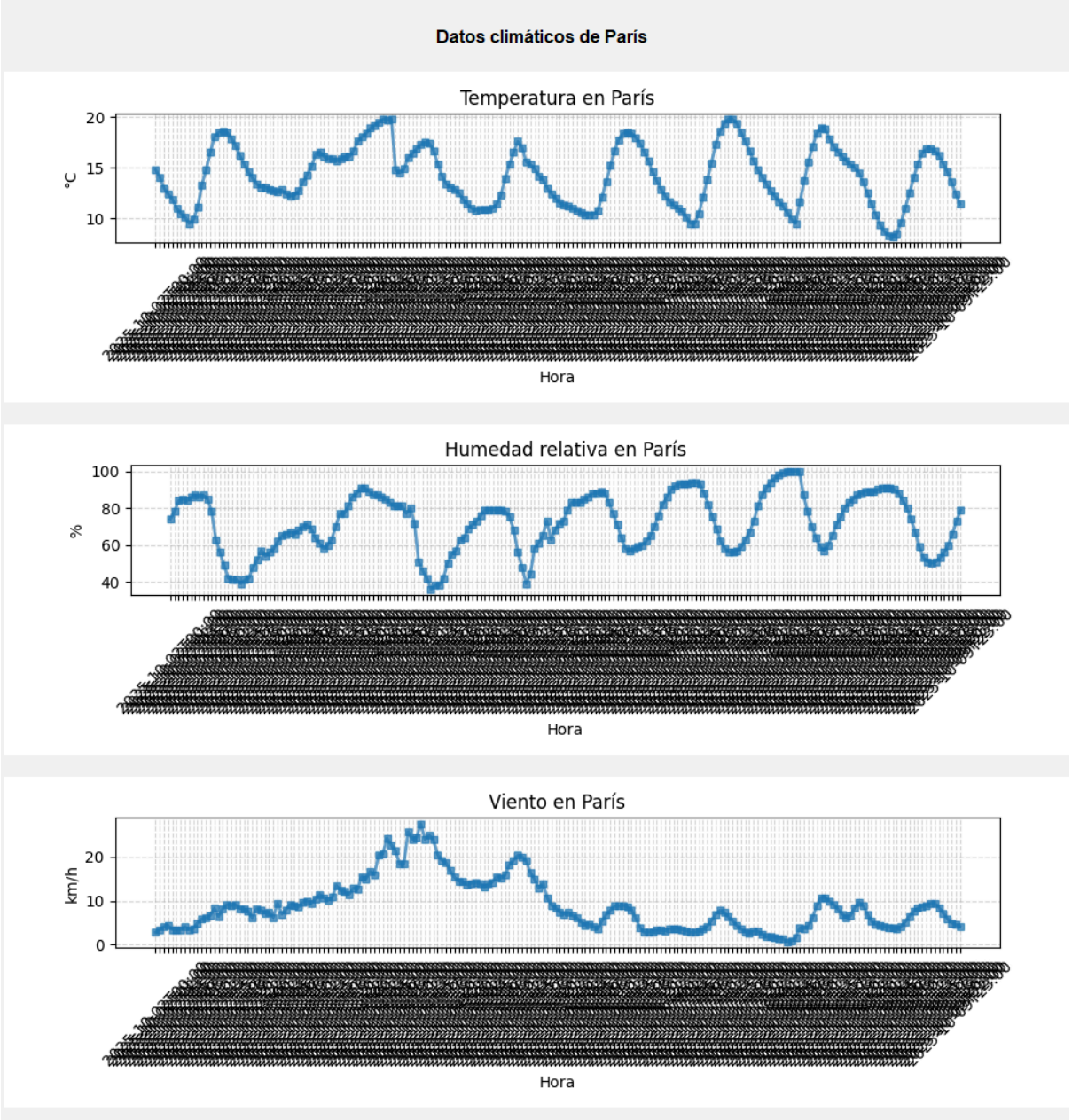
Sesión 7
Marcelo Vázquez González
A00574942



