1. Scenario: You are working on a project that involves analyzing student performance data for a class of 32 students. The data is stored in a NumPy array named student\_scores, where each row represents a student and each column represents a different subject. The subjects are arranged in the following order: Math, Science, English, and History. Your task is to calculate the average score for each subject and identify the subject with the highest average score.

Question: How would you use NumPy arrays to calculate the average score for each subject and determine the subject with the highest average score? Assume 4x4 matrix that stores marks of each student in given order.

**CODE:**

import numpy as np

student\_scores = np.array([

[80, 90, 85, 70],

[75, 88, 78, 85],

[92, 81, 86, 78],

[88, 79, 90, 82]

])

subject\_averages = np.mean(student\_scores, axis=0)

max\_index = np.argmax(subject\_averages)

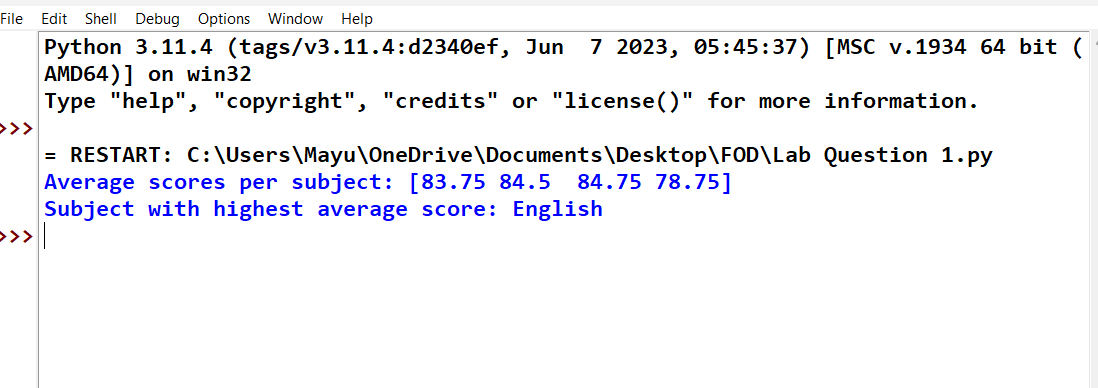
subjects = ["Math", "Science", "English", "History"]

highest\_avg\_subject = subjects[max\_index]

print("Average scores per subject:", subject\_averages)

print("Subject with highest average score:", highest\_avg\_subject)

**OUTPUT:**



2. Scenario: You are a data analyst working for a company that sells products online. You have been tasked with analyzing the sales data for the past month. The data is stored in a NumPy array.

Question: How would you find the average price of all the products sold in the past month?

Assume 3x3 matrix with each row representing the sales for a different product

**CODE:**

import numpy as np

sales\_data = np.array([

[10, 20, 200],

[15, 25, 375],

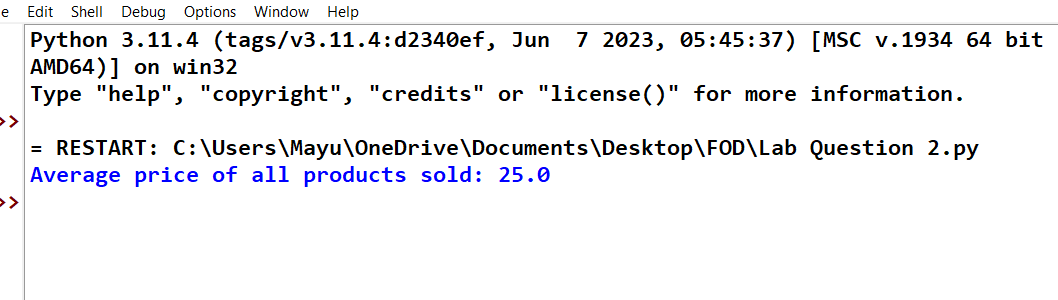
[20, 30, 600]

])

average\_price = np.mean(sales\_data[:, 1])

print("Average price of all products sold:", average\_price)

**OUTPUT:**

****

3. Scenario: You are working on a project that involves analyzing a dataset containing information about houses in a neighborhood. The dataset is stored in a CSV file, and you have imported it into a NumPy array named house\_data. Each row of the array represents a house, and the columns contain various features such as the number of bedrooms, square footage, and sale price.

Question: Using NumPy arrays and operations, how would you find the average sale price of

houses with more than four bedrooms in the neighborhood?

**CODE:**

import numpy as np

house\_data = np.array([

[3, 1500, 250000],

[5, 2500, 400000],

[4, 2000, 350000],

[6, 3000, 500000],

[3, 1800, 275000],

[7, 3500, 600000]

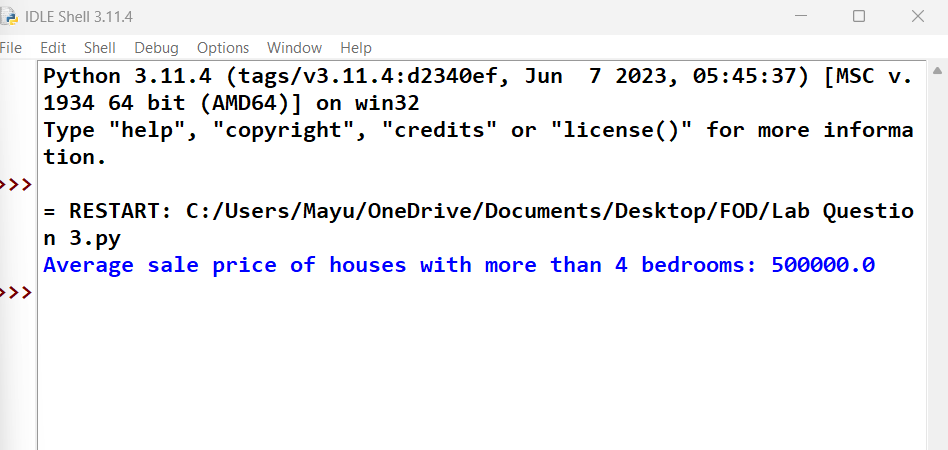
])

houses\_with\_more\_than\_4\_bedrooms = house\_data[house\_data[:, 0] > 4]

average\_sale\_price = np.mean(houses\_with\_more\_than\_4\_bedrooms[:, 2])

print("Average sale price of houses with more than 4 bedrooms:", average\_sale\_price)

**OUTPUT:**

****

4. Scenario: You are working on a project that involves analyzing the sales performance of a company over the past four quarters. The quarterly sales data is stored in a NumPy array named sales\_data, where each element represents the sales amount for a specific quarter. Your task is to calculate the total sales for the year and determine the percentage increase in sales from the first quarter to the fourth quarter.

Question: Using NumPy arrays and arithmetic operations calculate the total sales for the year and determine the percentage increase in sales from the first quarter to the fourth quarter?

**CODE:**

import numpy as np

sales\_data = np.array([15000, 18000, 21000, 25000])

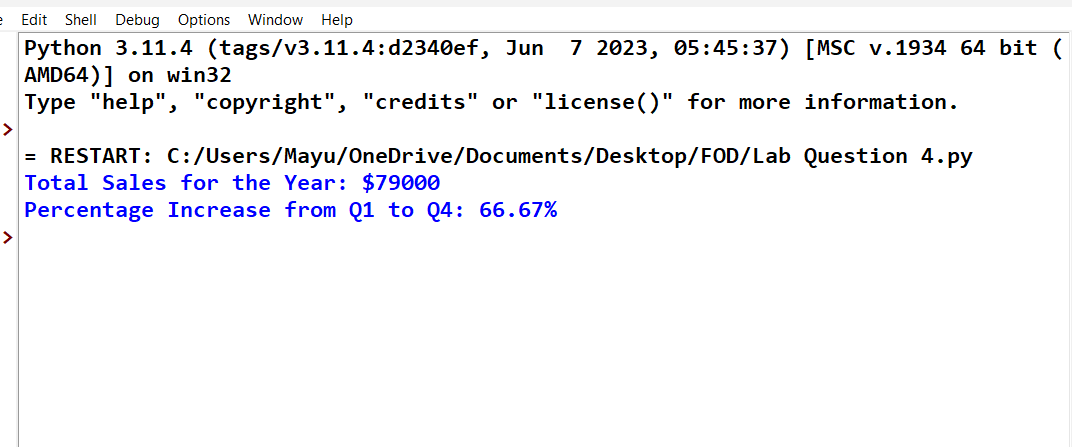
total\_sales = np.sum(sales\_data)

percentage\_increase = ((sales\_data[3] - sales\_data[0]) / sales\_data[0]) \* 100

print(f"Total Sales for the Year: ${total\_sales}")

print(f"Percentage Increase from Q1 to Q4: {percentage\_increase:.2f}%")

**OUTPUT:**



5. Scenario: You are a data analyst working for a car manufacturing company. As part of your analysis, you have a dataset containing information about the fuel efficiency of different car models. The dataset is stored in a NumPy array named fuel\_efficiency, where each element represents the fuel efficiency (in miles per gallon) of a specific car model. Your task is to calculate the average fuel efficiency and determine the percentage improvement in fuel efficiency between two car models.

Question: How would you use NumPy arrays and arithmetic operations to calculate the average fuel efficiency and determine the percentage improvement in fuel efficiency between two car models?

