Actividad: Tipos de Datos y Adquisición de Datos en Python

Parte 2: Adquisición y Almacenamiento de Datos

Objetivos:

- Aprender cómo obtener datos de fuentes externas (APIs, web scraping).
- Aprender a almacenar los datos adquiridos en archivos adecuados.

Instrucciones Paso a Paso

1. Investigación Teórica (1 hora)

Objetivo: Entender cómo funcionan las APIs, el web scraping y la conexión con bases de datos.

1. **Lee sobre las siguientes técnicas de adquisición de datos**:
    - **APIs**: Son interfaces que permiten interactuar con servicios externos para obtener datos. Por ejemplo, puedes obtener datos de Twitter, OpenWeather o cualquier otra fuente pública.
    - **Web Scraping**: Consiste en extraer datos de sitios web automáticamente. Se usan herramientas como **BeautifulSoup** o **Scrapy**.
    - **Bases de Datos**: Son sistemas de almacenamiento de datos organizados. Puedes acceder a ellas desde Python usando bibliotecas como **sqlite3** o **SQLAlchemy**.
2. **Responde las siguientes preguntas** en tu informe:
    - ¿Qué es una API y cómo puedes interactuar con ellas para obtener datos?
    - ¿Cuál es la diferencia entre el web scraping y el uso de APIs?
    - ¿Cómo puedes almacenar los datos adquiridos de una API o scraping en un archivo CSV?

2. Actividad Práctica

Objetivo: Obtener datos de una API pública y almacenarlos en un archivo CSV.

1. Trabajo con una API (Ejemplo con OpenWeather):

    - **Paso 1**: Regístrate en [OpenWeatherMap] y obtén una clave API.

    - **Paso 2**: Usa la biblioteca `requests` en Python para obtener datos del clima. El siguiente código te ayudará a hacer una solicitud a la API:

```
import requests
import pandas as pd
import matplotlib.pyplot as plt
import json

api_key = 'tu api key'
ciudad = "La Laguna"
url = f'http://api.openweathermap.org/data/2.5/weather?q={ciudad}&appid={api_key}'
response = requests.get(url)
data = response.json()

# Muestra los datos obtenidos
print(data)
```