Codigo:

```
import tkinter as tk
from tkinter import ttk, messagebox
```

```
import requests
import matplotlib.pyplot as plt
from matplotlib.backends.backend_tkagg import FigureCanvasTkAgg


def get_coordinates(city_name: str):
    """
    Usa la API de Open-Meteo Geocoding para obtener lat/lon de la
ciudad.
    """
    try:
        url =
f"https://geocoding-api.open-meteo.com/v1/search?name={city_name}&count
=1&language=es&format=json"
        response = requests.get(url, timeout=15)
        response.raise_for_status()
        data = response.json()

        if "results" not in data or not data["results"]:
            raise ValueError("No se encontraron coordenadas para la
ciudad")

        lat = data["results"][0]["latitude"]
        lon = data["results"][0]["longitude"]
        return lat, lon
    except Exception as e:
        messagebox.showerror("Error", f"No se pudieron obtener
coordenadas:\n{e}")
        return None, None


def fetch_data(city: str):
    """
    Obtiene temperaturas, humedad y viento de la ciudad dada.
    Devuelve: horas, temperaturas, humedades, vientos.
    """
    try:
        lat, lon = get_coordinates(city)
        if lat is None or lon is None:
            return [], [], [], []

        url = (
            f"https://api.open-meteo.com/v1/forecast"
```

```
            f"?latitude={lat}&longitude={lon}"
            "&hourly=temperature_2m,relativehumidity_2m,windspeed_10m"
            "&past_days=1&timezone=auto"
        )
        response = requests.get(url, timeout=15)
        response.raise_for_status()
        data = response.json()

        horas = data["hourly"]["time"]
        temps = data["hourly"]["temperature_2m"]
        hums = data["hourly"]["relativehumidity_2m"]
        winds = data["hourly"]["windspeed_10m"]

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


def create_line_chart(horas, valores, titulo, ylabel):
    """Gráfica de línea genérica."""
    fig, ax = plt.subplots(figsize=(6, 3))
    ax.plot(horas, valores, linestyle="-", marker="s", markersize=4,
            linewidth=2, alpha=0.7)
    ax.set_title(titulo)
    ax.set_xlabel("Hora")
    ax.set_ylabel(ylabel)
    ax.tick_params(axis="x", rotation=45)
    ax.grid(True, linestyle="--", alpha=0.5)
    fig.tight_layout()
    return fig


def mostrar_graficas(frm, horas, temps, hums, winds, ciudad):
    """Inserta las tres gráficas en el frame de la ventana tkinter."""
    for widget in frm.winfo_children():
        widget.destroy()  # limpia gráficas anteriores

    ttk.Label(frm, text=f"Datos climáticos de {ciudad}", font=("Arial",
12, "bold")).pack(pady=10)

    # Temperatura
```

```
    fig1 = create_line_chart(horas, temps, f"Temperatura en {ciudad}",
"°C")
    canvas1 = FigureCanvasTkAgg(fig1, master=frm)
    canvas1.draw()
    canvas1.get_tk_widget().pack(pady=10, fill="x")

    # Humedad
    fig2 = create_line_chart(horas, hums, f"Humedad relativa en
{ciudad}", "%")
    canvas2 = FigureCanvasTkAgg(fig2, master=frm)
    canvas2.draw()
    canvas2.get_tk_widget().pack(pady=10, fill="x")

    # Viento
    fig3 = create_line_chart(horas, winds, f"Viento en {ciudad}",
"km/h")
    canvas3 = FigureCanvasTkAgg(fig3, master=frm)
    canvas3.draw()
    canvas3.get_tk_widget().pack(pady=10, fill="x")


def open_win_canvas(parent: tk.Tk):
    """
    Crea la ventana secundaria con gráficas de la API.
    """
    win = tk.Toplevel(parent)
    win.title("Canvas con API (Open-Meteo) y gráficas")
    win.geometry("1000x1200")

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

    # Entrada de ciudad
    ciudad_var = tk.StringVar(value="León")
    entry_ciudad = ttk.Entry(frm, textvariable=ciudad_var, width=30)
    entry_ciudad.pack(pady=10)

    # Botón para cargar datos y graficar
    def cargar():
        ciudad = ciudad_var.get().strip()
        horas, temps, hums, winds = fetch_data(ciudad)
        if horas:
            mostrar_graficas(frm, horas, temps, hums, winds, ciudad)
```

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


# Para pruebas independientes
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()
```

Cambio realizados:

Empecé por poner los cambios sugeridos en la actividad, y luego agregue el apartado para elegir la ciudad de la que se muestra la información y se muestra una gráfica de temperatura, una de humedad relativa en la ciudad elegida y el viento de la misma.