**CODE:**

import numpy as np

fuel\_efficiency = np.array([22, 25, 30, 35, 40])

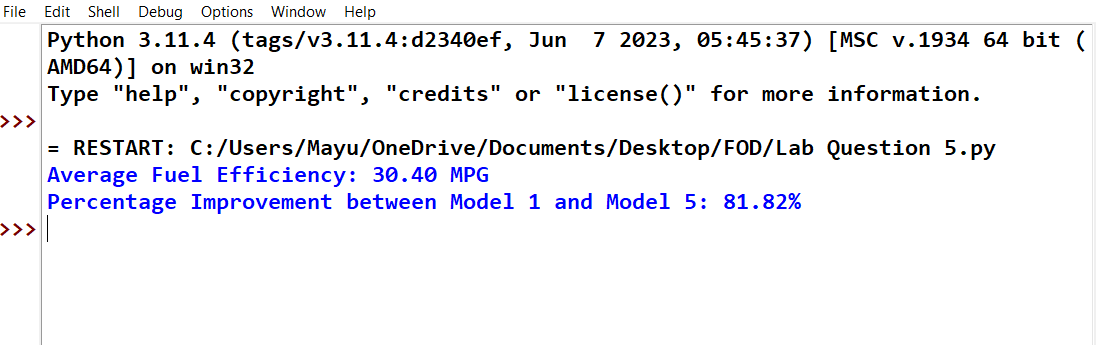
average\_efficiency = np.mean(fuel\_efficiency)

percentage\_improvement = ((fuel\_efficiency[4] - fuel\_efficiency[0]) / fuel\_efficiency[0]) \* 100

print(f"Average Fuel Efficiency: {average\_efficiency:.2f} MPG")

print(f"Percentage Improvement between Model 1 and Model 5: {percentage\_improvement:.2f}%"

**OUTPUT:**



6. Scenario: You are a cashier at a grocery store and need to calculate the total cost of a customer’s purchase, including applicable discounts and taxes. You have the item prices and quantities in separate lists, and the discount and tax rates are given as percentages. Your task is to calculate the total cost for the customer.

Question: Use arithmetic operations to calculate the total cost of a customer’s purchase, including discounts and taxes, given the item prices, quantities, discount rate, and tax rate?

**CODE:**

item\_prices = [10.0, 5.0, 2.5]

quantities = [2, 4, 3]

discount\_rate = 10

tax\_rate = 8

subtotals = [p \* q for p, q in zip(item\_prices, quantities)]

total\_before\_discount = sum(subtotals)

discount\_amount = total\_before\_discount \* (discount\_rate / 100)

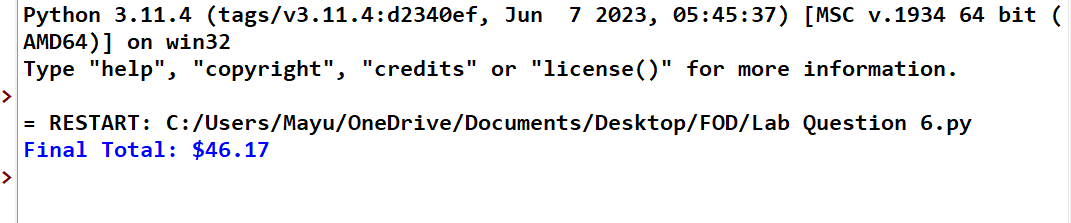
total\_after\_discount = total\_before\_discount - discount\_amount

tax\_amount = total\_after\_discount \* (tax\_rate / 100)

final\_total = total\_after\_discount + tax\_amount

print(f"Final Total: ${final\_total:.2f}")

**OUTPUT:**



7. Scenario: You are working as a data analyst for an e-commerce company. You have been given a dataset containing information about customer orders, stored in a Pandas DataFrame named order\_data. The DataFrame has columns for customer ID, order date, product name, and order quantity. Your task is to analyze the data and answer specific questions about the orders.

Question: Using Pandas DataFrame operations, how would you find the following information from the order\_data DataFrame:

1. The total number of orders made by each customer.

2. The average order quantity for each product.

3. The earliest and latest order dates in the dataset.

**CODE:**

import pandas as pd

order\_data = pd.DataFrame({

'customer\_id': [101, 102, 101, 103, 102],

'order\_date': pd.to\_datetime(['2023-01-10', '2023-01-15', '2023-02-01', '2023-01-20', '2023-03-05']),

'product\_name': ['Widget', 'Gadget', 'Widget', 'Widget', 'Gadget'],

'order\_quantity': [3, 5, 2, 4, 1]

})

orders\_per\_customer = order\_data['customer\_id'].value\_counts()

avg\_quantity\_per\_product = order\_data.groupby('product\_name')['order\_quantity'].mean()

earliest\_order = order\_data['order\_date'].min()

latest\_order = order\_data['order\_date'].max()

print("1.Orders per customer:")

print(orders\_per\_customer)

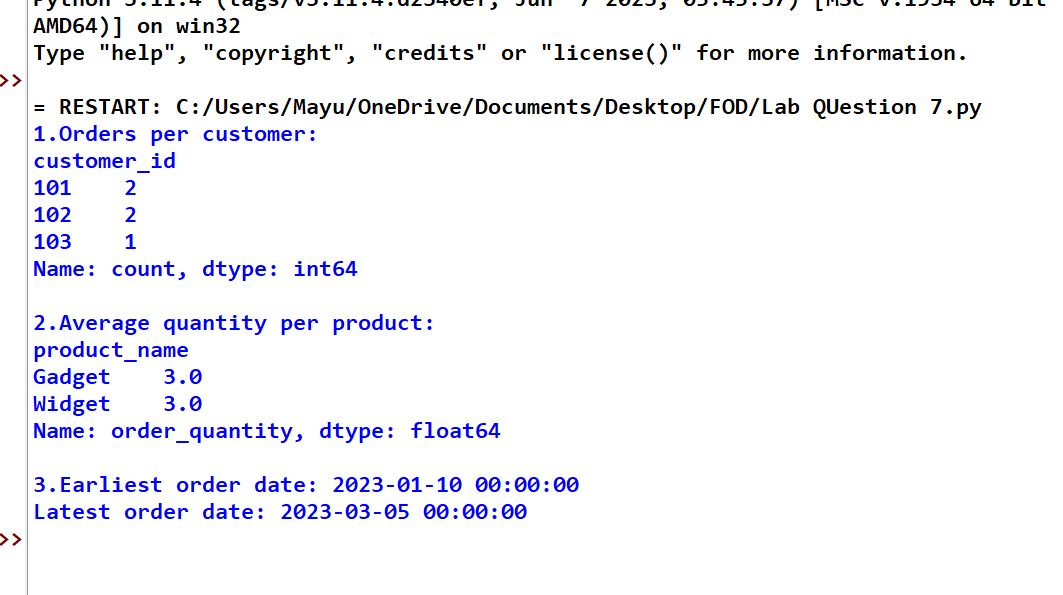
print("\n2.Average quantity per product:")

print(avg\_quantity\_per\_product)

print(f"\n3.Earliest order date: {earliest\_order}")

print(f"Latest order date: {latest\_order}")

**OUTPUT:**



8. Scenario: You are a data scientist working for a company that sells products online. You have been tasked with analyzing the sales data for the past month. The data is stored in a Pandas data frame.

Question: How would you find the top 5 products that have been sold the most in the past month?

**CODE:**

import pandas as pd

sales\_data = pd.DataFrame({

'product\_name': ['A', 'B', 'A', 'C', 'B', 'D', 'E', 'A', 'C', 'D'],

'quantity\_sold': [5, 3, 2, 7, 1, 4, 6, 3, 2, 5]

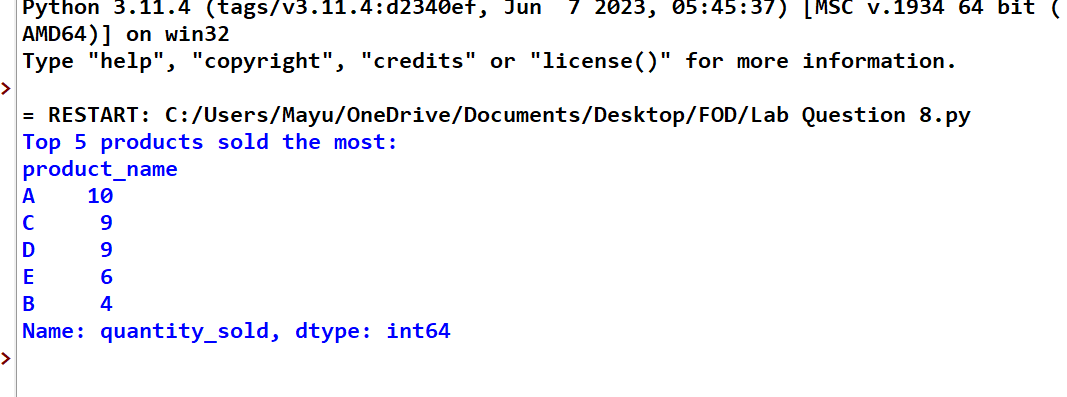
})

top\_products = sales\_data.groupby('product\_name')['quantity\_sold'].sum().sort\_values(ascending=False).head(5)

print("Top 5 products sold the most:")

print(top\_products)

**OUTPUT:**



9. Scenario: You work for a real estate agency and have been given a dataset containing information about properties for sale. The dataset is stored in a Pandas DataFrame named property\_data. The DataFrame has columns for property ID, location, number of bedrooms, area in square feet, and listing price. Your task is to analyze the data and answer specific questions about the properties.

