```python
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt

# Bước 1: Đọc dữ liệu từ tệp CSV

pf = pd.read_csv('/content/drive/MyDrive/Colab Notebooks/TQHDL/1_data_test.csv')
df = pd.DataFrame(pf)

# Bước 2: Vẽ biểu đồ bằng Seaborn
# Thiết lập kích thước cho biểu đồ để dễ nhìn hơn
plt.figure(figsize=(8, 6))

# Sử dụng sns.histplot để vẽ biểu đồ phân phối
# - data=df: Nguồn dữ liệu là DataFrame đã đọc
# - x="total_bill": Dữ liệu trên trục hoành là cột 'total_bill'
# - hue="sex": Phân tách và tô màu các cột dựa trên giá trị trong cột 'sex'
# - kde=True: Vẽ thêm đường ước tính mật độ (đường cong mượt) cho mỗi nhóm
sns.histplot(data=df, x="total_bill", hue="sex", kde=True)

# Bước 3: Đặt tiêu đề và hiển thị biểu đồ
plt.title("Phân phối giá trị hóa đơn theo giới tính")
plt.xlabel("Tổng giá trị hóa đơn (total_bill)")
plt.ylabel("Số lượng (Count)")

# Hiển thị biểu đồ
plt.show()
```

Cach 02:

```
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
from scipy.stats import gaussian_kde

# Bước 1: Đọc dữ liệu từ tệp CSV
pf = pd.read_csv('/content/drive/MyDrive/Colab Notebooks/TQHDL/1_data_test.csv')
df = pd.DataFrame(pf)

# Tách dữ liệu theo giới tính
male_data = df[df['sex'] == 'Male']['total_bill']
female_data = df[df['sex'] == 'Female']['total_bill']

# Bước 2: Thiết lập và vẽ biểu đồ
plt.figure(figsize=(8, 6))

# Tự động tạo các bins (khoảng chia) cho biểu đồ histogram
bins = np.linspace(df['total_bill'].min(), df['total_bill'].max(), 15)
bin_width = bins[1] - bins[0]

# Vẽ biểu đồ histogram cho giới tính Male
plt.hist(male_data, bins=bins, color='sandybrown', edgecolor='black', alpha=0.7, label='Male')

# Vẽ biểu đồ histogram cho giới tính Female
plt.hist(female_data, bins=bins, color='steelblue', edgecolor='black', alpha=0.7,
label='Female')
```

```
# Bước 3: Vẽ đường cong KDE (Kernel Density Estimate)

# Tạo một phạm vi giá trị x để đánh giá KDE
x_eval = np.linspace(df['total_bill'].min(), df['total_bill'].max(), 200)

# Ước tính KDE cho dữ liệu Male
kde_male = gaussian_kde(male_data)
kde_male_y = kde_male(x_eval)
# Nhân giá trị KDE với số lượng dữ liệu và độ rộng bin để khớp với trục Y (Count)
plt.plot(x_eval, kde_male_y * len(male_data) * bin_width, color='sandybrown')

# Ước tính KDE cho dữ liệu Female
kde_female = gaussian_kde(female_data)
kde_female_y = kde_female(x_eval)
# Nhân giá trị KDE với số lượng dữ liệu và độ rộng bin để khớp với trục Y (Count)
plt.plot(x_eval, kde_female_y * len(female_data) * bin_width, color='steelblue')

# Bước 4: Thêm các chi tiết cho biểu đồ
plt.title("Phân phối giá trị hóa đơn theo giới tính")
plt.xlabel("Tổng giá trị hóa đơn (total_bill)")
plt.ylabel("Số lượng (Count)")
plt.legend(title='sex')
plt.grid(False) # Tắt lưới
plt.show()
```

