externas o generar archivos.
# Por favor, ejecútalo en tu IDE local (VS Code, PyCharm, etc).
# ------------------------------------------------
# Texto a Voz Simple
# REQUISITO: pip install pyttsx3

import pyttsx3

engine = pyttsx3.init() # creación del objeto

""" VELOCIDAD """
rate = engine.getProperty('rate')   # obteniendo detalles de la
velocidad actual, por defecto es 200
#engine.setProperty('rate', 125) # Configurando nueva velocidad de voz

""" VOLUMEN """
volume = engine.getProperty('volume')   # conociendo el nivel de
volumen actual (min=0 y max=1)
engine.setProperty('volume',0.9)    # configurando nivel de volumen
entre 0 y 1
```

```
""" VOZ """
voices = engine.getProperty('voices')       # obteniendo detalles de
la voz actual
# engine.setProperty('voice', voices[0].id)  # cambiando índice,
cambia voces. 0 para hombre
# engine.setProperty('voice', voices[1].id)   # cambiando índice,
cambia voces. 1 para mujer

engine.say("¡Hola Mundo!")
engine.say('Mi velocidad actual es ' + str(rate))
engine.runAndWait()
engine.stop()

""" Guardando Voz en un archivo """
# En linux asegúrate de que 'espeak' y 'ffmpeg' están instalados
engine.save_to_file('Hola Mundo', 'test.mp3')
engine.runAndWait()

'''
voices = engine.getProperty('voices')
for voice in voices:
    print("Voz: %s" % voice.name)
    print(" - ID: %s" % voice.id)
    print(" - Idiomas: %s" % voice.languages)
    print(" - Género: %s" % voice.gender)
    print(" - Edad: %s" % voice.age)
    print("\n")

# https://pyttsx3.readthedocs.io/en/latest/
# https://pyttsx3.readthedocs.io/en/latest/engine.html#examples
'''



# --- EJERCICIO INTERACTIVO ---
# 1. Ejecuta el código en tu IDE local.
```

Espiral Arcoíris (SVG Turtle)

Estudia el siguiente código: Espiral Arcoíris (SVG Turtle)

Código de Ejemplo

```
#!/usr/bin/python
# --- IMPORTANTE: EJECUCIÓN LOCAL REQUERIDA ---
# Este ejercicio requiere librerías externas o generar archivos.
# Por favor, ejecútalo en tu IDE local (VS Code, PyCharm, etc).
# ------------------------------------------------------
# Espiral Arcoíris (SVG Turtle)
# REQUISITO: pip install svg_turtle

from svg_turtle import SvgTurtle

t = SvgTurtle(width=500, height=500)
t.bgcolor('black')
t.speed = 0 # Ignorado en SVG pero mantenido por compatibilidad

colors = ['red', 'purple', 'blue', 'green', 'orange', 'yellow']

print('Generando espiral...')
for x in range(360):
    t.pencolor(colors[x % 6])
    t.width = x / 100 + 1
    t.forward(x)
    t.left(59)

filename = 'espiral_turtle.svg'
t.save_as(filename)
print(f'¡Dibujo guardado como {filename}!')

# --- EJERCICIO INTERACTIVO ---
# 1. Ejecuta el código en tu IDE local.
# 2. Cambia el ángulo t.left(59) a 90 o 91 para ver patrones
diferentes.
```

Triángulo de Sierpinski (SVG)

Estudia el siguiente código: Triángulo de Sierpinski (SVG)

Código de Ejemplo

```
#!/usr/bin/python
# --- IMPORTANTE: EJECUCIÓN LOCAL REQUERIDA ---
# Este ejercicio requiere librerías externas o generar archivos.
# Por favor, ejecútalo en tu IDE local (VS Code, PyCharm, etc).
# ------------------------------------------------------
# Triángulo de Sierpinski (SVG)
# REQUISITO: pip install svg_turtle

from svg_turtle import SvgTurtle

def draw_triangle(t, points, color):
    # Dibujo simple de triángulo con líneas
    t.pencolor(color)
    t.penup()
    t.goto(points[0][0], points[0][1])
    t.pendown()
    t.goto(points[1][0], points[1][1])
    t.goto(points[2][0], points[2][1])
    t.goto(points[0][0], points[0][1])

def get_mid(p1, p2):
    return ((p1[0] + p2[0]) / 2, (p1[1] + p2[1]) / 2)

def sierpinski(points, degree, t):
    colormap = ['blue', 'red', 'green', 'white', 'yellow', 'violet',
'orange']
    draw_triangle(t, points, colormap[degree])

    if degree > 0:
        sierpinski([points[0],
                    get_mid(points[0], points[1]),
                    get_mid(points[0], points[2])],
                   degree-1, t)
        sierpinski([points[1],
```

```
                    get_mid(points[0], points[1]),
                    get_mid(points[1], points[2])],
                   degree-1, t)
        sierpinski([points[2],
                    get_mid(points[2], points[1]),
                    get_mid(points[0], points[2])],
                   degree-1, t)

t = SvgTurtle(width=400, height=400)
points = [[-100, -50], [0, 100], [100, -50]]

print('Generando Sierpinski...')
sierpinski(points, 3, t)

filename = 'sierpinski.svg'
t.save_as(filename)

# --- EJERCICIO INTERACTIVO ---
# 1. Ejecuta el código en tu IDE local.
# 2. Cambia el grado de recursión de 3 a 4 (cuidado, crece
exponencialmente).
```

Arte Geométrico (SVG)

Estudia el siguiente código: Arte Geométrico (SVG)

Código de Ejemplo

```
#!/usr/bin/python
# --- IMPORTANTE: EJECUCIÓN LOCAL REQUERIDA ---
# Este ejercicio requiere librerías externas o generar archivos.
# Por favor, ejecútalo en tu IDE local (VS Code, PyCharm, etc).
# ------------------------------------------------
# Arte Geométrico (SVG)
# REQUISITO: pip install svg_turtle

from svg_turtle import SvgTurtle
```

```
def draw_square(t):
    for i in range(4):
        t.forward(100)
        t.right(90)

t = SvgTurtle(width=500, height=500)
t.pencolor('blue')

print('Generando arte geométrico...')
for i in range(36):
    draw_square(t)
    t.right(10)

filename = 'arte_geo.svg'
t.save_as(filename)

# --- EJERCICIO INTERACTIVO ---
# 1. Ejecuta el código en tu IDE local.
# 2. Cambia el ángulo de rotación o el tamaño del cuadrado.
```

Cuadrado y Círculo (SVG)

Estudia el siguiente código: Cuadrado y Círculo (SVG)

Código de Ejemplo

```
#!/usr/bin/python
# --- IMPORTANTE: EJECUCIÓN LOCAL REQUERIDA ---
# Este ejercicio requiere librerías externas o generar archivos.
# Por favor, ejecútalo en tu IDE local (VS Code, PyCharm, etc).
# ------------------------------------------------------
# Cuadrado y Círculo (SVG)
# REQUISITO: pip install svg_turtle

from svg_turtle import SvgTurtle
import math
```

```
def draw_circle_approx(t, radius):
    # Aproximación de círculo con líneas
    circumference = 2 * math.pi * radius
    step_length = circumference / 36
    for _ in range(36):
        t.forward(step_length)
        t.right(10)

t = SvgTurtle(width=400, height=400)

# Cuadrado
t.pencolor("red")
for _ in range(4):
    t.forward(100)
    t.right(90)

# Círculo (Aproximado)
t.penup()
t.goto(50, 50)
t.pendown()
t.pencolor("blue")
draw_circle_approx(t, 50)

filename = "shapes.svg"
t.save_as(filename)

# --- EJERCICIO INTERACTIVO ---
# 1. Ejecuta el código en tu IDE local.
```

Cuadrado Simple (SVG)

Estudia el siguiente código: Cuadrado Simple (SVG)

Código de Ejemplo