Question: Using Pandas DataFrame operations, how would you find the following information from the property\_data DataFrame:

1. The average listing price of properties in each location.

2. The number of properties with more than four bedrooms.

3. The property with the largest area.

**CODE:**

import pandas as pd

property\_data = pd.DataFrame({

'property\_id': [1, 2, 3, 4, 5],

'location': ['Downtown', 'Suburb', 'Downtown', 'Suburb', 'Rural'],

'bedrooms': [3, 5, 4, 6, 2],

'area\_sqft': [1500, 2500, 1800, 3000, 1200],

'listing\_price': [300000, 450000, 350000, 500000, 200000]

})

avg\_price\_by\_location = property\_data.groupby('location')['listing\_price'].mean()

num\_properties\_gt\_4\_bedrooms = property\_data[property\_data['bedrooms'] > 4].shape[0]

largest\_area\_property = property\_data.loc[property\_data['area\_sqft'].idxmax()]

print("1.Average listing price by location:")

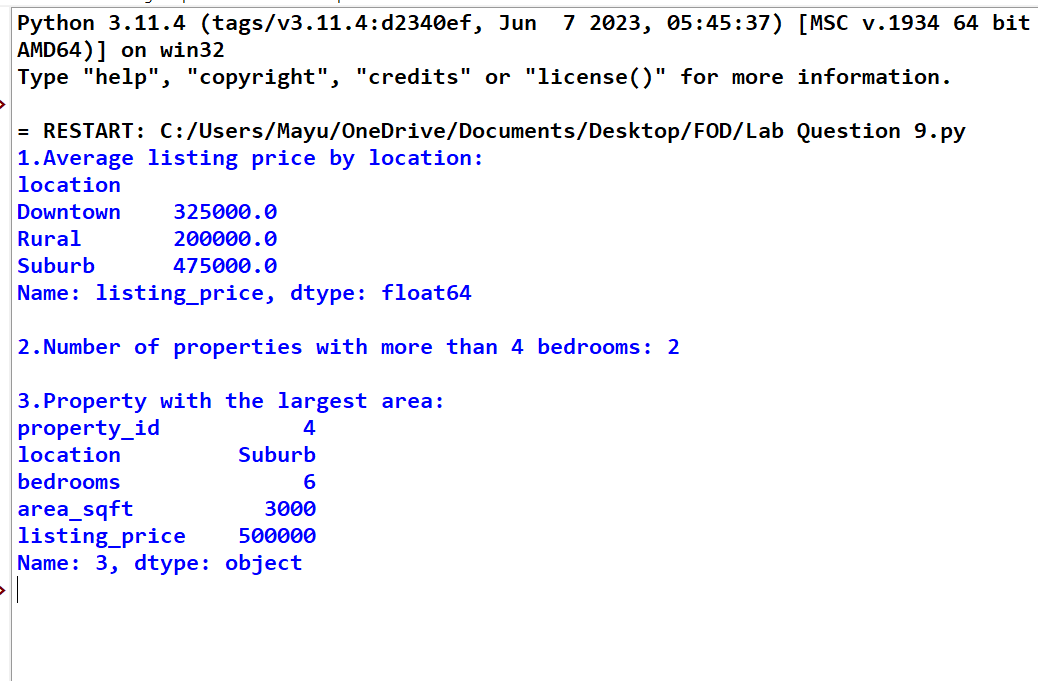
print(avg\_price\_by\_location)

print(f"\n2.Number of properties with more than 4 bedrooms: {num\_properties\_gt\_4\_bedrooms}")

print("\n3.Property with the largest area:")

print(largest\_area\_property)

**OUTPUT:**



10. Scenario: You are working on a data visualization project and need to create basic plots using Matplotlib. You have a dataset containing the monthly sales data for a company, including the month and corresponding sales values. Your task is to develop a Python program that generates line plots and bar plots to visualize the sales data.

Question:

1. How would you develop a Python program to create a line plot of the monthly sales data?

2: How would you develop a Python program to create a bar plot of the monthly sales data?

**CODE:**

import matplotlib.pyplot as plt

months = ['Jan', 'Feb', 'Mar', 'Apr', 'May', 'Jun']

sales = [1000, 1200, 1100, 1500, 1700, 1600]

plt.figure(figsize=(10, 4))

plt.subplot(1, 2, 1)

plt.plot(months, sales, marker='o', linestyle='-', color='b')

plt.title('Monthly Sales (Line Plot)')

plt.xlabel('Month')

plt.ylabel('Sales')

plt.subplot(1, 2, 2)

plt.bar(months, sales, color='pink')

plt.title('Monthly Sales (Bar Plot)')

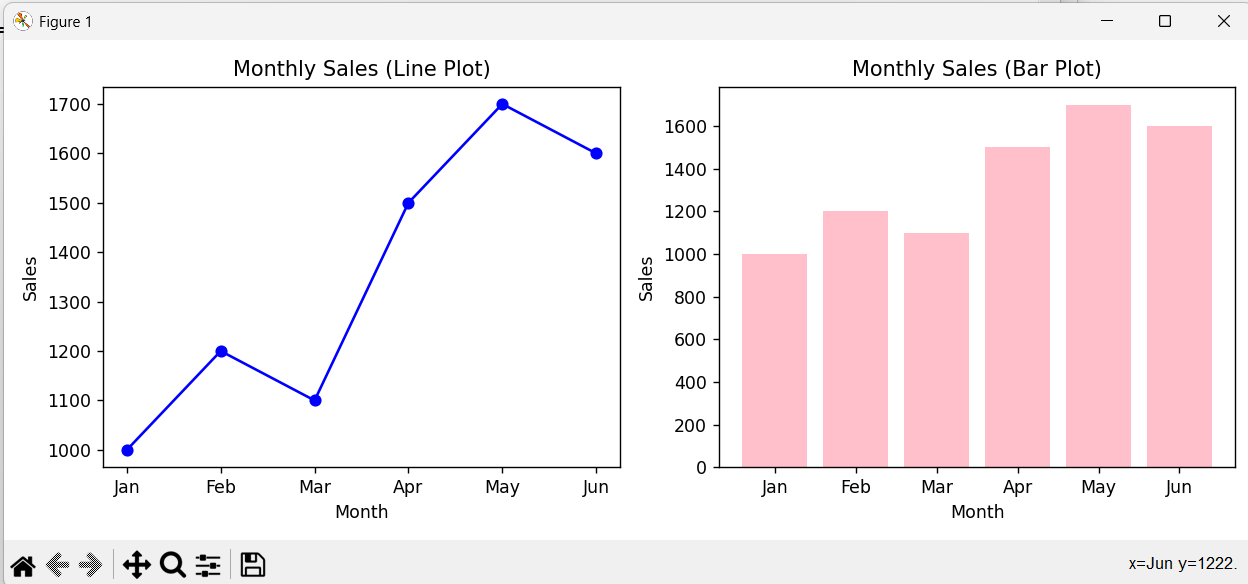
plt.xlabel('Month')

plt.ylabel('Sales')

plt.tight\_layout()

plt.show()

**OUTPUT:**

****

**11. Scenario : You are a data scientist working for a company that sells products online. You have been tasked with creating a simple plot to show the sales of a product over time.**

**Question:**

**1. Write code to create a simple line plot in Python using Matplotlib to predict sales happened in a month?**

**2. Write code to create a scatter plot in Python using Matplotlib to predict sales happened in a month?**

**3. Develop a Python program to create a bar plot of the monthly sales data.**

import matplotlib.pyplot as plt

months = ['Jan', 'Feb', 'Mar', 'Apr', 'May', 'Jun',

'Jul', 'Aug', 'Sep', 'Oct', 'Nov', 'Dec']

sales = [150, 200, 250, 300, 280, 350, 400, 420, 390, 450, 470, 500]

plt.figure(figsize=(8, 5))

plt.plot(months, sales, marker='o', linestyle='-', color='blue')

plt.title('NEWS.txtMonthly Sales (Line Plot)')

plt.xlabel('Month')

plt.ylabel('Sales')

plt.grid(True)

plt.show()

plt.figure(figsize=(8, 5))

plt.scatter(months, sales, color='green')

plt.title('2.Monthly Sales (Scatter Plot)')

plt.xlabel('Month')

plt.ylabel('Sales')

plt.grid(True)

plt.show()

plt.figure(figsize=(8, 5))