Bai 02:

```python
import pandas as pd
import matplotlib.pyplot as plt

# doc file csv
df = pd.read_csv('/content/drive/MyDrive/Colab Notebooks/TQHDL/2_ty_le_that_nghiep.csv')

# Configure Matplotlib to use a generic font family
plt.rcParams['font.family'] = 'sans-serif'

# Bước 2: Vẽ biểu đồ
plt.figure(figsize=(8, 6)) # Thiết lập kích thước cho biểu đồ (tùy chọn)

# Vẽ biểu đồ đường
plt.plot(df['Năm'], df['Tỷ lệ thất nghiệp'],
         color='green',
         marker='o',
         label='Tỷ lệ thất nghiệp')

# Bước 3: Thêm các chi tiết cho biểu đồ
plt.title('Tỷ lệ thất nghiệp so với năm', fontsize=16)
plt.xlabel('Năm', fontsize=12)
plt.ylabel('Tỷ lệ thất nghiệp', fontsize=12)
plt.grid(True)
plt.legend()

# Bước 4: Hiển thị biểu đồ
plt.show()
```

Bai 3

```python
# Dữ liệu
labels = ['Nước', 'Điện', 'Xăng', 'Dầu']
sizes = [15, 30, 45, 10]
colors = ['#1f77b4', '#ff7f0e', '#2ca02c', '#d62728']

# Nhấn mạnh phần "Dầu" nhiều hơn các phần còn lại
explode = (0.05, 0.05, 0.05, 0.25)  # Dầu được "nhích" nhiều hơn

# Vẽ biểu đồ
plt.figure(figsize=(6, 6))
plt.pie(
    sizes,
    explode=explode,
    labels=labels,
    colors=colors,
    autopct='%1.1f%%',
    shadow=False,
    startangle=90,
    textprops={'fontsize': 12}
)

# Cân bằng trục để hình tròn không bị méo
plt.axis('equal')

# Thêm tiêu đề giống mẫu
plt.title('BIỂU ĐỒ TỔNG QUÁT HÓA', fontsize=16, fontweight='bold')

# Hiển thị
plt.show()
```

Tôi có thể giúp bạn tạo gì?