```
#!/usr/bin/python
# --- IMPORTANTE: EJECUCIÓN LOCAL REQUERIDA ---
# Este ejercicio requiere librerías externas o generar archivos.
# Por favor, ejecútalo en tu IDE local (VS Code, PyCharm, etc).
# ------------------------------------------------------
# Cuadrado Simple (SVG)
# REQUISITO: pip install svg_turtle

from svg_turtle import SvgTurtle

t = SvgTurtle()
t.pencolor("green")

print("Dibujando cuadrado...")
t.forward(100)
t.right(90)
t.forward(100)
t.right(90)
t.forward(100)
t.right(90)
t.forward(100)
t.right(90)

t.save_as("cuadrado.svg")

# --- EJERCICIO INTERACTIVO ---
# 1. Ejecuta el código en tu IDE local.
```

Colores en Terminal (Colorama)

Estudia el siguiente código: Colores en Terminal (Colorama)

Código de Ejemplo

```
#!/usr/bin/python
# --- IMPORTANTE: EJECUCIÓN LOCAL REQUERIDA ---
# Este ejercicio requiere librerías externas o generar archivos.
# Por favor, ejecútalo en tu IDE local (VS Code, PyCharm, etc).
# ------------------------------------------------------
# Colores en Terminal con Colorama
# REQUISITO: pip install colorama

from colorama import init, Fore, Back, Style
init(autoreset=True)


print(Fore.RED + 'texto rojo')
print(Fore.GREEN + 'texto verde')
print(Fore.BLUE + 'texto azul')
print(Fore.CYAN + 'texto cian')
print(Fore.MAGENTA + 'texto magenta')
print(Back.GREEN + 'y con fondo verde')
print(Style.DIM + 'y en texto tenue')
print(Style.BRIGHT + Fore.GREEN + 'y color verde en texto brillante')
print('automáticamente de vuelta al color por defecto')



# Estos son los colores disponibles
"""
Fore: BLACK, RED, GREEN, YELLOW, BLUE, MAGENTA, CYAN, WHITE, RESET.
Back: BLACK, RED, GREEN, YELLOW, BLUE, MAGENTA, CYAN, WHITE, RESET.
Style: DIM, NORMAL, BRIGHT, RESET_ALL
"""


# --- EJERCICIO INTERACTIVO ---
# 1. Ejecuta el código en tu IDE local.
```

Descargar Archivos

Estudia el siguiente código: Descargar Archivos

Código de Ejemplo

```
import sys
import requests
import re
from colorama import init, Fore, Back, Style
import validators

# --- IMPORTANTE: EJECUCIÓN LOCAL REQUERIDA ---
# Este ejercicio requiere librerías externas o generar archivos.
# Por favor, ejecútalo en tu IDE local (VS Code, PyCharm, etc).
# ----------------------------------------------------
# PROPÓSITO de este script
# Obtener una url con varios formatos de archivos para descargar

init(autoreset=True)

# --- CORRECCIÓN PARA EJECUCIÓN WEB ---
# Simulando argumentos de línea de comandos si no se proporcionan
if len(sys.argv) < 2:
    print(Fore.YELLOW + "No se proporcionaron argumentos. Usando URL
por defecto para demo.")
    sys.argv.append("https://www.python.org/static/img/python-
logo.png")
# --------------------------

file_url = sys.argv[1]

if not validators.url(file_url):
    print(Fore.RED + Style.BRIGHT + "URL no válida")
    # exit() eliminado para prevenir caída del servidor
else:
    # extraer nombre de archivo de la url
    matched_file = re.search("(?=[\\w\\d-]+\\.\\w{3,4}$).+", file_url)
    if matched_file is not None:
        file_name_with_extension = matched_file.group(0)
```

```
    else:
        file_name_with_extension = "downloaded_file.png"

    print("Descargando %s" % file_name_with_extension)

    try:
        response = requests.get(file_url, stream = True)
        with open(file_name_with_extension, "wb") as file_download:
            total_length = response.headers.get('content-length')
            if total_length:
                total_length = int(total_length)
                print("El tamaño total del archivo es: {0:.3f}
KB".format(total_length/1024))
            else:
                print("Tamaño total desconocido.")
                total_length = 1000 # Valor dummy

            dl_bar = 0
            for chunk in response.iter_content(chunk_size = 1024):
                if chunk:
                    dl_bar += len(chunk)
                    file_download.write(chunk)
                    done = int(50 * dl_bar / total_length)
                    sys.stdout.write(Fore.GREEN + Style.BRIGHT +
"\r[%s%s]" % ('=' * done, ' ' * (50-done)) )
                    sys.stdout.flush()
        print("\n¡Descarga Completa! \n")
    except Exception as e:
        print(f"Error descargando: {e}")

# --- EJERCICIO INTERACTIVO ---
# 1. Ejecuta el código en tu IDE local.
```

Generar OTP

Estudia el siguiente código: Generar OTP

Código de Ejemplo

```
#!/usr/bin/python
# Generar OTP de 6 dígitos
import string
import secrets

number = string.digits
otp = ''

for i in range(6):
    otp += ''.join(secrets.choice(number))

print(f'Tu OTP es: {otp}')

# --- EJERCICIO INTERACTIVO ---
# 1. Ejecuta el código en tu IDE local.
```

Generador de Contraseñas

Estudia el siguiente código: Generador de Contraseñas

Código de Ejemplo

```
import string
import secrets

def get_alphabet():
    letters = string.ascii_letters
    digits = string.digits
    special_chars = string.punctuation
    alphabet = letters + digits + special_chars
    return alphabet

# Restricciones de contraseña
password_len = 12
```

```
print(f"Generando contraseña de longitud: {password_len}")

if password_len <8:
    print("La longitud debe ser de al menos 8 caracteres")
else:
    password = ''
    for i in range(password_len):
        password += ''.join(secrets.choice(get_alphabet()))

    print(f"La contraseña es: {password}")

# --- EJERCICIO INTERACTIVO ---
# 1. Ejecuta el código tal como está para ver el resultado.
# 2. Modifica el código para cambiar el comportamiento.
```

Extraer Links de Web

Estudia el siguiente código: Extraer Links de Web

Código de Ejemplo

```
import requests
import sys
import re
from bs4 import BeautifulSoup as bs
import validators

# --- IMPORTANTE: EJECUCIÓN LOCAL REQUERIDA ---
# Este ejercicio requiere librerías externas o generar archivos.
# Por favor, ejecútalo en tu IDE local (VS Code, PyCharm, etc).
# ------------------------------------------------

# --- CORRECCIÓN PARA EJECUCIÓN WEB ---
if len(sys.argv) < 2:
    print("No se proporcionó URL. Usando por defecto: https://
www.python.org")
    sys.argv.append("https://www.python.org")
```

```
# -------------------------

url = sys.argv[1]

if not validators.url(url):
    print("URL no válida")
else:
    try:
        print(f"Obteniendo {url}...")
        get_content = requests.get(url)
        getpage_soup = bs(get_content.text, 'html.parser')

        all_links= getpage_soup.findAll('a', attrs={'href' :
re.compile("^https?://")})

        domain_name = url.split('//')[1]
        print(f"Nombre de dominio: {domain_name}")

        print(f"Encontrados {len(all_links)} enlaces. Mostrando los
primeros 5:")
        for i, link in enumerate(all_links[:5]):
            print(f"- {link.get('href')}")

    except Exception as e:
        print(f"Error: {e}")

# --- EJERCICIO INTERACTIVO ---
# 1. Ejecuta el código en tu IDE local.
```

Info Usuario GitHub

Estudia el siguiente código: Info Usuario GitHub

Código de Ejemplo

```
import json
import requests
```

```
import sys

# --- IMPORTANTE: EJECUCIÓN LOCAL REQUERIDA ---
# Este ejercicio requiere librerías externas o generar archivos.
# Por favor, ejecútalo en tu IDE local (VS Code, PyCharm, etc).
# ---------------------------------------------------