    - **Paso 3**: Almacena los datos en un archivo CSV. Por ejemplo:

```
# Supón que 'data' tiene la información del clima
df = pd.DataFrame([data])
df.to_csv('clima_london.csv', index=False)
```

```
### 1️⃣ Formatear los datos y mostrarlos en consola
print(f"""
📍 Ubicación: {data['name']}, {data['sys']['country']}
🌍 Coordenadas: Latitud {data['coord']['lat']}, Longitud {data['coord']['lon']}
🌤️ Clima: {data['weather'][0]['main']} ({data['weather'][0]['description']})
🌡️ Temperatura: {data['main']['temp']} K (≈ {round(data['main']['temp'] - 273.15,
2)} °C)
🤔 Sensación térmica: {data['main']['feels_like']} K (≈
{round(data['main']['feels_like'] - 273.15, 2)} °C)
```

```
🔽 Mínima: {data['main']['temp_min']} K (≈ {round(data['main']['temp_min'] -
273.15, 2)} °C)
🔼 Máxima: {data['main']['temp_max']} K (≈ {round(data['main']['temp_max'] -
273.15, 2)} °C)
💨 Viento: {data['wind']['speed']} m/s, Dirección: {data['wind']['deg']}°
💧 Humedad: {data['main']['humidity']}%
📏 Presión: {data['main']['pressure']} hPa
☁️ Nubosidad: {data['clouds']['all']}%
👁️ Visibilidad: {data['visibility']} m
""")

### 2️⃣ Guardar los datos en un archivo CSV con Pandas
df = pd.DataFrame([{
    "Ubicación": data["name"],
    "País": data["sys"]["country"],
    "Latitud": data["coord"]["lat"],
    "Longitud": data["coord"]["lon"],
    "Clima": data["weather"][0]["description"],
    "Temperatura (K)": data["main"]["temp"],
    "Temperatura (°C)": round(data["main"]["temp"] - 273.15, 2),
    "Sensación térmica (°C)": round(data["main"]["feels_like"] - 273.15, 2),
    "Temp Mín (°C)": round(data["main"]["temp_min"] - 273.15, 2),
    "Temp Máx (°C)": round(data["main"]["temp_max"] - 273.15, 2),
    "Humedad (%)": data["main"]["humidity"],
    "Presión (hPa)": data["main"]["pressure"],
    "Viento (m/s)": data["wind"]["speed"],
    "Dirección Viento (°)": data["wind"]["deg"],
    "Nubosidad (%)": data["clouds"]["all"],
    "Visibilidad (m)": data["visibility"]
}])

df.to_csv("clima_{ciudad}.csv", index=False)
print(f"Datos guardados en clima_{ciudad}.csv")

### 3️⃣ Crear gráficos para visualizar los datos

# 📊 Gráfico de temperaturas
plt.figure(figsize=(8, 5))
temps = ["Temp Actual", "Sensación", "Mínima", "Máxima"]
values = [
    round(data["main"]["temp"] - 273.15, 2),
    round(data["main"]["feels_like"] - 273.15, 2),
    round(data["main"]["temp_min"] - 273.15, 2),
    round(data["main"]["temp_max"] - 273.15, 2),
]
plt.bar(temps, values, color=["blue", "cyan", "green", "red"])
plt.ylabel("Temperatura (°C)")
```

```
plt.title("Temperaturas en Londres")
plt.show()

# 📊 Gráfico de otras variables (Humedad, Presión, Viento)
plt.figure(figsize=(8, 5))
labels = ["Humedad (%)", "Presión (hPa)", "Viento (m/s)", "Nubosidad (%)"]
values = [data["main"]["humidity"], data["main"]["pressure"],
data["wind"]["speed"], data["clouds"]["all"]]

plt.barh(labels, values, color=["purple", "orange", "blue", "gray"])
plt.xlabel("Valores")
plt.title("Otras Condiciones Climáticas")
plt.show()
```

2. Trabajo con Web Scraping (Ejemplo con BeautifulSoup):
    - Paso 1: Instala BeautifulSoup:

```
pip install beautifulsoup4 requests
```

    - Paso 2: Realiza scraping de un sitio web (por ejemplo, extraer los títulos de noticias de un sitio):

```
from bs4 import BeautifulSoup  # Importamos BeautifulSoup para analizar el contenido HTML
import requests  # Importamos la librería requests para hacer peticiones HTTP

# Definimos la URL del sitio web que queremos scrape
url = 'https://www.bbc.com'

# Realizamos la solicitud HTTP al sitio web y obtenemos la respuesta
response = requests.get(url)

# Usamos BeautifulSoup para analizar el contenido de la respuesta en formato HTML
soup = BeautifulSoup(response.content, 'html.parser')

# Buscamos todas las etiquetas <h2> en el contenido HTML, que deberían contener los titulares
headlines = soup.find_all('h2')
```

```
# Almacenamos los titulares en una lista
titles = []

# Recorremos cada uno de los elementos encontrados
for headline in headlines:
    text = headline.text.strip()
    if text:  # Solo añadimos si no está vacío
        titles.append(text)

# Imprimimos todos los titulares juntos, cada uno en una línea
print("\n".join(titles))
```

- Paso 3: Almacena los datos obtenidos en un archivo CSV:

```
headlines_text = [headline.text for headline in headlines]
df = pd.DataFrame(headlines_text, columns=['Headline'])
df.to_csv('bbc_headlines.csv', index=False)
```

3. Entregable:

- Un informe que incluya:
  - Respuestas a las preguntas teóricas.
  - El código utilizado para obtener y almacenar los datos (de la API o mediante scraping).
  - Los archivos CSV con los datos obtenidos.
