# Question 1. Analyzing Sales Data: Load sales data from a CSV file into a Pandas DataFrame, calculate total revenue for each product, find the product with highest total revenue, and export results to a new CSV file.

import the library and load the datasets.

"""

import pandas as pd

df = pd.read\_csv("/content/sales\_data.csv")

"""# calculate total revenue for rach product"""

total\_revenue\_per\_product = df.groupby('ProductName')['TotalRevenue'].sum().reset\_index()

total\_revenue\_per\_product

"""# find the product with highest total revenue"""

highest\_revenue\_product = total\_revenue\_per\_product.loc[total\_revenue\_per\_product['TotalRevenue'].idxmax()]

highest\_revenue\_product

"""# export results to a new CSV file."""

results\_file\_path = '/content/total\_revenue\_per\_product.csv'

results\_file\_path

"""/content/sales\_data.csv"""

total\_revenue\_per\_product.to\_csv(results\_file\_path, index=False)

print(f"Total revenue per product saved to {results\_file\_path}")

"""# prints the highest revenue products."""

print(highest\_revenue\_product)

"""# Question 2. Weather Data Analysis: Load weather data stored in a NumPy array, calculate mean temperature for each city across all years, find city with highest average temperature, create new array with only that city's data, and convert temperatures from Celsius to Fahrenheit.

"""

import pandas as pd

import numpy as np

df\_weather = pd.read\_csv("/content/weather\_data\_dataset.csv")

df\_weather

"""# convert dataframe to numPy array."""

df\_np = df\_weather.to\_numpy()

df\_np

# Extract relevant columns for calculations

cities = np.unique(df\_np[:, 0])

temperatures = df\_np[:, 5]

# initialize arrays to store mean temperatures

mean\_temperatures = []

# Calculate mean temperature for each city

for city in cities:

city\_temps = temperatures[df\_np[:, 0] == city].astype(float)

mean\_temp = np.mean(city\_temps)

mean\_temperatures.append((city, mean\_temp))

# Print mean temperatures for verification

for city, mean\_temp in mean\_temperatures:

print(f"Mean temperature in {city}: {mean\_temp:.2f} Celsius")

"""find city with highest average temperature"""

# Find city with highest average temperature

highest\_avg\_temp\_city = max(mean\_temperatures, key=lambda x: x[1])

highest\_avg\_temp\_city

"""create new array with only that city's data"""

city\_data\_np = df\_np[df\_np[:, 0] == highest\_avg\_temp\_city[0]]

city\_data\_np

"""convert temperatures from Celsius to Fahrenheit.

F= 5/9(C+32)

"""

# convert temp from celsius to fahrenheit..

temperatures\_celsius = city\_data\_np[:, 5].astype(float)

temperatures\_fahrenheit = (temperatures\_celsius \* 9/5) + 32

# Update the temperature column with Fahrenheit temperatures

city\_data\_np[:, 5] = temperatures\_fahrenheit

# Print the updated array

print("Updated City Data with Temperatures in Fahrenheit:")

print(city\_data\_np)

"""# Question 3. You have been tasked with analyzing stock market data for a portfolio of companies. The data is stored in a CSV file with columns for company, date, open price, high price, low price, close price, and trading volume. Write a Python script using NumPy and Pandas to:

• Load the data into a Pandas DataFrame • Calculate the daily price range (high - low) for each company

• Find the company with the highest average daily price range

• Create a new DataFrame containing only the data for the company with the highest average daily price range

• Calculate the daily returns for this company and store them in a new column

• Perform a rolling window analysis to calculate the 5-day and 20-day moving average of the daily returns

• Plot the daily returns and the moving averages using Matplotlib

"""

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

"""# Load the stock market data"""

df\_stock = pd.read\_csv('/stock\_market\_data.csv')

df\_stock

"""# Calculate the daily price range (high - low) for each company"""

df\_stock['Daily Price Range'] = df\_stock['High Price'] - df\_stock['Low Price']

df\_stock

"""# Find the company with the highest average daily price range"""

mean\_price\_range = df\_stock.groupby('Company')['Daily Price Range'].mean()

highest\_avg\_price\_range = mean\_price\_range.idxmax()

highest\_avg\_price\_range

"""# Create a new DataFrame containing only the data for the company with the highest average daily price range"""

df\_highest\_avg\_range = df\_stock[df\_stock['Company'] == highest\_avg\_price\_range].copy()

df\_highest\_avg\_range

"""# Calculate the daily returns for this company and store them in a new column"""

df\_highest\_avg\_range['Daily Returns'] = df\_highest\_avg\_range['Close Price'].pct\_change()

df\_highest\_avg\_range

"""# • Perform a rolling window analysis to calculate the 5-day and 20-day moving average of the daily returns

"""

df\_highest\_avg\_range['5-day MA'] = df\_highest\_avg\_range['Daily Returns'].rolling(window=5).mean()

df\_highest\_avg\_range['20-day MA'] = df\_highest\_avg\_range['Daily Returns'].rolling(window=20).mean()

"""# Plot the daily returns and the moving averages using Matplotlib"""

import matplotlib.pyplot as plt

# Plot the daily returns and the moving averages using Matplotlib

plt.figure(figsize=(14, 7))

plt.plot(df\_highest\_avg\_range['Date'], df\_highest\_avg\_range['Daily Returns'], label='Daily Returns')

plt.plot(df\_highest\_avg\_range['Date'], df\_highest\_avg\_range['5-day MA'], label='5-day MA', color='orange')

plt.plot(df\_highest\_avg\_range['Date'], df\_highest\_avg\_range['20-day MA'], label='20-day MA', color='red')

plt.xlabel('Date')

plt.ylabel('Returns')

plt.title(f'Daily Returns and Moving Averages for {highest\_avg\_price\_range}')

plt.legend()

plt.show()