# --- CORRECCIÓN PARA EJECUCIÓN WEB ---
if len(sys.argv) < 2:
    print("No se proporcionó usuario. Usando por defecto: octocat")
    sys.argv.append("octocat")
# -------------------------

username = sys.argv[1]
api_url_base = 'https://api.github.com/'
headers = {'Content-Type': 'application/json', 'User-Agent': 'Python
Student', 'Accept': 'application/vnd.github.v3+json'}

def get_user_details(username):
    user_url = '{}users/{}'.format(api_url_base, username)
    response = requests.get(user_url, headers=headers)
    if response.status_code == 200:
        return response.content
    else:
        return None

print(f"Obteniendo info para el usuario: {username}")
user_details = get_user_details(username)

if user_details:
    user_in_json = user_details.decode('utf-8')
    user_detail_dict = json.loads(user_in_json)
    print("="*10 + " Detalles de Usuario " + "="*10)
    print("Nombre: {}".format(user_detail_dict.get('name', 'N/A')))
    print("Bio: {}".format(user_detail_dict.get('bio', 'N/A')))
    print("Ubicación: {}".format(user_detail_dict.get('location', 'N/
A')))
    print("Repos Públicos:
{}".format(user_detail_dict.get('public_repos', 'N/A')))
else:
    print("Usuario no encontrado.")
```

```
# --- EJERCICIO INTERACTIVO ---
# 1. Ejecuta el código en tu IDE local.
```

Convertir HEIC a PNG

Estudia el siguiente código: Convertir HEIC a PNG

Código de Ejemplo

```
#!/usr/bin/python
# --- IMPORTANTE: EJECUCIÓN LOCAL REQUERIDA ---
# Este ejercicio requiere librerías externas o generar archivos.
# Por favor, ejecútalo en tu IDE local (VS Code, PyCharm, etc).
# ----------------------------------------------------
# Convierte imágenes HEIC a PNG usando pillow-heif
# REQUISITO: pip install pillow-heif

from PIL import Image
import os

# Intentamos importar pillow_heif, si no está, avisamos
try:
    from pillow_heif import register_heif_opener
    register_heif_opener()
    HAS_HEIF = True
except ImportError:
    HAS_HEIF = False
    print("NOTA: Instala 'pillow-heif' para soporte real de HEIC.")

def convert_heic_to_png(heic_path):
    if not os.path.exists(heic_path):
        print(f"Archivo no encontrado: {heic_path}")
        return

    if not HAS_HEIF:
        print("Librería pillow-heif no encontrada. No se puede
convertir.")
```

```
        return

    print(f"Convirtiendo {heic_path}...")
    try:
        image = Image.open(heic_path)
        png_path = heic_path.lower().replace(".heic", ".png")
        image.save(png_path, format="PNG")
        print(f"Guardado como: {png_path}")
    except Exception as e:
        print(f"Error al convertir: {e}")

# Creamos un archivo dummy para probar si no existe
if not os.path.exists("test.heic"):
    print("Creando archivo dummy test.heic para demostración...")
    # En realidad creamos un JPG y lo renombramos para que el script
intente abrirlo
    # (Fallará si es un JPG real pero esperamos HEIC, pero sirve para
demo de flujo)
    img = Image.new('RGB', (100, 100), color = 'red')
    img.save('test.heic') # Esto no es un HEIC válido, pero permite
correr el script

convert_heic_to_png("test.heic")

# --- EJERCICIO INTERACTIVO ---
# 1. Instala la librería: pip install pillow-heif
# 2. Usa una foto real de iPhone (.HEIC) y prueba el script.
```

Rangos IP (JSON)

Estudia el siguiente código: Rangos IP (JSON)

Código de Ejemplo

```
import json

# Datos truncados para brevedad
```

```
data = {
  "syncToken": "1721411590",
  "createDate": "2024-07-19-17-53-10",
  "prefixes": [
    {
      "ip_prefix": "3.5.140.0/22",
      "region": "ap-northeast-2",
      "service": "AMAZON",
      "network_border_group": "ap-northeast-2"
    },
    {
      "ip_prefix": "13.34.1.109/32",
      "region": "ap-southeast-2",
      "service": "AMAZON",
      "network_border_group": "ap-southeast-2"
    }
  ]
}

print(json.dumps(data, indent=2))

# --- EJERCICIO INTERACTIVO ---
# 1. Ejecuta el código tal como está para ver el resultado.
```

Scraping con Selenium

Estudia el siguiente código: Scraping con Selenium

Código de Ejemplo

```
#!/usr/bin/python
# --- IMPORTANTE: EJECUCIÓN LOCAL REQUERIDA ---
# Este ejercicio requiere librerías externas o generar archivos.
# Por favor, ejecútalo en tu IDE local (VS Code, PyCharm, etc).
# -----------------------------------------------------
# Scraping real con Selenium
# REQUISITO: pip install selenium webdriver-manager
```

```
import time
from selenium import webdriver
from selenium.webdriver.chrome.service import Service
from webdriver_manager.chrome import ChromeDriverManager
from selenium.webdriver.common.by import By

def run_scraper():
    print("Iniciando Chrome...")
    # Configuración para modo headless (sin ventana gráfica)
    options = webdriver.ChromeOptions()
    options.add_argument("--headless")
    options.add_argument("--no-sandbox")
    options.add_argument("--disable-dev-shm-usage")

    try:
        # Instalación automática del driver
        service = Service(ChromeDriverManager().install())
        driver = webdriver.Chrome(service=service, options=options)

        url = "https://books.toscrape.com/"
        print(f"Navegando a {url}...")
        driver.get(url)

        print("Extrayendo títulos de libros...")
        # Buscamos elementos h3 > a
        books = driver.find_elements(By.CSS_SELECTOR, "h3 a")

        for i, book in enumerate(books[:5], 1):
            title = book.get_attribute("title")
            print(f"{i}. {title}")

        driver.quit()
        print("¡Scraping completado!")

    except Exception as e:
        print(f"Error al ejecutar Selenium: {e}")
        print("Asegúrate de tener Chrome instalado en tu sistema.")

# Ejecutar
run_scraper()
```

```
# --- EJERCICIO INTERACTIVO ---
# 1. Este código requiere un entorno local con Chrome instalado.
# 2. Instala las librerías necesarias y ejecútalo en tu IDE.
```

Análisis de Datos CSV

Estudia el siguiente código: Análisis de Datos CSV

Código de Ejemplo

```
#!/usr/bin/python
# --- IMPORTANTE: EJECUCIÓN LOCAL REQUERIDA ---
# Este ejercicio requiere librerías externas o generar archivos.
# Por favor, ejecútalo en tu IDE local (VS Code, PyCharm, etc).
# ------------------------------------------------------
# Analizar un archivo CSV real
import csv
import os

# Creamos un archivo CSV de prueba
csv_file = "datos_ventas.csv"
with open(csv_file, "w", newline="") as f:
    writer = csv.writer(f)
    writer.writerow(["Producto", "Precio", "Cantidad"])
    writer.writerow(["Laptop", 1200, 5])
    writer.writerow(["Mouse", 25, 50])
    writer.writerow(["Teclado", 45, 30])
    writer.writerow(["Monitor", 300, 10])

print(f"Analizando {csv_file}...")

total_ventas = 0
productos_stock = 0

with open(csv_file, "r") as f:
    reader = csv.DictReader(f)
    for row in reader:
```

```
        precio = float(row["Precio"])
        cantidad = int(row["Cantidad"])
        total = precio * cantidad
        total_ventas += total
        productos_stock += cantidad
        print(f"Producto: {row['Producto']} - Total: ${total}")

print("-" * 30)
print(f"Ventas Totales: ${total_ventas}")
print(f"Total Productos: {productos_stock}")

# Limpieza
os.remove(csv_file)

# --- EJERCICIO INTERACTIVO ---
# 1. Modifica el código para calcular el precio promedio.
# 2. Añade más productos al archivo CSV.
```

Manipulación de PDF

Estudia el siguiente código: Manipulación de PDF

Código de Ejemplo