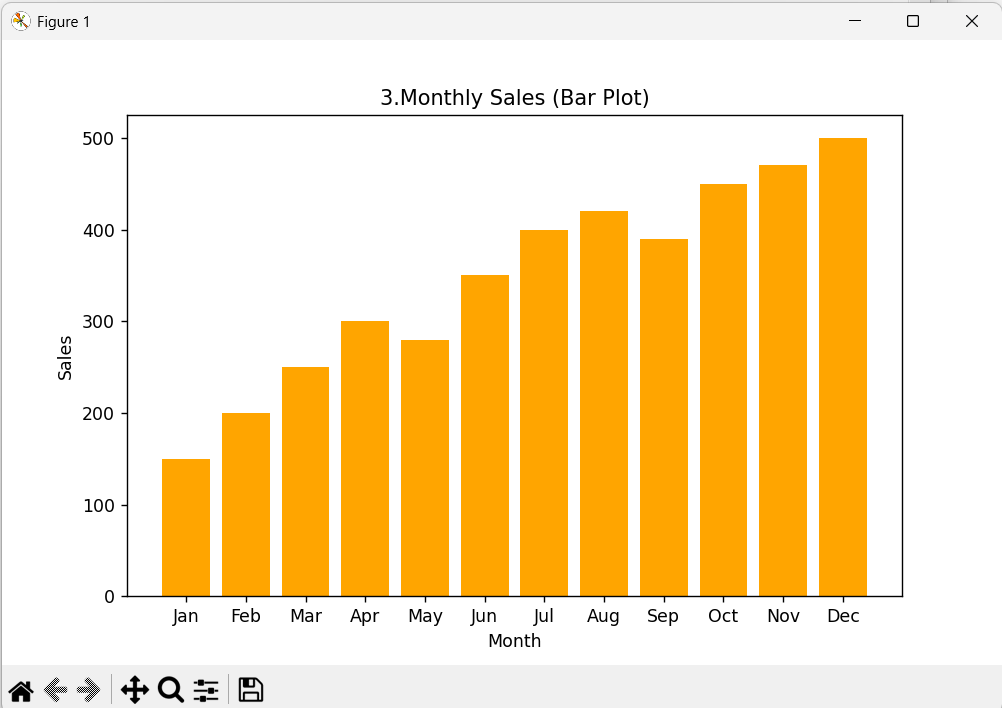
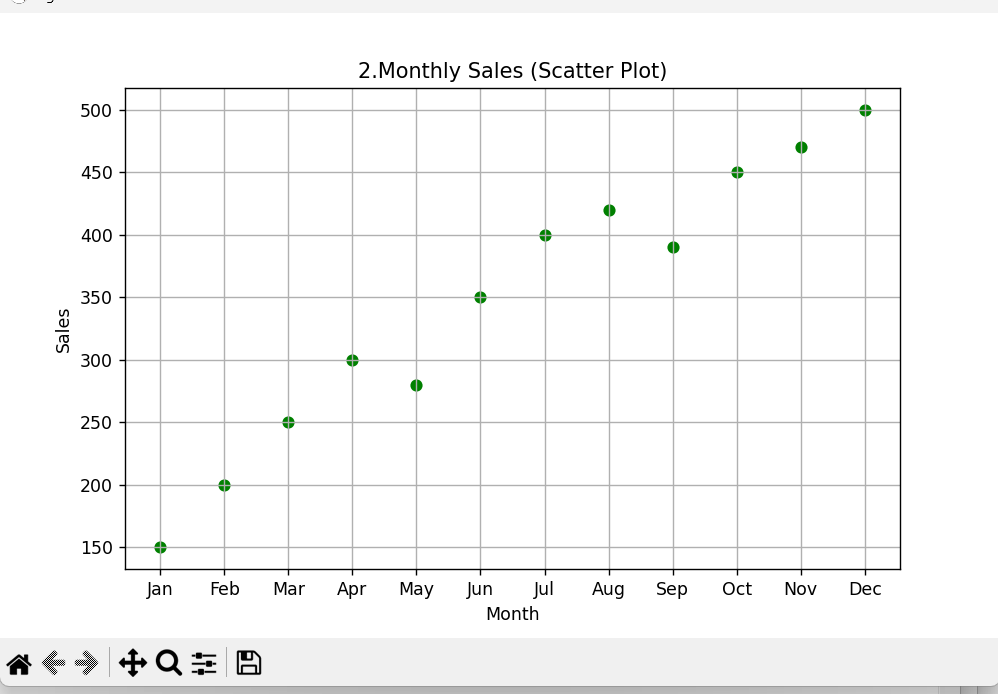
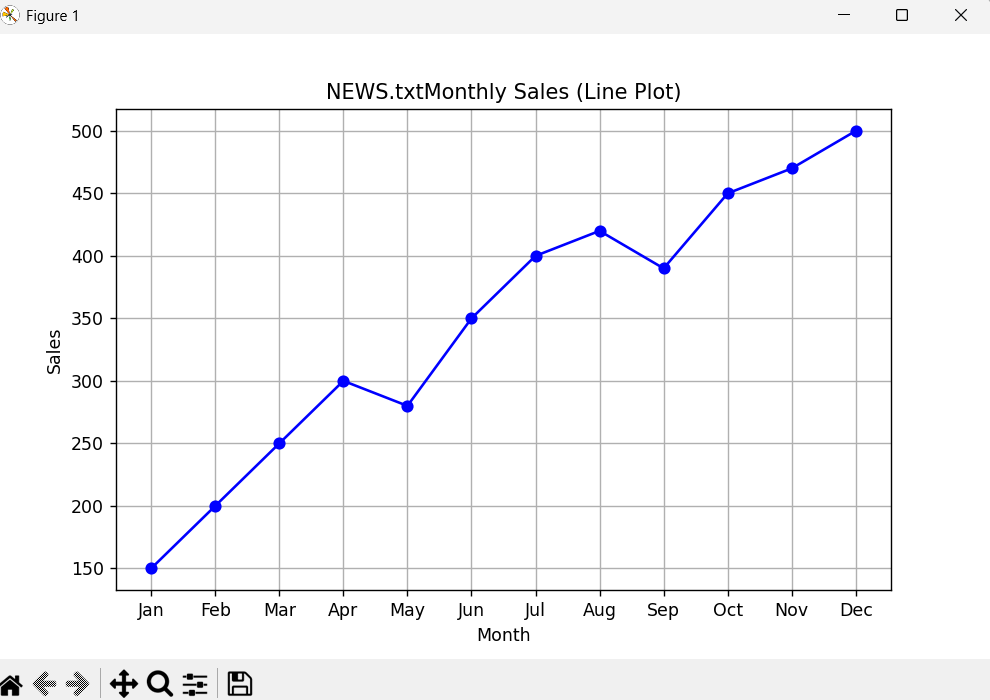
plt.bar(months, sales, color='orange')

plt.title('3.Monthly Sales (Bar Plot)')

plt.xlabel('Month')

plt.ylabel('Sales')

plt.show()

**12. Scenario: You are working on a data analysis project that involves analyzing the monthly temperature and rainfall data for a city. You have a dataset containing the monthly temperature and rainfall values for each month of a year. Your task is to develop a Python program that generates line plots and scatter plots to visualize the temperature and rainfall data.**

**Question:**

**1. Develop a Python program to create a line plot of the monthly temperature data.**

**2: Develop a Python program to create a scatter plot of the monthly rainfall data.**

import matplotlib.pyplot as plt

months = ['Jan', 'Feb', 'Mar', 'Apr', 'May', 'Jun',

'Jul', 'Aug', 'Sep', 'Oct', 'Nov', 'Dec']

temperature = [4, 6, 10, 15, 20, 25, 28, 27, 22, 16, 9, 5]

rainfall = [78, 60, 72, 55, 48, 35, 30, 40, 58, 70, 85, 90]

plt.figure(figsize=(8, 5))

plt.plot(months, temperature, marker='o', color='red')

plt.title('1.Monthly Temperature (Line Plot)')

plt.xlabel('Month')

plt.ylabel('Temperature (°C)')

plt.grid(True)

plt.show()

plt.figure(figsize=(8, 5))

plt.scatter(months, rainfall, color='blue')

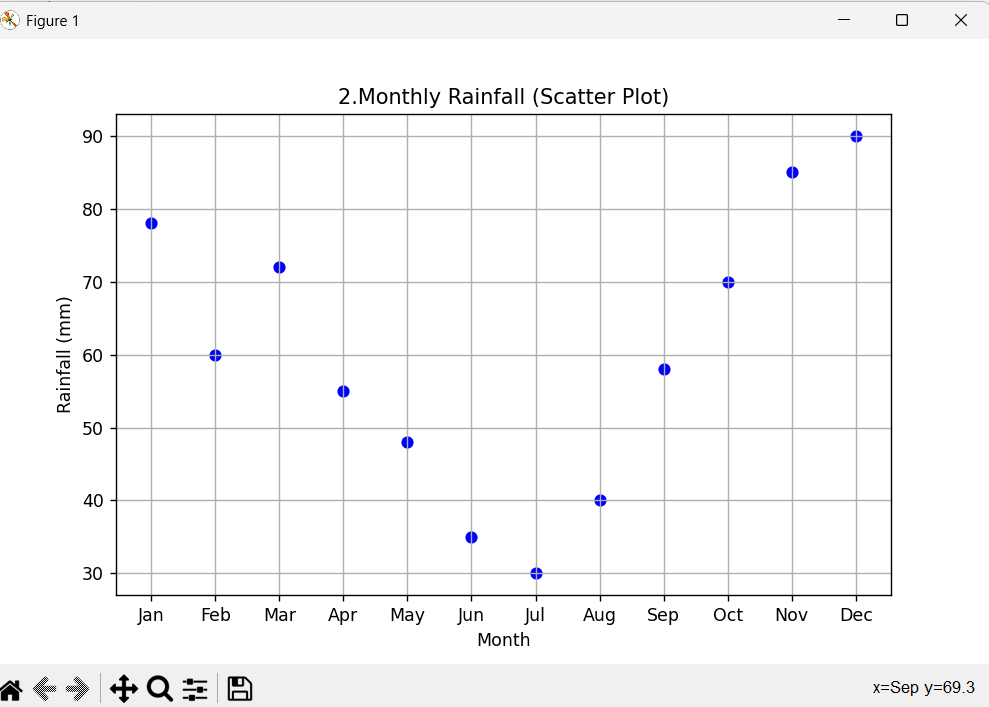
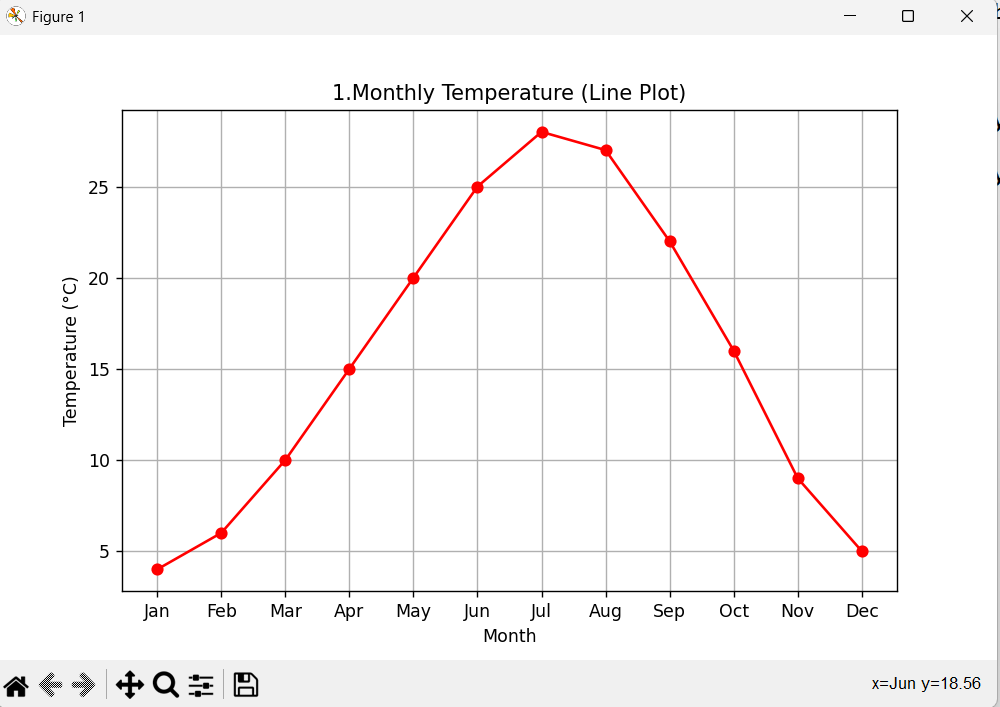
plt.title('2.Monthly Rainfall (Scatter Plot)')

plt.xlabel('Month')

plt.ylabel('Rainfall (mm)')

plt.grid(True)

plt.show()



**13. Scenario: You are working on a text analysis project and need to determine the frequency distribution of words in a given text document. You have a text document named “sample\_text.txt” containing a paragraph of text. Your task is to develop a Python program that reads the text document, processes the text, and generates a frequency distribution of the words.**