Bai 04:

```python
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

# --- PHẦN THIẾT LẬP BAN ĐẦU ---

# Tên file CSV cần đọc
file_path = '/content/drive/MyDrive/Colab Notebooks/TQHDL/4_air_quality_sensor_data.csv'

df = pd.read_csv(file_path)
print(f"Đã đọc thành công file '{file_path}'.")

# --- Tiền xử lý dữ liệu ---
# Chuyển đổi cột 'Timestamp' sang định dạng datetime để xử lý chuỗi thời gian
df['Timestamp'] = pd.to_datetime(df['Timestamp'])

# Đặt 'Timestamp' làm chỉ số (index) để dễ dàng thao tác
df.set_index('Timestamp', inplace=True)

# --- Cấu hình font để hiển thị tiếng Việt trên biểu đồ ---
# Điều này quan trọng để các nhãn và tiêu đề không bị lỗi font
plt.rcParams['font.family'] = 'sans-serif'
plt.rcParams['font.sans-serif'] = ['Segoe UI', 'DejaVu Sans', 'Arial'] # Sử dụng font có sẵn trên máy bạn
plt.rcParams['axes.unicode_minus'] = False # Để hiển thị dấu trừ đúng cách
```

```
print("Dữ liệu đã được nạp và tiền xử lý xong. Bắt đầu vẽ biểu đồ...")

#=========================================================================
====
# --- CÂU 4.1: Trực quan hóa mối quan hệ giữa nhiệt độ, độ ẩm và PM2.5 ---
#=========================================================================
====
print("\n--- Đang thực hiện Câu 4.1 ---")

plt.figure(figsize=(12, 8))

# Sử dụng biểu đồ phân tán (scatter plot) để thể hiện 3 biến
# Trục X: Nhiệt độ, Trục Y: PM2.5, Màu sắc: Độ ẩm
scatter_plot = sns.scatterplot(
   data=df,
   x='Temperature',
   y='PM2.5',
   hue='Humidity',
   palette='viridis_r', # Bảng màu 'viridis' đảo ngược (màu tối cho giá trị thấp)
   alpha=0.7,
   s=50 # Kích thước các điểm
)

# Đặt tiêu đề và nhãn cho các trục
plt.title('Mối quan hệ giữa Nhiệt độ, Độ ẩm và Bụi mịn PM2.5', fontsize=16, pad=20)
plt.xlabel('Nhiệt độ (°C)', fontsize=12)
plt.ylabel('Nồng độ PM2.5 (μg/m^3)', fontsize=12)
plt.grid(True, linestyle='--', alpha=0.6)

# Tinh chỉnh chú giải (legend)
legend = scatter_plot.get_legend()
legend.set_title('Độ ẩm (%)')

# Hiển thị biểu đồ của câu 4.1
plt.show()

#=========================================================================
====
# --- CÂU 4.2: Phát hiện và đánh dấu những thời điểm bất thường (anomalies) ---
#=========================================================================
====
print("\n--- Đang thực hiện Câu 4.2 ---")

# Sử dụng phương pháp IQR (Interquartile Range) để xác định điểm bất thường
# Tính Q1 (tứ phân vị thứ nhất) và Q3 (tứ phân vị thứ ba)
Q1 = df['PM2.5'].quantile(0.25)
Q3 = df['PM2.5'].quantile(0.75)
IQR = Q3 - Q1

# Ngưỡng trên được định nghĩa là Q3 + 1.5 * IQR
upper_bound = Q3 + 1.5 * IQR
```

```
print(f"Ngưỡng trên để xác định điểm bất thường (Q3 + 1.5 * IQR): {upper_bound:.2f} µg/m^3")

# Lọc ra những dòng dữ liệu có PM2.5 vượt ngưỡng
anomalies = df[df['PM2.5'] > upper_bound]
print(f"Tìm thấy {len(anomalies)} thời điểm bất thường.")

# Trực quan hóa chuỗi thời gian PM2.5 và đánh dấu các điểm bất thường
plt.figure(figsize=(18, 8))
plt.plot(df.index, df['PM2.5'], label='Nồng độ PM2.5 thông thường', color='dodgerblue',
zorder=1)
plt.scatter(anomalies.index, anomalies['PM2.5'], color='red', s=50, label=f'Điểm bất thường (>
{upper_bound:.2f} µg/m^3)', zorder=2)

# Vẽ đường kẻ ngang tại ngưỡng bất thường
plt.axhline(y=upper_bound, color='red', linestyle='--', label='Ngưỡng bất thường')

# Đặt tiêu đề và nhãn
plt.title('Phát hiện các thời điểm bất thường trong dữ liệu PM2.5', fontsize=16, pad=20)
plt.xlabel('Thời gian', fontsize=12)
plt.ylabel('Nồng độ PM2.5 (µg/m^3)', fontsize=12)
plt.legend()
plt.grid(True, linestyle='--', alpha=0.6)
plt.tight_layout()

# Hiển thị biểu đồ của câu 4.2
plt.show()

#=========================================================================
====
# --- CÂU 4.3: Trình bày trực quan để thuyết phục ban quản lý ---
#=========================================================================
====
print("\n--- Đang thực hiện Câu 4.3 ---")

# Tính toán ma trận tương quan giữa tất cả các cột dữ liệu số
correlation_matrix = df.corr()

# Vẽ ma trận tương quan dưới dạng bản đồ nhiệt (heatmap)
plt.figure(figsize=(10, 8))
sns.heatmap(
  correlation_matrix,
  annot=True,       # Hiển thị giá trị số trong mỗi ô
  cmap='coolwarm',    # Bảng màu: Đỏ (tương quan dương), Xanh (tương quan âm)
  fmt='.2f',        # Định dạng số với 2 chữ số thập phân
  linewidths=.5
)

# Đặt tiêu đề
plt.title('Ma trận tương quan giữa các yếu tố thời tiết và ô nhiễm', fontsize=16, pad=20)

# Hiển thị biểu đồ của câu 4.3
plt.show()
```