```
#!/usr/bin/python
# --- IMPORTANTE: EJECUCIÓN LOCAL REQUERIDA ---
# Este ejercicio requiere librerías externas o generar archivos.
# Por favor, ejecútalo en tu IDE local (VS Code, PyCharm, etc).
# -------------------------------------------------
# Manipulación real de PDF usando pypdf
# REQUISITO: pip install pypdf

from pypdf import PdfReader, PdfWriter
import os

# Crear un PDF simple para la demo (usando fpdf2 si está disponible, o
simulando creación)
```

```
try:
    from fpdf import FPDF
    pdf = FPDF()
    pdf.add_page()
    pdf.set_font("helvetica", size=12)
    pdf.cell(text="Hola Mundo PDF!", new_x="LMARGIN", new_y="NEXT")
    pdf.output("demo.pdf")
    HAS_FPDF = True
except ImportError:
    HAS_FPDF = False
    print("Instala fpdf2 para generar el PDF de prueba.")

if os.path.exists("demo.pdf"):
    print("Leyendo demo.pdf...")
    reader = PdfReader("demo.pdf")
    print(f"Número de páginas: {len(reader.pages)}")

    page = reader.pages[0]
    text = page.extract_text()
    print("Texto extraído:")
    print(text)

    # Crear un nuevo PDF con solo la primera página
    writer = PdfWriter()
    writer.add_page(page)

    with open("demo_copia.pdf", "wb") as f:
        writer.write(f)
    print("Copia guardada como demo_copia.pdf")
else:
    print("No se encontró demo.pdf para leer.")

# --- EJERCICIO INTERACTIVO ---
# 1. Instala pypdf: pip install pypdf
# 2. Prueba con un PDF real de tu ordenador.
```

PDF a Imágenes

Estudia el siguiente código: PDF a Imágenes

Código de Ejemplo

```
#!/usr/bin/python
# --- IMPORTANTE: EJECUCIÓN LOCAL REQUERIDA ---
# Este ejercicio requiere librerías externas o generar archivos.
# Por favor, ejecútalo en tu IDE local (VS Code, PyCharm, etc).
# ----------------------------------------------------
# Conversión real de PDF a Imagen
# REQUISITO: pip install pdf2image
# REQUISITO: Poppler instalado en el sistema (apt-get install poppler-
utils)

import os

try:
    from pdf2image import convert_from_path
    HAS_LIB = True
except ImportError:
    HAS_LIB = False
    print("Librería pdf2image no encontrada.")

if HAS_LIB and os.path.exists("demo.pdf"):
    print("Convirtiendo demo.pdf a imágenes...")
    try:
        images = convert_from_path("demo.pdf")

        for i, image in enumerate(images):
            image_name = f"pagina_{i+1}.jpg"
            image.save(image_name, "JPEG")
            print(f"Guardada: {image_name}")

    except Exception as e:
        print(f"Error (posiblemente falta Poppler): {e}")
else:
    print("Saltando conversión (falta librería o archivo PDF).")
```

```
# --- EJERCICIO INTERACTIVO ---
# 1. Este ejercicio requiere dependencias del sistema (Poppler).
# 2. Es ideal para ejecutar en tu máquina local.
```

Validador Regex

Estudia el siguiente código: Validador Regex

Código de Ejemplo

```
#!/usr/bin/python
# pip install regex
import re

print("Iniciando Validador Regex...")

def validate_phone_number():
    phone = "123-456-7890"
    pattern = r"\d{3}-\d{3}-\d{4}"
    if re.match(pattern, phone):
        print(f"Teléfono {phone} es válido")
    else:
        print(f"Teléfono {phone} es inválido")

def validate_username():
    user = "User_123"
    pattern = r"^[a-zA-Z0-9_]+$"
    if re.match(pattern, user):
        print(f"Usuario {user} es válido")
    else:
        print(f"Usuario {user} es inválido")

# Ejecutar validaciones
validate_phone_number()
validate_username()
print("Validación completa.")
```

```
# --- EJERCICIO INTERACTIVO ---
# 1. Ejecuta el código tal como está para ver el resultado.
# 2. Modifica el código para cambiar el comportamiento.
#    (Pista: Intenta cambiar los valores de las variables o la lógica)
# 3. Vuelve a ejecutar para ver tus cambios.
```

Renombrar Archivos Masivamente

Estudia el siguiente código: Renombrar Archivos Masivamente

Código de Ejemplo

```
#!/usr/bin/python
# --- IMPORTANTE: EJECUCIÓN LOCAL REQUERIDA ---
# Este ejercicio requiere librerías externas o generar archivos.
# Por favor, ejecútalo en tu IDE local (VS Code, PyCharm, etc).
# ------------------------------------------------------
# Script real para renombrar archivos
import os
import shutil

# Configuración
dir_objetivo = "fotos_viaje_test"

# 1. Preparación: Crear directorio y archivos de prueba
if not os.path.exists(dir_objetivo):
    os.makedirs(dir_objetivo)
    # Crear archivos dummy
    open(f"{dir_objetivo}/101_playa.jpg", "w").close()
    open(f"{dir_objetivo}/102_montana.jpg", "w").close()
    open(f"{dir_objetivo}/103_cena.jpg", "w").close()
    print(f"Archivos creados en {dir_objetivo}")

# 2. Renombrado Real
print("\nIniciando renombrado...")
archivos = os.listdir(dir_objetivo)
```

```
for nombre_archivo in archivos:
    ruta_completa = os.path.join(dir_objetivo, nombre_archivo)

    if os.path.isfile(ruta_completa):
        # Lógica: Eliminar dígitos y guiones bajos al inicio
        nuevo_nombre = "".join([c for c in nombre_archivo if not
c.isdigit()]).lstrip("_")

        nueva_ruta = os.path.join(dir_objetivo, nuevo_nombre)

        # Renombrar
        os.rename(ruta_completa, nueva_ruta)
        print(f"Renombrado: {nombre_archivo} -> {nuevo_nombre}")

print("\nVerificando resultados:")
print(os.listdir(dir_objetivo))

# Limpieza (opcional)
# shutil.rmtree(dir_objetivo)

# --- EJERCICIO INTERACTIVO ---
# 1. Ejecuta el código. Verás que realmente crea y renombra archivos.
# 2. Modifica la lógica para añadir un prefijo "Vacaciones_" a todos.
```

Información del Sistema

Estudia el siguiente código: Información del Sistema

Código de Ejemplo

```
#!/usr/bin/python
import platform
import sys
import os
import socket
import time
```

```
# Este script está probado en MacOS y funcionó bien. Debería funcionar
en Linux también.

unumber = os.getuid()
pnumber = os.getpid()
where   = os.getcwd()
now     = time.time()
means   = time.ctime(now)

print ("Número de usuario",unumber)
print ("ID de proceso",pnumber)
print ("Directorio actual",where)

print ("Nombre: " +socket.gethostname( ))
print ("Plataforma del sistema: "+sys.platform)
print ("Nodo " +platform.node())
print ("Plataforma: "+platform.platform())
print ("SO del sistema: "+platform.system())
print ("Release: " +platform.release())
print ("Versión: " +platform.version())
print ("Versión de Python: " +platform.python_version())

# mostrando arquitectura de la plataforma
print('Arquitectura de la plataforma:', platform.architecture())
# mostrando procesador de la plataforma
print('Procesador de la plataforma:', platform.processor())
# mostrando tipo de máquina
print('Tipo de máquina:', platform.machine())


# --- EJERCICIO INTERACTIVO ---
# 1. Ejecuta el código tal como está para ver el resultado.
# 2. Modifica el código para cambiar el comportamiento.
#    (Pista: Intenta cambiar los valores de las variables o la lógica)
# 3. Vuelve a ejecutar para ver tus cambios.
```

Recordatorio de Descanso

Estudia el siguiente código: Recordatorio de Descanso

Código de Ejemplo

```
#!/usr/bin/python
# --- IMPORTANTE: EJECUCIÓN LOCAL REQUERIDA ---
# Este ejercicio requiere librerías externas o generar archivos.
# Por favor, ejecútalo en tu IDE local (VS Code, PyCharm, etc).
# ------------------------------------------------------
# Recordatorio real usando webbrowser
import time
import webbrowser

print("Iniciando monitor de descansos...")
print("Para la demo, esperaremos solo 3 segundos.")