**Question: How would you develop a Python program to calculate the frequency distribution of words in a text document?**

import string

from collections import Counter

with open('sample\_text.txt', 'r') as file:

text = file.read()

text = text.lower()

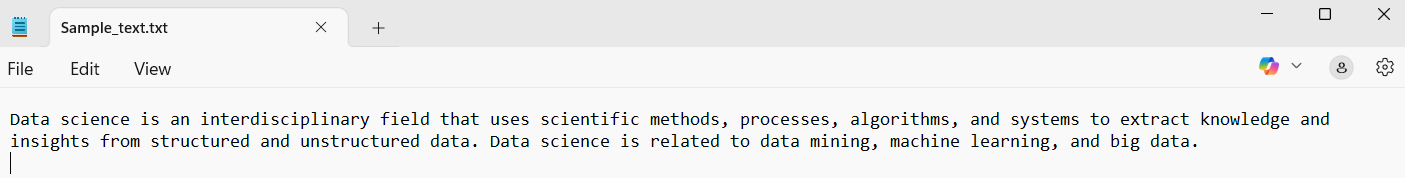
text = text.translate(str.maketrans('', '', string.punctuation))

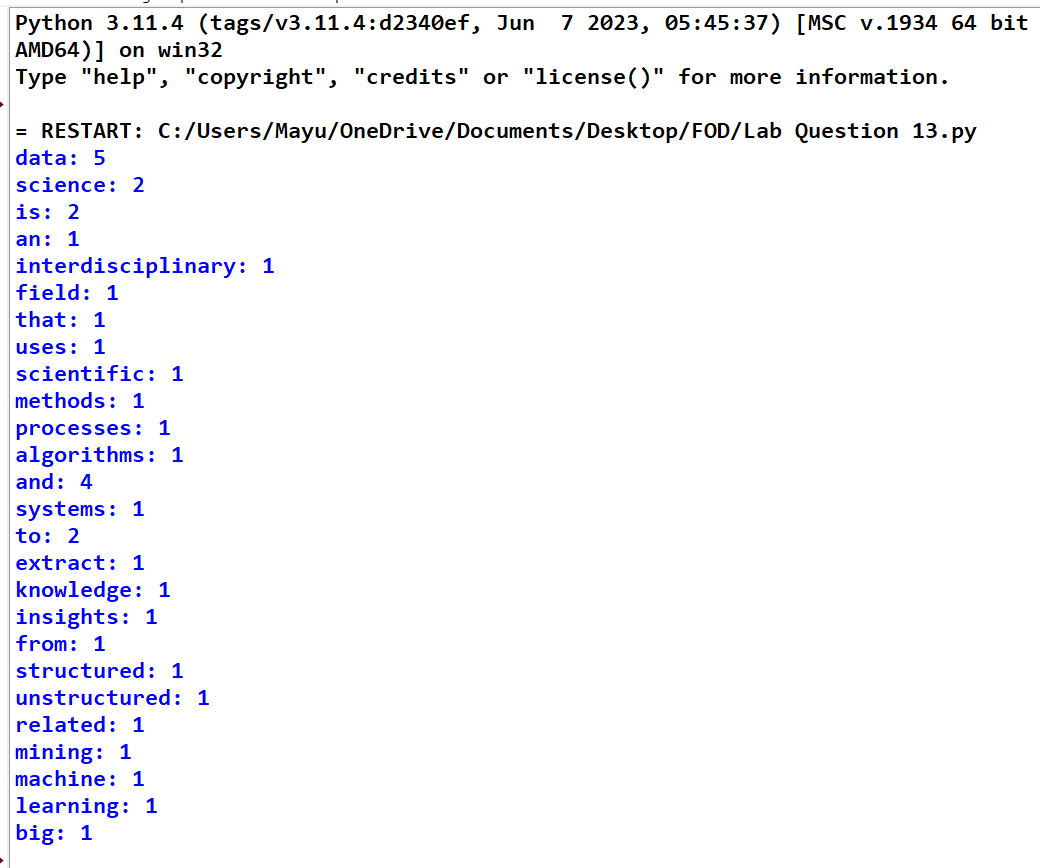
words = text.split()

word\_freq = Counter(words)

for word, freq in word\_freq.items():

print(f"{word}: {freq}")





**14. Scenario: You are a data analyst working for a company that sells products online. You have been tasked with analyzing the sales data for the past month. The data is stored in a Pandas data frame.**

**Question: Develop a code in python to find the frequency distribution of the ages of the customers who have made a purchase in the past month.**

import pandas as pd

data = {

'CustomerID': [101, 102, 103, 104, 105, 106, 107, 108],

'Age': [25, 30, 22, 25, 30, 40, 22, 25],

'PurchaseAmount': [200, 150, 180, 210, 160, 300, 190, 220]

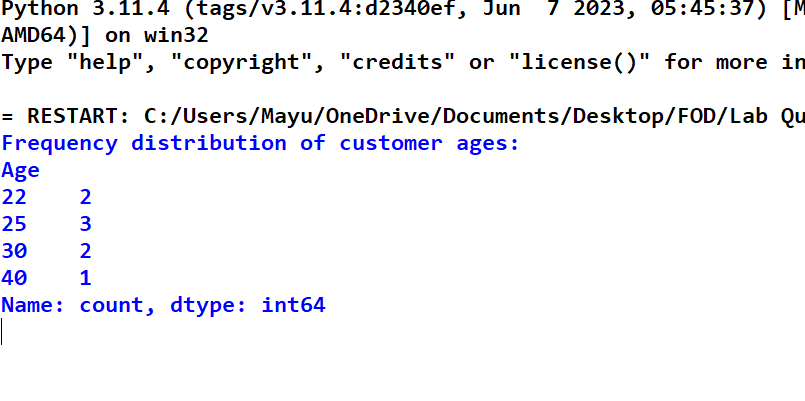
}

df = pd.DataFrame(data)

age\_frequency = df['Age'].value\_counts().sort\_index()

print("Frequency distribution of customer ages:")

print(age\_frequency)



**15. Scenario: You are a data analyst working for a social media platform. As part of your analysis, you have a dataset containing user interaction data, including the number of likes received by each post. Your task is to develop a Python program that calculates the frequency distribution of likes among the posts.**

**Question: Develop a Python program to calculate the frequency distribution of likes among the posts?**

import pandas as pd

data = {

'PostID': [201, 202, 203, 204, 205, 206, 207, 208],

'Likes': [10, 15, 10, 20, 15, 10, 25, 20]

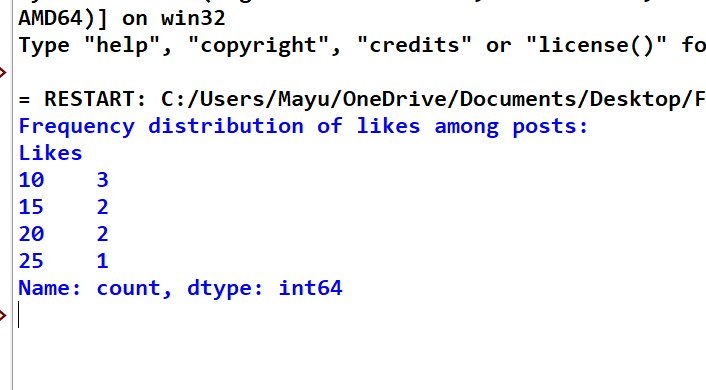
}

df = pd.DataFrame(data)

like\_frequency = df['Likes'].value\_counts().sort\_index()

print("Frequency distribution of likes among posts:")

print(like\_frequency)

****

**16. Scenario: You are working on a project that involves analyzing customer reviews for a product. You have a dataset containing customer reviews, and your task is to develop a Python program that calculates the frequency distribution of words in the reviews.**

**Question: Develop a Python program to calculate the frequency distribution of words in the customer reviews dataset?**

import pandas as pd

from collections import Counter

import string

data = {

'ReviewID': [1, 2, 3, 4],

'ReviewText': [

"Great product, really loved it!",

"Good quality, but too expensive.",

"Amazing product, worth the price.",

"Not bad, but expected better quality."

]

}

df = pd.DataFrame(data)

all\_reviews = ' '.join(df['ReviewText'].str.lower())

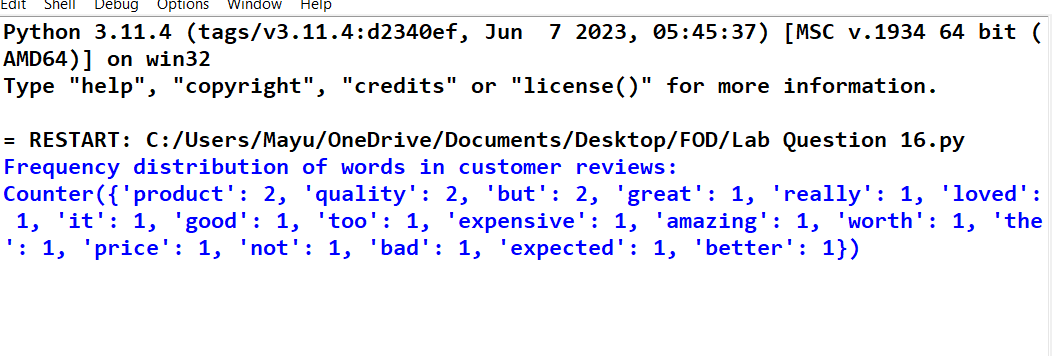
all\_reviews = all\_reviews.translate(str.maketrans('', '', string.punctuation))

words = all\_reviews.split()

word\_freq = Counter(words)

print("Frequency distribution of words in customer reviews:")

print(word\_freq)

****

**17. Scenario: You are a data analyst working for a marketing research company. Your team has collected a large dataset containing customer feedback from various social media platforms. The dataset consists of thousands of text entries, and your task is to develop a Python program to analyze the frequency distribution of words in this dataset. Your program should be able to perform the following tasks:**

