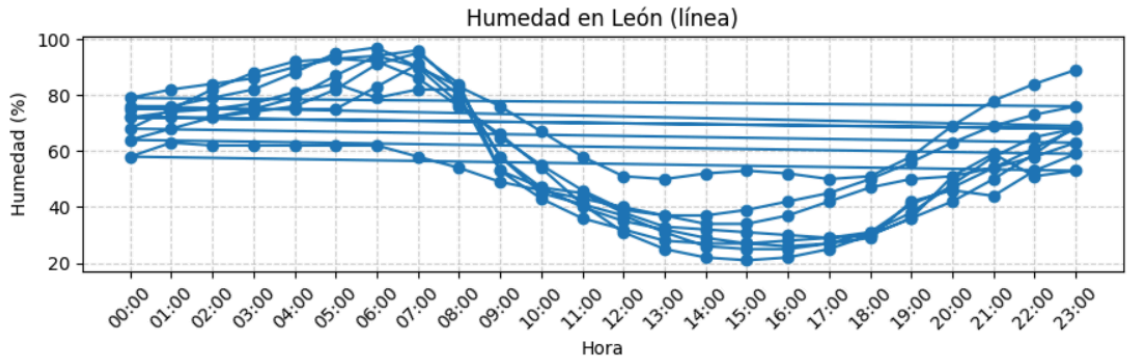
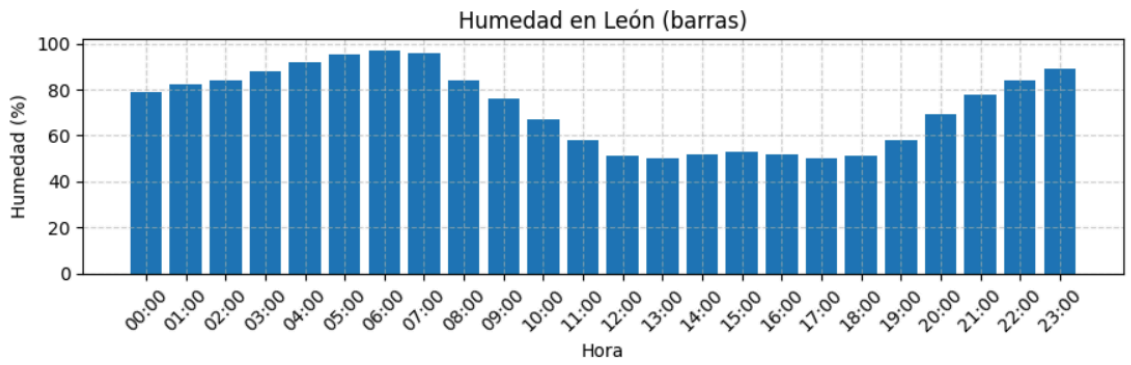
```python
import tkinter as tk
from tkinter import ttk, messagebox
import requests
import matplotlib.pyplot as plt
from matplotlib.backends.backend_tkagg import FigureCanvasTkAgg


def fetch_data():
    """
    Llama a la API de Open-Meteo y obtiene la humedad relativa horaria
    de León, Gto (últimas 24 horas).
    Retorna listas paralelas: horas y humedad.
    """
    try:
        url = (
            "https://api.open-meteo.com/v1/forecast"
            "?latitude=21.12&longitude=-101.68"
            "&hourly=relativehumidity_2m&past_days=1"
            "&timezone=auto"
        )
        response = requests.get(url, timeout=15)
        response.raise_for_status()
        data = response.json()


        horas = [h.split("T")[1] for h in data["hourly"]["time"]]
        humedad = data["hourly"]["relativehumidity_2m"]

        return horas, humedad
    except Exception as e:
        messagebox.showerror("Error", f"No se pudieron obtener los
datos:\n{e}")
        return [], []


def create_line_chart(horas, humedad):
    """Gráfica lineal simple con rejilla (humedad)."""
    fig, ax = plt.subplots(figsize=(6, 3))
    ax.plot(horas, humedad, marker="o")
    ax.set_title("Humedad en León (línea)")
    ax.set_xlabel("Hora")
    ax.set_ylabel("Humedad (%)")
    ax.tick_params(axis="x", rotation=45)
```

```
    ax.grid(True, linestyle="--", alpha=0.6)
    fig.tight_layout()
    return fig


def create_bar_chart(horas, humedad):
    """Gráfica de barras simple con rejilla (humedad)."""
    fig, ax = plt.subplots(figsize=(6, 3))
    ax.bar(horas, humedad)
    ax.set_title("Humedad en León (barras)")
    ax.set_xlabel("Hora")
    ax.set_ylabel("Humedad (%)")
    ax.tick_params(axis="x", rotation=45)
    ax.grid(True, linestyle="--", alpha=0.6)
    fig.tight_layout()
    return fig


def mostrar_graficas(frm, horas, humedad):
    """Inserta las dos gráficas en el frame de la ventana tkinter."""

    for widget in frm.winfo_children():
        widget.destroy()

    # Gráfica de línea
    fig1 = create_line_chart(horas, humedad)
    canvas1 = FigureCanvasTkAgg(fig1, master=frm)
    canvas1.draw()
    canvas1.get_tk_widget().pack(pady=10, fill="x")

    # Gráfica de barras
    fig2 = create_bar_chart(horas, humedad)
    canvas2 = FigureCanvasTkAgg(fig2, master=frm)
    canvas2.draw()
    canvas2.get_tk_widget().pack(pady=10, fill="x")


def open_win_canvas(parent: tk.Tk):
    """Crea la ventana secundaria con las gráficas."""
    win = tk.Toplevel(parent)
    win.title("Canvas con API (Open-Meteo) - Humedad")
    win.geometry("960x800")
```

```
    frm = ttk.Frame(win, padding=12)
    frm.pack(fill="both", expand=True)

    def cargar():
        horas, humedad = fetch_data()
        if horas and humedad:
            mostrar_graficas(frm, horas, humedad)

    ttk.Button(frm, text="Cargar y mostrar gráficas",
command=cargar).pack(pady=10)


if __name__ == "__main__":
    root = tk.Tk()
    root.title("Prueba win_canvas")
    ttk.Button(root, text="Abrir ventana Canvas",
               command=lambda: open_win_canvas(root)).pack(pady=20)
    root.mainloop()
```





Este programa abre una ventana y muestra datos de humedad tomados de una página del clima (API de Open-Meteo). Presenta dos gráficas: una de línea para ver cómo sube o baja la humedad con las horas y otra de barras para comparar mejor los valores. Se agregaron rejillas y nombres en los ejes para que se entienda más fácil. Se le podrían poner mejoras como mostrar el promedio de humedad, limitar los valores de 0 a 100 o dejar que el usuario elija otra variable del clima.

Este programa usa Tkinter para la ventana, requests para conectarse a la página del clima (Open-Meteo) y traer los datos, y matplotlib para dibujar las gráficas. Con eso se muestran dos gráficas de humedad: una de línea y otra de barras, ambas con rejillas y etiquetas para que sea fácil de entender. Se podrían añadir mejoras como calcular el promedio, ajustar los valores de 0 a 100 o dejar elegir qué dato del clima mostrar.