# Esperar (simulando trabajo)
time.sleep(3)

print("¡Hora del descanso!")
url = "https://www.google.com/search?q=ejercicios+de+estiramiento"

# Abrir navegador real
# Nota: Esto intentará abrir un navegador en el servidor si se ejecuta
aquí.
# En tu máquina local, abrirá tu Chrome/Firefox.
try:
    webbrowser.open(url)
    print(f"Se ha abierto el navegador en: {url}")
except Exception as e:
    print(f"No se pudo abrir el navegador: {e}")

# --- EJERCICIO INTERACTIVO ---
# 1. Ejecuta este código en tu IDE local.
# 2. Verás que realmente abre tu navegador web.
```

Validador de Contraseñas

Estudia el siguiente código: Validador de Contraseñas

Código de Ejemplo

```
#!/usr/bin/python
import re

# Criterios de Contraseña
# 1. Debe contener alfanuméricos
# 2. Al menos una Letra Mayúscula
# 3. Al menos una letra minúscula
# 4. 8-20 caracteres
# 5. al menos un caracter especial !@#$%^&*_
# 6. Sin espacios en blanco por favor

passwords = ['JassiSidhu', 'Jassi Sidhu0$','JassiSidhu0$',
'Jalantu_123*', '12Falcon#', 'Sh0rt5!','Sh0rt5!89***^AWS+', 'short']

# Usando Expresiones Regulares
pattern = '^((?=.*\\d)(?=.*[a-z])(?=.*[A-Z])(?=.*[!@#$%^&*_])(?=.*[\
\S])[0-9a-zA-Z!@#$%^&*_]{8,20})$'
for password in passwords:
    print(f"Probando: {password}")
    result = re.match(pattern,password)
    if result:
        print("  -> Contraseña Aceptada")
    else:
        print("  -> Contraseña Denegada")


# --- EJERCICIO INTERACTIVO ---
# 1. Ejecuta el código tal como está para ver el resultado.
# 2. Modifica el código para cambiar el comportamiento.
#    (Pista: Intenta cambiar los valores de las variables o la lógica)
# 3. Vuelve a ejecutar para ver tus cambios.
```

Duración de Video

Estudia el siguiente código: Duración de Video

Código de Ejemplo

```
#!/usr/bin/python
# --- IMPORTANTE: EJECUCIÓN LOCAL REQUERIDA ---
# Este ejercicio requiere librerías externas o generar archivos.
# Por favor, ejecútalo en tu IDE local (VS Code, PyCharm, etc).
# ----------------------------------------------------
# Cálculo real de duración de video usando moviepy
# REQUISITO: pip install moviepy

import os

try:
    from moviepy.editor import VideoFileClip
    HAS_MOVIEPY = True
except ImportError:
    HAS_MOVIEPY = False
    print("Instala moviepy para analizar videos reales.")

def analizar_video(ruta_video):
    if not HAS_MOVIEPY:
        return

    if not os.path.exists(ruta_video):
        print(f"Video no encontrado: {ruta_video}")
        return

    try:
        clip = VideoFileClip(ruta_video)
        duracion = clip.duration
        print(f"Video: {ruta_video}")
        print(f"Duración: {duracion:.2f} segundos")
        print(f"Resolución: {clip.size}")
        clip.close()
    except Exception as e:
        print(f"Error al leer video: {e}")
```

```
# Demo: Crear un archivo dummy para que no falle la lógica de archivo
if not os.path.exists("demo.mp4"):
    print("No hay video demo.mp4. Si tuvieras uno, lo analizaríamos
así:")
    print("analizar_video('demo.mp4')")
else:
    analizar_video("demo.mp4")

# --- EJERCICIO INTERACTIVO ---
# 1. Instala moviepy: pip install moviepy
# 2. Coloca un archivo .mp4 en la carpeta y cambia el nombre en el
código.
```

Módulo 05: Ciberseguridad y Criptografía

Criptografía Básica

Estudia el siguiente código: Criptografía Básica

Código de Ejemplo

```
#!/usr/bin/python
# --- IMPORTANTE: EJECUCIÓN LOCAL REQUERIDA ---
# Este ejercicio requiere librerías externas o generar archivos.
# Por favor, ejecútalo en tu IDE local (VS Code, PyCharm, etc).
# ------------------------------------------------------
# Criptografía Básica con Fernet
# REQUISITO: pip install cryptography

from cryptography.fernet import Fernet

key = Fernet.generate_key()
cipher_suite = Fernet(key)
message = "Este es un mensaje secreto para un grupo secreto. No para
crackers, pero quizás para Criptoanalistas!.".encode()

cipher_text = cipher_suite.encrypt(message)
print('El texto cifrado es: ' + str(cipher_text))

plain_text = cipher_suite.decrypt(cipher_text)
print('Aquí está el texto plano:' + str(plain_text))

print('\nComprobando si el texto descifrado es igual a nuestro mensaje
o no')

if plain_text == message: print("Sí, ambos son iguales\n")
```

```
# --- EJERCICIO INTERACTIVO ---
# 1. Ejecuta el código en tu IDE local.
```

Hashing de Contraseñas

Estudia el siguiente código: Hashing de Contraseñas

Código de Ejemplo

```
#!/usr/bin/python
# Hashing de Contraseñas con hashlib
# Aprende cómo se almacenan las contraseñas de forma segura.

import hashlib

password = "MiPasswordSeguro123"
print(f"Contraseña original: {password}")

# 1. MD5 (Inseguro, rápido)
md5_hash = hashlib.md5(password.encode()).hexdigest()
print(f"MD5:    {md5_hash}")

# 2. SHA-256 (Estándar actual)
sha256_hash = hashlib.sha256(password.encode()).hexdigest()
print(f"SHA256: {sha256_hash}")

# 3. Salted Hash (Más seguro)
# Añadimos un valor aleatorio para evitar ataques de Rainbow Table
salt = "s4lT_r4nd0m"
salted_password = password + salt
salted_hash = hashlib.sha256(salted_password.encode()).hexdigest()
print(f"Salted: {salted_hash} (Salt: {salt})")

# --- EJERCICIO INTERACTIVO ---
# 1. Ejecuta el código.
```

```
# 2. Cambia la contraseña y observa cómo cambian los hashes
completamente (Efecto avalancha).
```

Cracking de Contraseñas (Hash)

Estudia el siguiente código: Cracking de Contraseñas (Hash)

Código de Ejemplo

```
#!/usr/bin/python
# Cracking de Contraseñas (Ataque de Diccionario/Fuerza Bruta)
# En un escenario real, el atacante obtiene el HASH de la contraseña,
no la contraseña en texto plano.

import hashlib
import time

# 1. El Hash robado (SHA-256 de un PIN de 4 dígitos)
# Supongamos que obtuvimos esto de una base de datos filtrada.
# (El hash corresponde a "4829")
TARGET_HASH =
"a2c4e6f323977e58455793f20e547622995a15a303863166544165576566143"

print("--- Iniciando Cracking de Hash SHA-256 ---")
print(f"Objetivo: Encontrar el PIN que genera el hash:
{TARGET_HASH[:10]}...")

start_time = time.time()
found = False

# 2. Ataque de Fuerza Bruta (Espacio de claves: 0000-9999)
for i in range(10000):
    # Generar candidato (ej. "0005")
    pin_candidate = f"{i:04d}"

    # Calcular hash del candidato
    candidate_hash =
```

```
hashlib.sha256(pin_candidate.encode()).hexdigest()

    # Comparar
    if candidate_hash == TARGET_HASH:
        end_time = time.time()
        print(f"\n[¡ÉXITO!] Contraseña encontrada: {pin_candidate}")
        print(f"Hash verificado: {candidate_hash}")
        print(f"Tiempo: {end_time - start_time:.4f} segundos")
        found = True
        break

    if i % 2000 == 0:
        print(f"Probando... {pin_candidate} ->
{candidate_hash[:10]}...")

if not found:
    print("\n[FALLO] No se encontró la contraseña en el rango
probado.")