** Load the dataset from a CSV file (data.csv) containing a single column named “feedback”**

**with each row representing a customer comment.**

** Preprocess the text data by removing punctuation, converting all text to lowercase, and**

**eliminating any stop words (common words like “the,” “and,” “is” etc. that don’t carry**

**significant meaning).**

** Calculate the frequency distribution of words in the preprocessed dataset.**

** Display the top N most frequent words and their corresponding frequencies, where N is**

**provided as user input.**

** Plot a bar graph to visualize the top N most frequent words and their frequencies.**

**Question: Create a Python program that fulfills these requirements and gain insights from the customer feedback data.**

import pandas as pd

import string

from collections import Counter

import matplotlib.pyplot as plt

from nltk.corpus import stopwords

import nltk

nltk.download('stopwords')

df = pd.read\_csv('data.csv')

stop\_words = set(stopwords.words('english'))

df['feedback'] = df['feedback'].str.lower()

df['feedback'] = df['feedback'].apply(lambda x: x.translate(str.maketrans('', '', string.punctuation)))

def preprocess\_text(text):

words = text.split()

return [word for word in words if word not in stop\_words]

df['processed\_feedback'] = df['feedback'].apply(preprocess\_text)

all\_words = [word for feedback in df['processed\_feedback'] for word in feedback]

word\_freq = Counter(all\_words)

N = int(input("Enter the number of top frequent words to display: "))

top\_n\_words = word\_freq.most\_common(N)

print(f"\nTop {N} most frequent words:")

for word, freq in top\_n\_words:

print(f"{word}: {freq}")

words, frequencies = zip(\*top\_n\_words)

plt.figure(figsize=(10, 6))

plt.bar(words, frequencies, color='skyblue')

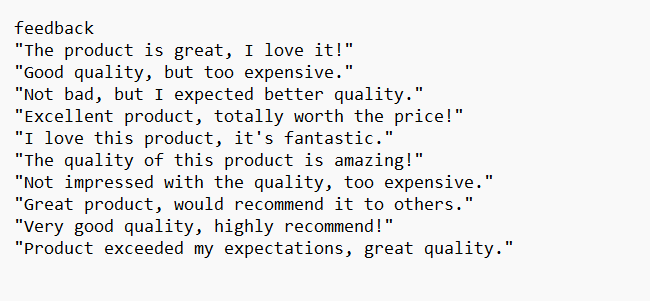
plt.title(f"Top {N} Most Frequent Words in Customer Feedback")

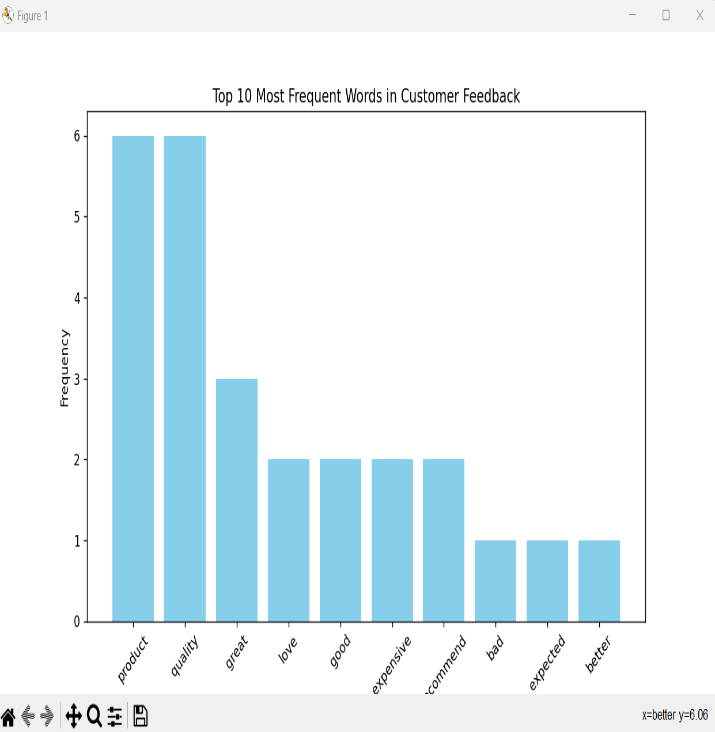
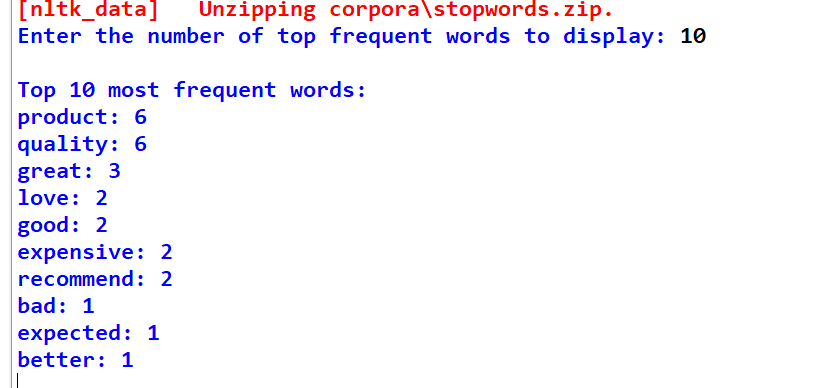
plt.xlabel('Words')

plt.ylabel('Frequency')

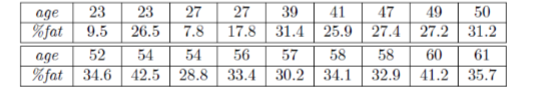
plt.xticks(rotation=45)

plt.show()





**18. Suppose a hospital tested the age and body fat data for 18 randomly selected adults with the following result.**

****

**Question:**

** Calculate the mean, median and standard deviation of age and %fat using Pandas.**

** Draw the boxplots for age and %fat.**

** Draw a scatter plot and a q-q plot based on these two variables.**

**CODE:**

import pandas as pd

import matplotlib.pyplot as plt

import statsmodels.api as sm

data = {

'age': [23, 23, 27, 27, 39, 41, 47, 49, 50,

52, 54, 54, 56, 57, 58, 58, 60, 61],

'%fat': [9.5, 26.5, 7.8, 17.8, 31.4, 25.9, 27.4, 27.2, 31.2,

34.6, 42.5, 28.8, 33.4, 30.2, 34.1, 32.9, 41.2, 35.7]

}

df = pd.DataFrame(data)

stats = df.agg(['mean', 'median', 'std'])

plt.figure(figsize=(15, 10))

plt.subplots\_adjust(hspace=0.4, wspace=0.3)

plt.subplot(2, 2, 1)

df.boxplot(column='age')

plt.title('Age Distribution')

plt.subplot(2, 2, 2)

df.boxplot(column='%fat')

plt.title('Body Fat Percentage')

plt.subplot(2, 2, 3)

plt.scatter(df['age'], df['%fat'], c='teal', alpha=0.7)

plt.xlabel('Age')

plt.ylabel('% Fat')

plt.grid(True, linestyle='--', alpha=0.7)

plt.subplot(2, 2, 4)

sm.qqplot(df['age'], line='s', label='Age')

sm.qqplot(df['%fat'], line='s', label='% Fat')

plt.legend()

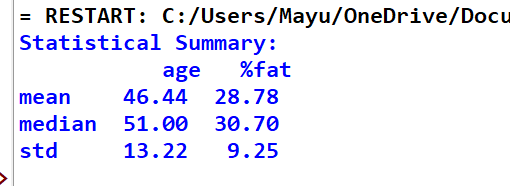
plt.title('Q-Q Plot Comparison')

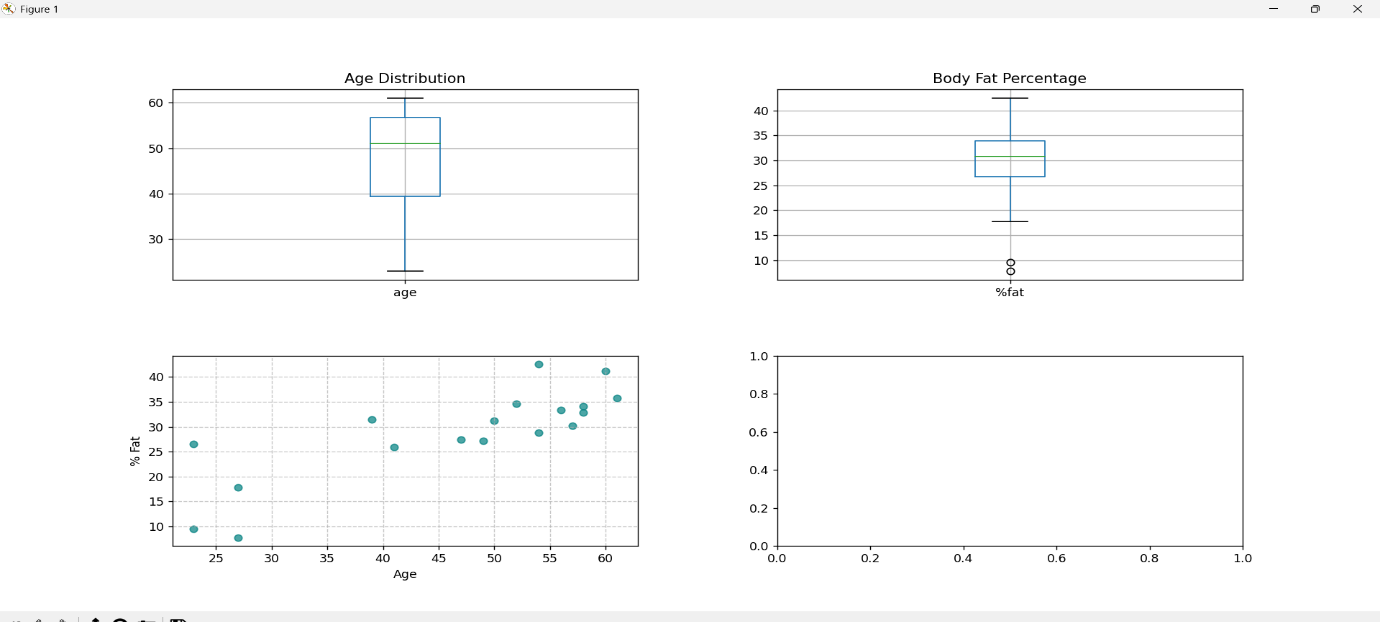
plt.tight\_layout()

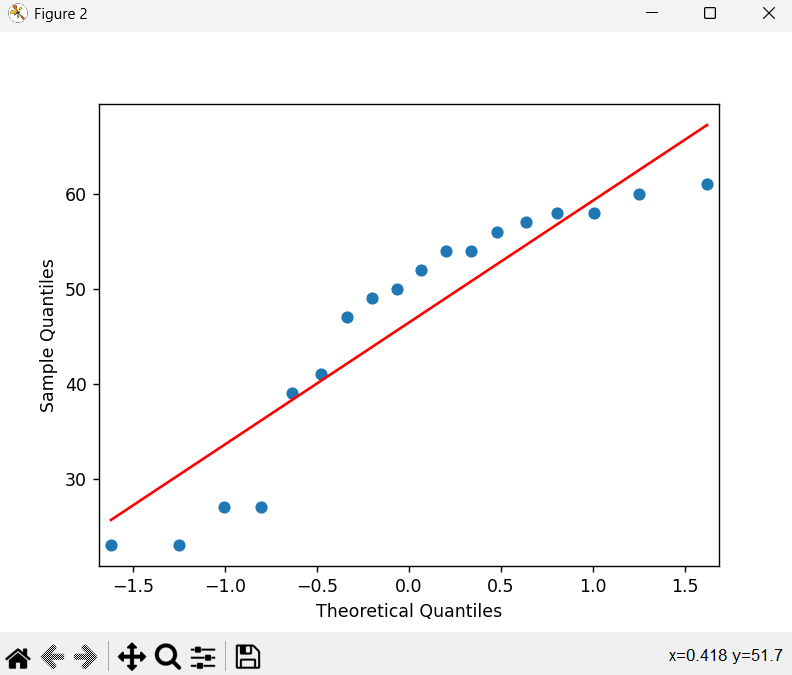
plt.show()

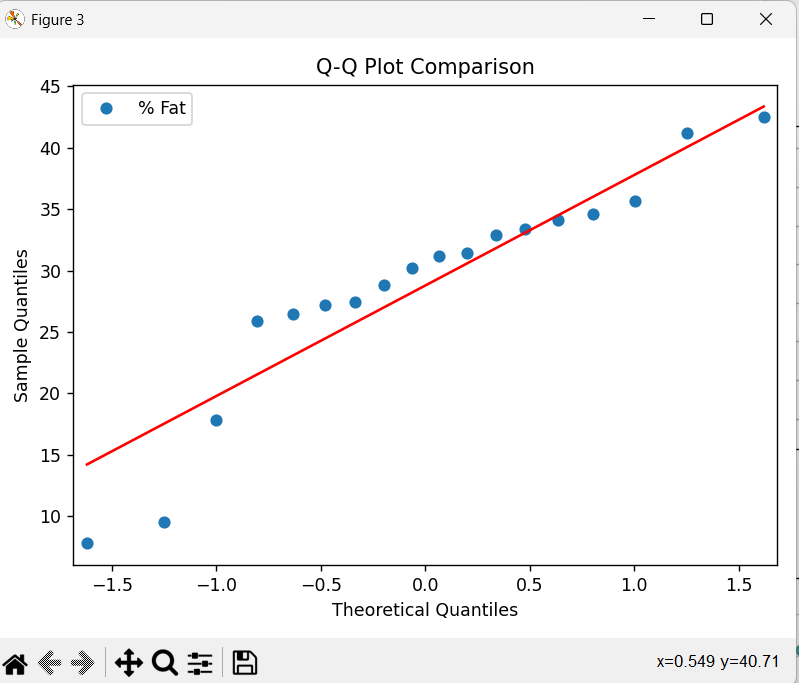
print("Statistical Summary:\n", stats.round(2))

**OUTPUT:**









**19. Sales and Profit Analysis: a) Load the “sales\_data.csv” file into a Pandas data frame, which contains columns “Date,” “Product,” “Quantity Sold,” and “Unit Price”**

**b) Create a new column named “Total Sales” that calculates the total sales for each transaction (Quantity Sold \* Unit Price).**

**c) Calculate the total sales for each product and the overall profit, considering a 20% profit margin on each product. Display the top 5 most profitable products.**

**CODE:**

import pandas as pd

df = pd.read\_csv('sales\_data.csv')

df['Total Sales'] = df['Quantity Sold'] \* df['Unit Price']

product\_sales = df.groupby('Product').agg({'Total Sales': 'sum'}).reset\_index()

product\_sales['Profit'] = product\_sales['Total Sales'] \* 0.20

overall\_profit = product\_sales['Profit'].sum()

top\_products = product\_sales.nlargest(5, 'Profit')

print("Total Sales per Product:")

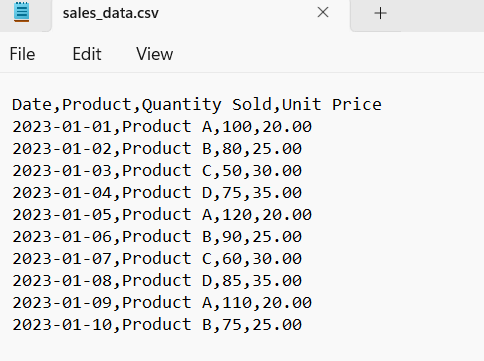
print(product\_sales.sort\_values('Total Sales', ascending=False).to\_string(index=False))

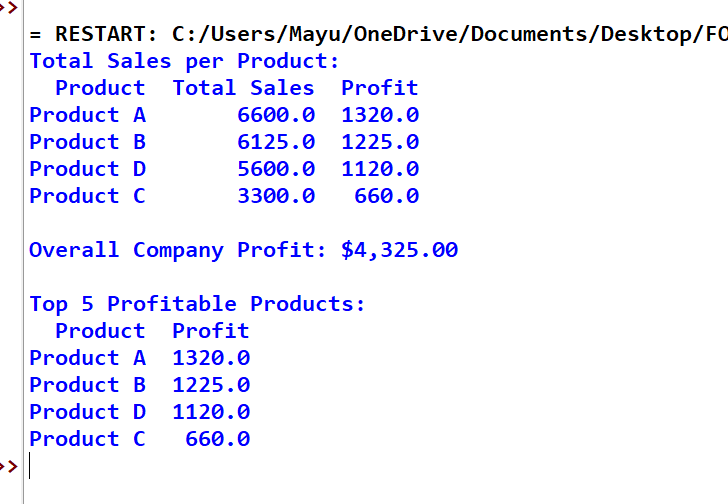
print(f"\nOverall Company Profit: ${overall\_profit:,.2f}")

print("\nTop 5 Profitable Products:")

print(top\_products[['Product', 'Profit']].to\_string(index=False))

**OUTPUT:**

****



**20. Customer Segmentation: a) Load “customer\_data.” file into a Pandas data frame, which**

**contains “Customer ID,” ”Age,” “Gender,” and “Total Spending.”**

**b) Segment customers into three groups based on their total spending: “High Spenders,” ”Medium Spenders,” and “Low Spenders.” Assign these segments to a new column in the data frame.**

**c) Calculate the average age of customers in each spending segment.**

**CODE:**

import pandas as pd

df = pd.read\_csv('customer\_data.csv')

quantiles = df['Total Spending'].quantile([0.33, 0.67])

df['Spending Segment'] = pd.cut(df['Total Spending'], bins=[-1, quantiles[0.33], quantiles[0.67], float('inf')],

labels=['Low Spenders', 'Medium Spenders', 'High Spenders'])

avg\_age = df.groupby('Spending Segment')['Age'].mean()

print("Customer Segmentation:")

print(df[['Customer ID', 'Spending Segment']])

print("\nAverage Age per Spending Segment:")

print(avg\_age)

print("\nData Quality Checks:")

df.info()

print("\nMissing Values:")

print(df.isna().sum())

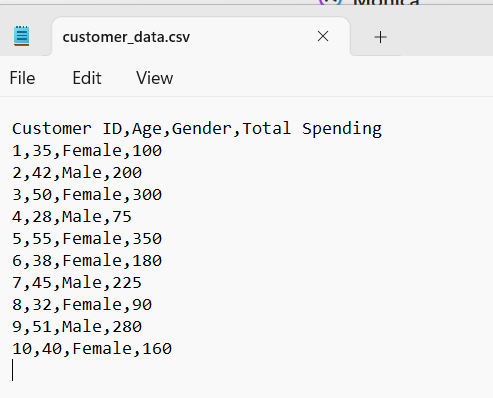
print("\nGender Distribution:")

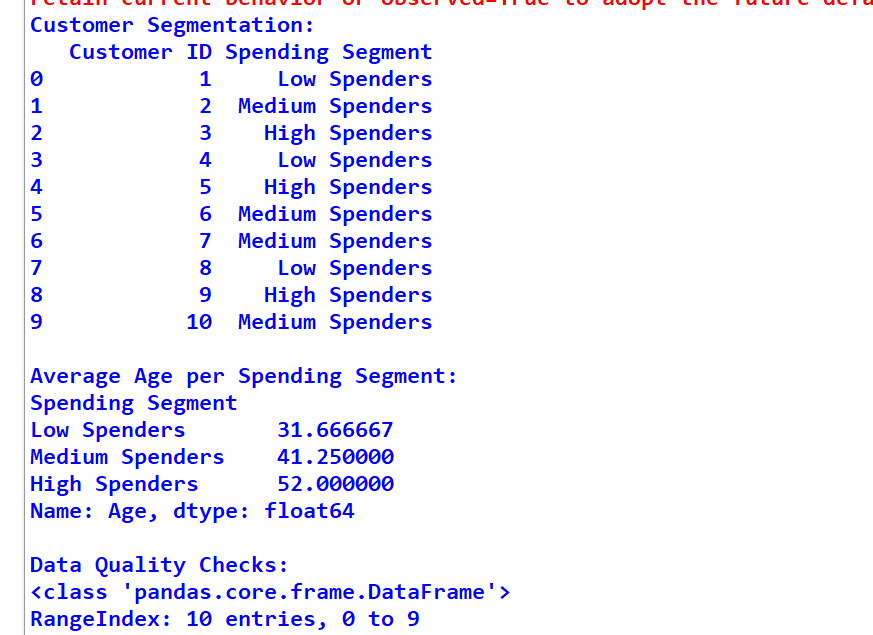
print(df['Gender'].value\_counts())