# --- EJERCICIO INTERACTIVO ---
# 1. Ejecuta el código.
# 2. Genera un nuevo hash de otro PIN en el ejercicio anterior y
actualiza TARGET_HASH aquí para intentar crackearlo.
```

Inyección SQL (Demo SQLite)

Estudia el siguiente código: Inyección SQL (Demo SQLite)

Código de Ejemplo

```
#!/usr/bin/python
# Demostración de Inyección SQL usando SQLite en memoria

import sqlite3

# Configuración de la Base de Datos
conn = sqlite3.connect(":memory:")
```

```
cursor = conn.cursor()
cursor.execute("CREATE TABLE users (id INTEGER, username TEXT,
password TEXT)")
cursor.execute("INSERT INTO users VALUES (1, 'admin', 'admin123')")
cursor.execute("INSERT INTO users VALUES (2, 'user', 'pass')")

def login_vulnerable(user, pwd):
    print(f"\n[Intento de Login] User: {user} | Pass: {pwd}")
    # VULNERABLE: Concatenación directa de strings
    query = f"SELECT * FROM users WHERE username = '{user}' AND
password = '{pwd}'"
    print(f"[QUERY] {query}")

    try:
        cursor.execute(query)
        result = cursor.fetchone()
        if result:
            print(f"[RESULTADO] ¡Login Exitoso! Bienvenido
{result[1]}")
        else:
            print("[RESULTADO] Acceso Denegado")
    except Exception as e:
        print(f"[ERROR SQL] {e}")

# 1. Login Normal
login_vulnerable("admin", "admin123")

# 2. Login Fallido
login_vulnerable("admin", "wrongpass")

# 3. ATAQUE DE INYECCIÓN SQL
# Usamos ' OR '1'='1 para hacer la condición siempre verdadera
login_vulnerable("admin' OR '1'='1", "basura")

# --- EJERCICIO INTERACTIVO ---
# 1. Ejecuta el código para ver cómo funciona el ataque.
# 2. Intenta inyectar para loguearte como el usuario 'user'.
```

Detector de Phishing (URL)

Estudia el siguiente código: Detector de Phishing (URL)

Código de Ejemplo

```
#!/usr/bin/python
# Analizador de URLs para detectar Phishing

import re

def analizar_url(url):
    score = 0
    razones = []

    # 1. Longitud excesiva
    if len(url) > 75:
        score += 10
        razones.append("URL muy larga")

    # 2. Uso de dirección IP
    ip_regex = r"\d{1,3}\.\d{1,3}\.\d{1,3}\.\d{1,3}"
    if re.search(ip_regex, url):
        score += 20
        razones.append("Usa dirección IP en lugar de dominio")

    # 3. Uso de @ para ofuscar
    if "@" in url:
        score += 15
        razones.append("Contiene '@' (posible redirección)")

    # 4. Palabras clave sospechosas
    sospechosas = ["secure", "account", "update", "login", "bank"]
    for s in sospechosas:
        if s in url and "https" not in url:
            score += 5
            razones.append(f"Palabra clave '{s}' en sitio no seguro")

    return score, razones
```

```
urls = [
    "https://www.google.com",
    "http://192.168.1.1/login.php",
    "http://secure-bank-update.com.badsite.org/login",
    "https://google.com@malicious.com/file"
]

print("--- Detector de Phishing ---")
for url in urls:
    riesgo, detalles = analizar_url(url)
    print(f"\nURL: {url}")
    print(f"Riesgo: {riesgo}%")
    if detalles:
        print(f"Alertas: {detalles}")

# --- EJERCICIO INTERACTIVO ---
# 1. Ejecuta el código.
# 2. Añade más URLs para probar el detector.
```

Info de Cabeceras HTTP

Estudia el siguiente código: Info de Cabeceras HTTP

Código de Ejemplo

```
#!/usr/bin/python
# --- IMPORTANTE: EJECUCIÓN LOCAL REQUERIDA ---
# Este ejercicio requiere librerías externas o generar archivos.
# Por favor, ejecútalo en tu IDE local (VS Code, PyCharm, etc).
# ------------------------------------------------------
# Info de Cabeceras HTTP
# REQUISITO: pip install requests validators colorama

import requests
import sys
import validators
from colorama import init, Fore, Back, Style
```

```
init(autoreset=True)

# --- CORRECCIÓN PARA EJECUCIÓN WEB ---
if len(sys.argv) < 2:
    print(Fore.YELLOW + "No se proporcionó URL. Usando por defecto:
https://www.python.org")
    sys.argv.append("https://www.python.org")
# -------------------------

url = sys.argv[1]

if validators.url(url):
    try:
        print(f"Obteniendo cabeceras para: {url}")
        url_request = requests.get(url)

        print(Fore.MAGENTA + Style.BRIGHT + "\nCabeceras de Petición")
        for req_header, value in url_request.request.headers.items():
            print(f"{req_header}: {value}")

        print(Fore.CYAN + Style.BRIGHT + "\nCabeceras de Respuesta")
        for resp_header, value in url_request.headers.items():
            print(f"{resp_header}: {value}")

        print(Fore.GREEN + Style.DIM + "\n\n====== Hecho ====== \n")
    except Exception as e:
        print(f"Error: {e}")
else:
    print("No es una url válida")

# --- EJERCICIO INTERACTIVO ---
# 1. Ejecuta el código en tu IDE local.
```

Cabeceras de Seguridad

Estudia el siguiente código: Cabeceras de Seguridad

Código de Ejemplo

```
#!/usr/bin/python
# --- IMPORTANTE: EJECUCIÓN LOCAL REQUERIDA ---
# Este ejercicio requiere librerías externas o generar archivos.
# Por favor, ejecútalo en tu IDE local (VS Code, PyCharm, etc).
# ----------------------------------------------------
# Cabeceras de Seguridad
# REQUISITO: pip install requests validators colorama

import requests
import sys
import validators
from colorama import init, Fore, Back, Style

init(autoreset=True)

# --- CORRECCIÓN PARA EJECUCIÓN WEB ---
if len(sys.argv) < 2:
    print(Fore.YELLOW + "No se proporcionó URL. Usando por defecto:
https://www.python.org")
    sys.argv.append("https://www.python.org")
# --------------------------

try:
    url = sys.argv[1]
    if(validators.url(url)):
        user_agent = 'Mozilla/5.0 (Windows NT 10.0; Win64; x64)
AppleWebKit/537.36 (KHTML, like Gecko) Chrome/78.0.3904.108 Safari/
537.36'
        headers = {'User-Agent': user_agent}

        session = requests.Session()
        url_request = session.get(url, headers = headers)

        print("Imprimiendo cabeceras de respuesta de url: " +
Fore.GREEN + Style.BRIGHT + url )
        print("Método de Petición:
{}".format(url_request.request.method))

        response_security_headers = ['Server', 'Content-Type', 'Via',
```

```
'X-Frame-Options','X-Powered-By', 'Strict-Transport-Security',
                           'X-Content-Type-Options', 'X-XSS-
Protection', 'X-Permitted-Cross-Domain-Policies', 'Set-Cookie',
'Cache-Control',
                           'Transfer-Encoding', 'Access-Control-
Allow-Methods', 'Access-Control-Allow-Origin', 'Content-Security-
Policy', 'Referrer-Policy']
        for header in response_security_headers:
            try:
                result = url_request.headers[header]
                print('%s: %s' % (header, result))
            except KeyError:
                print (header + ': ' + Fore.RED + Style.BRIGHT + 'No
encontrado')

        print(Fore.GREEN + Style.DIM + "\n\n====== Hecho ====== \n")
    else:
        print("No es una url válida")
except Exception as e:
    print(Fore.MAGENTA + Style.BRIGHT + f"¡Algo salió mal! {e}")

# --- EJERCICIO INTERACTIVO ---
# 1. Ejecuta el código en tu IDE local.
```

Escáner de Puertos (Socket)

Estudia el siguiente código: Escáner de Puertos (Socket)

Código de Ejemplo

```
#!/usr/bin/python
# --- IMPORTANTE: EJECUCIÓN LOCAL REQUERIDA ---
# Este ejercicio requiere librerías externas o generar archivos.
# Por favor, ejecútalo en tu IDE local (VS Code, PyCharm, etc).
# ------------------------------------------------------
# Escáner de Puertos Simple usando Sockets
# Escanea puertos comunes en localhost.
```

```
import socket
import sys

target = "127.0.0.1" # Localhost
ports = [21, 22, 80, 443, 8000, 8080]

print(f"--- Escaneando {target} ---")

for port in ports:
    s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
    s.settimeout(0.5)

    # Intentar conectar
    result = s.connect_ex((target, port))

    if result == 0:
        print(f"Puerto {port}: ABIERTO")
    else:
        print(f"Puerto {port}: CERRADO")

    s.close()

print("\nEscaneo completado.")

# --- EJERCICIO INTERACTIVO ---
# 1. Ejecuta el código en tu IDE local.
```

Módulo 06: Proyectos Finales

Proyecto 1: Web Scraper (Petición)

Estudia el siguiente código: Proyecto 1: Web Scraper (Petición)

Código de Ejemplo

```
#!/usr/bin/python
# --- IMPORTANTE: EJECUCIÓN LOCAL REQUERIDA ---
# Este ejercicio requiere librerías externas o generar archivos.
# Por favor, ejecútalo en tu IDE local (VS Code, PyCharm, etc).
# ------------------------------------------------------
# Proyecto Final 1: Web Scraper de Libros
# Parte 1: Realizar la petición HTTP

import requests

# Sitio de pruebas para scraping
URL = "http://books.toscrape.com/"

print(f"Conectando a {URL}...")

try:
    response = requests.get(URL)

    if response.status_code == 200:
        print("¡Conexión exitosa!")
        print(f"Status Code: {response.status_code}")
        print(f"Encoding: {response.encoding}")

        # Mostrar los primeros 500 caracteres del HTML
        html_content = response.text
        print("\n--- Contenido HTML (Fragmento) ---")
        print(html_content[:500])
        print("...")
```

```
        # Guardamos el HTML para la siguiente parte
        with open("books.html", "w", encoding="utf-8") as f:
            f.write(html_content)
        print("\nHTML guardado en 'books.html'")

    else:
        print(f"Error: Status Code {response.status_code}")

except Exception as e:
    print(f"Error de conexión: {e}")

# --- EJERCICIO INTERACTIVO ---
# 1. Ejecuta el código en tu IDE local.
# 2. Verifica que se haya creado el archivo 'books.html'.
```

Proyecto 1: Web Scraper (Extracción)

Estudia el siguiente código: Proyecto 1: Web Scraper (Extracción)

Código de Ejemplo

```
#!/usr/bin/python
# --- IMPORTANTE: EJECUCIÓN LOCAL REQUERIDA ---
# Este ejercicio requiere librerías externas o generar archivos.
# Por favor, ejecútalo en tu IDE local (VS Code, PyCharm, etc).
# ------------------------------------------------------
# Proyecto Final 1: Web Scraper de Libros
# Parte 2: Extraer datos con BeautifulSoup

from bs4 import BeautifulSoup
import os
import csv

filename = "books.html"

if not os.path.exists(filename):
    print(f"Error: {filename} no existe. Ejecuta la lección
```

```
anterior.")
else:
    print(f"Analizando {filename}...")
    with open(filename, "r", encoding="utf-8") as f:
        html_content = f.read()

    soup = BeautifulSoup(html_content, "html.parser")

    # Encontrar todos los artículos de libros
    articles = soup.find_all("article", class_="product_pod")
    print(f"Se encontraron {len(articles)} libros.\n")

    books_data = []

    for article in articles:
        # Título (está en el atributo title del enlace h3 -> a)
        h3 = article.find("h3")
        link = h3.find("a")
        title = link["title"]

        # Precio (está en div class="product_price" -> p
class="price_color")
        price_tag = article.find("p", class_="price_color")
        price = price_tag.text

        # Disponibilidad
        stock_tag = article.find("p", class_="instock availability")
        stock = stock_tag.text.strip()

        print(f"Libro: {title[:30]}... | Precio: {price}")
        books_data.append([title, price, stock])

    # Guardar a CSV
    csv_file = "libros_extraidos.csv"
    with open(csv_file, "w", newline="", encoding="utf-8") as f:
        writer = csv.writer(f)
        writer.writerow(["Titulo", "Precio", "Stock"])
        writer.writerows(books_data)

    print(f"\nDatos exportados a {csv_file}")

# --- EJERCICIO INTERACTIVO ---
```

```
# 1. Ejecuta el código en tu IDE local.
# 2. Abre el archivo CSV generado (o léelo con Python) para ver los
resultados.
```

Proyecto 2: Analizador de Logs (Lectura)

Estudia el siguiente código: Proyecto 2: Analizador de Logs (Lectura)

Código de Ejemplo

```
#!/usr/bin/python
# --- IMPORTANTE: EJECUCIÓN LOCAL REQUERIDA ---
# Este ejercicio requiere librerías externas o generar archivos.
# Por favor, ejecútalo en tu IDE local (VS Code, PyCharm, etc).
# ------------------------------------------------
# Proyecto Final 2: Analizador de Logs de Servidor
# Parte 1: Generar y Leer Logs

# Simulamos un archivo de logs de Apache/Nginx
log_data = """
192.168.1.10 - - [23/Nov/2025:10:00:01 +0000] "GET /index.html HTTP/
1.1" 200 1024
192.168.1.11 - - [23/Nov/2025:10:00:05 +0000] "GET /about.html HTTP/
1.1" 200 512
192.168.1.12 - - [23/Nov/2025:10:00:10 +0000] "GET /contact.php HTTP/
1.1" 404 150
192.168.1.10 - - [23/Nov/2025:10:00:15 +0000] "POST /login HTTP/1.1"
200 0
10.0.0.5 - - [23/Nov/2025:10:01:00 +0000] "GET /admin HTTP/1.1" 403
200
192.168.1.11 - - [23/Nov/2025:10:01:05 +0000] "GET /style.css HTTP/
1.1" 200 3000
"""

filename = "server.log"
print(f"Creando archivo de logs simulado: {filename}")
with open(filename, "w") as f:
```

```
    f.write(log_data.strip())

print("Leyendo archivo línea por línea...")
with open(filename, "r") as f:
    for i, line in enumerate(f, 1):
        print(f"Línea {i}: {line.strip()}")

# --- EJERCICIO INTERACTIVO ---
# 1. Ejecuta el código en tu IDE local.
# 2. Añade manualmente una línea más al archivo log_data con un error
500.
```

Proyecto 2: Analizador de Logs (Reporte)

Estudia el siguiente código: Proyecto 2: Analizador de Logs (Reporte)

Código de Ejemplo

```
#!/usr/bin/python
# --- IMPORTANTE: EJECUCIÓN LOCAL REQUERIDA ---
# Este ejercicio requiere librerías externas o generar archivos.
# Por favor, ejecútalo en tu IDE local (VS Code, PyCharm, etc).
# ------------------------------------------------------
# Proyecto Final 2: Analizador de Logs de Servidor
# Parte 2: Generar Reporte de Estadísticas

import re
from collections import Counter

filename = "server.log"

print("Analizando logs...")

ip_pattern = r"(\d{1,3}\.\d{1,3}\.\d{1,3}\.\d{1,3})"
status_pattern = r"\" (\d{3}) "

ips = []
```

```
status_codes = []

try:
    with open(filename, "r") as f:
        for line in f:
            # Extraer IP
            ip_match = re.search(ip_pattern, line)
            if ip_match:
                ips.append(ip_match.group(1))

            # Extraer Código de Estado
            status_match = re.search(status_pattern, line)
            if status_match:
                status_codes.append(status_match.group(1))

    # Estadísticas
    total_requests = len(ips)
    unique_ips = len(set(ips))
    top_ips = Counter(ips).most_common(3)
    errors_404 = status_codes.count("404")
    errors_403 = status_codes.count("403")

    print(f"\n--- Reporte de Tráfico ---")
    print(f"Total Peticiones: {total_requests}")
    print(f"IPs Únicas: {unique_ips}")
    print(f"Errores 404: {errors_404}")
    print(f"Errores 403: {errors_403}")
    print(f"\nTop 3 IPs Activas:")
    for ip, count in top_ips:
        print(f"  {ip}: {count} peticiones")

    # Guardar reporte
    with open("reporte_logs.txt", "w") as f:
        f.write(f"Reporte generado.\nTotal: {total_requests}\nErrores
404: {errors_404}")
    print("\nReporte guardado en 'reporte_logs.txt'")

except FileNotFoundError:
    print("Error: server.log no encontrado.")

# --- EJERCICIO INTERACTIVO ---
# 1. Ejecuta el código en tu IDE local.
```

Proyecto 3: Organizador de Archivos

Estudia el siguiente código: Proyecto 3: Organizador de Archivos

Código de Ejemplo

```
#!/usr/bin/python
# --- IMPORTANTE: EJECUCIÓN LOCAL REQUERIDA ---
# Este ejercicio requiere librerías externas o generar archivos.
# Por favor, ejecútalo en tu IDE local (VS Code, PyCharm, etc).
# ------------------------------------------------------
# Proyecto Final 3: Organizador Automático de Archivos

import os
import shutil

# 1. Configuración: Crear entorno de prueba
base_dir = "downloads_test"
if not os.path.exists(base_dir):
    os.mkdir(base_dir)

# Crear archivos dummy
files_to_create = ["foto1.jpg", "documento.pdf", "notas.txt",
"cancion.mp3", "foto2.png", "script.py"]
print(f"Creando archivos de prueba en {base_dir}...")
for f in files_to_create:
    with open(os.path.join(base_dir, f), "w") as file:
        file.write("dummy content")

# 2. Lógica de Organización
def organizar_archivos(directorio):
    extensions = {
        "Imagenes": [".jpg", ".png", ".gif"],
        "Documentos": [".pdf", ".txt", ".docx"],
        "Audio": [".mp3", ".wav"],
        "Codigo": [".py", ".js", ".html"]
    }
```

```
    print(f"\nOrganizando {directorio}...")

    for filename in os.listdir(directorio):
        filepath = os.path.join(directorio, filename)

        if os.path.isdir(filepath):
            continue

        _, ext = os.path.splitext(filename)
        moved = False

        for folder, ext_list in extensions.items():
            if ext.lower() in ext_list:
                target_folder = os.path.join(directorio, folder)
                if not os.path.exists(target_folder):
                    os.mkdir(target_folder)

                shutil.move(filepath, os.path.join(target_folder,
filename))
                print(f"Movido: {filename} -> {folder}/")
                moved = True
                break

        if not moved:
            print(f"Omitido: {filename} (Extensión desconocida)")

# 3. Ejecutar
organizar_archivos(base_dir)
print("\n¡Organización completada!")

# --- EJERCICIO INTERACTIVO ---
# 1. Ejecuta el código en tu IDE local.
# 2. Verifica la carpeta 'downloads_test' para ver las subcarpetas
creadas.
```

Módulo 07: Certificación

Obtener Certificado

Proyecto Final: Generador de Certificados

Has llegado al final. Como último ejercicio, utilizarás la librería `fpdf` para generar tu propio certificado.

1. Analiza el código proporcionado.
2. Cambia la variable `nombre_estudiante` con tu nombre real.
3. Ejecuta el código para descargar tu PDF.

Código de Ejemplo

```
from fpdf import FPDF
from fpdf.enums import XPos, YPos
from datetime import datetime

# --- CONFIGURACIÓN ---
# NOTA: Si ejecutas este código en tu ordenador (PyCharm, VSCode),
# necesitas instalar la librería primero ejecutando en tu terminal:
# pip install fpdf2

# ¡Escribe tu nombre aquí para generar tu certificado!
nombre_estudiante = "TU NOMBRE AQUÍ"

def generar_certificado(nombre):
    print(f"Generando certificado para: {nombre}...")

    # Crear PDF (Horizontal, A4)
    pdf = FPDF(orientation='L', unit='mm', format='A4')
    pdf.add_page()

    # Borde
    pdf.set_line_width(1)
```

```
    pdf.rect(5, 5, 287, 200)
    pdf.rect(8, 8, 281, 194)

    # Título
    pdf.set_font('Helvetica', 'B', 30)
    pdf.cell(0, 40, 'CERTIFICADO DE FINALIZACIÓN', new_x=XPos.LMARGIN,
new_y=YPos.NEXT, align='C')
    pdf.ln(10)

    # Cuerpo
    pdf.set_font('Helvetica', '', 16)
    pdf.cell(0, 10, 'Este documento certifica que',
new_x=XPos.LMARGIN, new_y=YPos.NEXT, align='C')
    pdf.ln(10)

    # Nombre del Estudiante
    pdf.set_font('Helvetica', 'B', 40)
    pdf.set_text_color(44, 62, 80) # Azul Oscuro
    pdf.cell(0, 20, nombre, new_x=XPos.LMARGIN, new_y=YPos.NEXT,
align='C')
    pdf.ln(10)

    # Texto
    pdf.set_text_color(0, 0, 0)
    pdf.set_font('Helvetica', '', 16)
    pdf.cell(0, 10, 'ha completado satisfactoriamente el curso',
new_x=XPos.LMARGIN, new_y=YPos.NEXT, align='C')
    pdf.ln(5)

    # Nombre del Curso
    pdf.set_font('Helvetica', 'B', 24)
    pdf.set_text_color(192, 57, 43) # Rojo
    pdf.cell(0, 20, 'Python para Ciberseguridad', new_x=XPos.LMARGIN,
new_y=YPos.NEXT, align='C')
    pdf.ln(20)

    # Fecha
    date_str = datetime.now().strftime("%d/%m/%Y")
    pdf.set_font('Helvetica', 'I', 12)
    pdf.set_text_color(0, 0, 0)
    pdf.cell(0, 10, f"Fecha: {date_str}", new_x=XPos.LMARGIN,
new_y=YPos.NEXT, align='C')
```

```
    pdf.ln(20)

    # Firma
    pdf.set_font('Helvetica', '', 14)
    pdf.cell(0, 10, '_' * 30, new_x=XPos.LMARGIN, new_y=YPos.NEXT,
align='C')
    pdf.cell(0, 10, 'Carlos Dominguez', new_x=XPos.LMARGIN,
new_y=YPos.NEXT, align='C')
    pdf.set_font('Helvetica', 'I', 10)
    pdf.cell(0, 5, 'Instructor & Creador', new_x=XPos.LMARGIN,
new_y=YPos.NEXT, align='C')

    # Guardar archivo
    filename = f"certificado_{nombre.replace(' ', '_')}.pdf"
    pdf.output(filename)
    print(f"¡Éxito! Certificado guardado como: {filename}")
    return filename

if __name__ == "__main__":
    if nombre_estudiante == "TU NOMBRE AQUÍ":
        print("¡Error! Por favor actualiza la variable
'nombre_estudiante' con tu nombre real.")
    else:
        generar_certificado(nombre_estudiante)
```