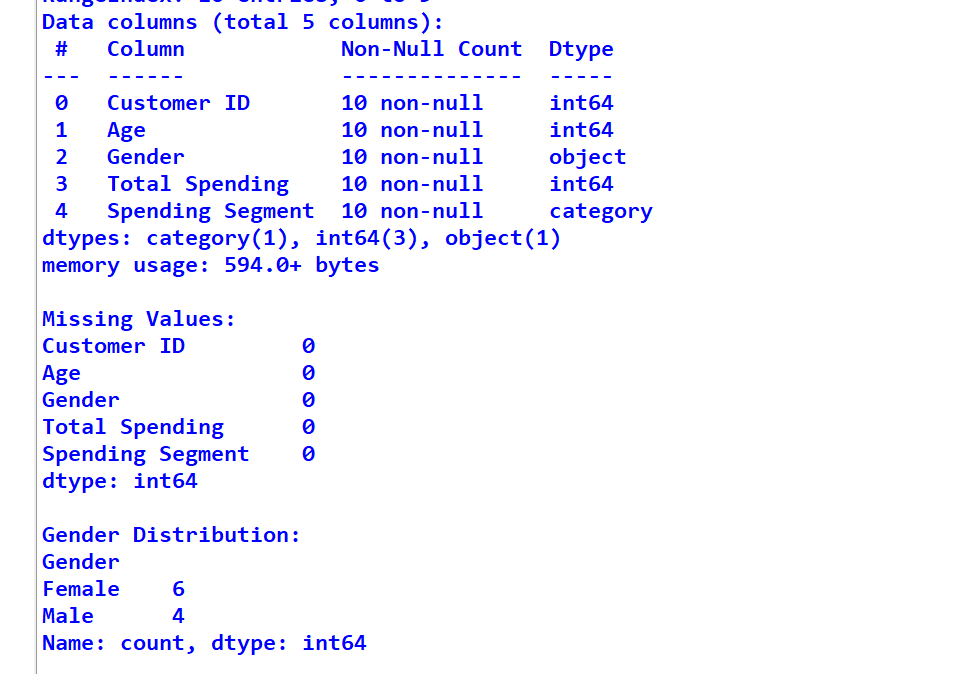
print("\nSpending Segment Statistics:")

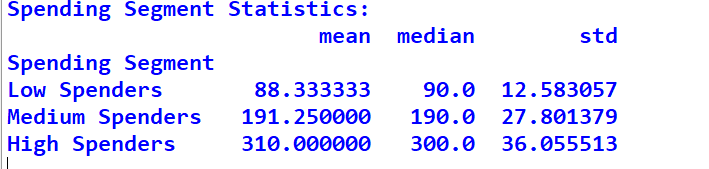
print(df.groupby('Spending Segment', observed=True)['Total Spending'].agg(['mean', 'median', 'std']))

**OUTPUT:**

****







**21. Data Cleaning and Transformation**

**Task:**

* **a) Load the employee\_data.csv file into a Pandas DataFrame. The file contains the columns:  
  Employee ID, Full Name, Department, and Salary.**
* **b) Convert the Salary column to a numeric data type.**
* **c) Remove any rows where the Department column has missing values.**
* **d) Create a new column named First Name that extracts the first name from the Full Name column.**

**CODE:**

import pandas as pd

employee\_df = pd.read\_csv("employee\_data.csv")

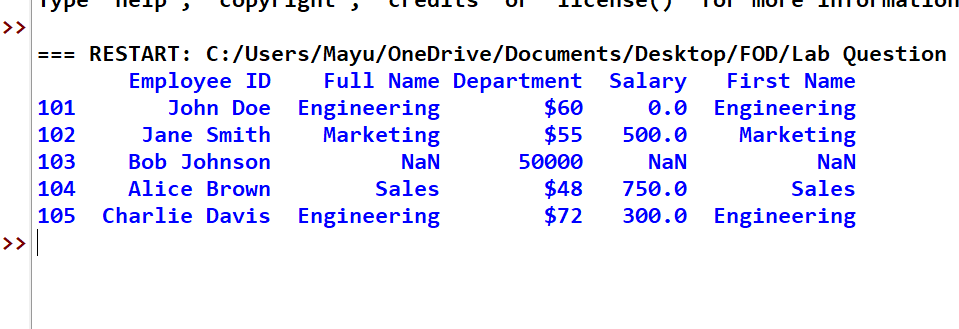
employee\_df['Salary'] = pd.to\_numeric(employee\_df['Salary'].replace('[\$,]', '', regex=True), errors='coerce')

employee\_df = employee\_df.dropna(subset=["Department"])

employee\_df['First Name'] = employee\_df['Full Name'].str.split().str[0]

print(employee\_df.head())

**OUTPUT:**



**22. Time Series Analysis**

**Task:**

* **a) Load the temperature\_data.csv file into a Pandas DataFrame. The file contains the columns:  
  Date and Temperature (Celsius).**
* **b) Convert the Date column to the Pandas datetime data type.**
* **c) Calculate the average temperature for each month and display the results in chronological order.**
* **d) Plot a line chart to visualize the temperature trend over time.**

**CODE:**

import pandas as pd

import matplotlib.pyplot as plt

temp\_df = pd.read\_csv("temperature\_data.csv")

temp\_df['Date'] = pd.to\_datetime(temp\_df['Date'])

temp\_df['Month'] = temp\_df['Date'].dt.to\_period('M')

monthly\_avg\_temp = temp\_df.groupby('Month')['Temperature (Celsius)'].mean().reset\_index()

monthly\_avg\_temp['Month'] = monthly\_avg\_temp['Month'].dt.to\_timestamp()

plt.figure(figsize=(10, 5))

plt.plot(temp\_df['Date'], temp\_df['Temperature (Celsius)'])

plt.plot(monthly\_avg\_temp['Month'], monthly\_avg\_temp['Temperature (Celsius)'], color='red')

plt.title('Temperature Trend Over Time')

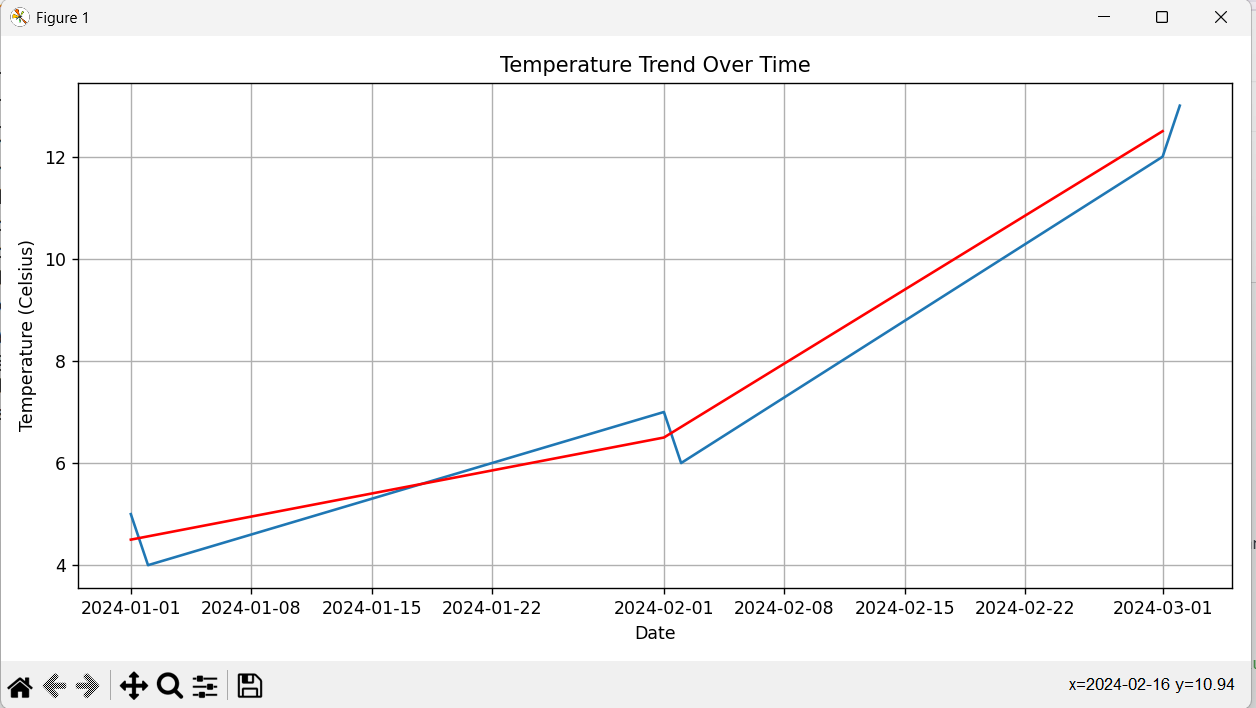
plt.xlabel('Date')

plt.ylabel('Temperature (Celsius)')

plt.grid(True)

plt.tight\_layout()

plt.show()  
**OUTPUT:**



**23. Joining DataFrames**

**Task:**

* **a) Load the orders\_data.csv file into a Pandas DataFrame. This file contains:  
  Order ID, Customer ID, and Order Date.**
* **b) Load the customer\_info.csv file into another Pandas DataFrame. This file contains:  
  Customer ID, Name, Email, and Phone Number.**
* **c) Merge the two DataFrames on the Customer ID column to create a single DataFrame that includes both order and customer details.**
* **d) Calculate the average time it takes for a customer to place another order after their first one (i.e., time between consecutive orders).**

**CODE:**

import pandas as pd

orders\_df = pd.read\_csv("order\_data.csv")

customers\_df = pd.read\_csv("customer\_info.csv")

merged\_df = pd.merge(orders\_df, customers\_df, on="Customer ID", how="inner")

print(merged\_df)

merged\_df['Order Date'] = pd.to\_datetime(merged\_df['Order Date'])

merged\_df = merged\_df.sort\_values(by=['Customer ID', 'Order Date'])

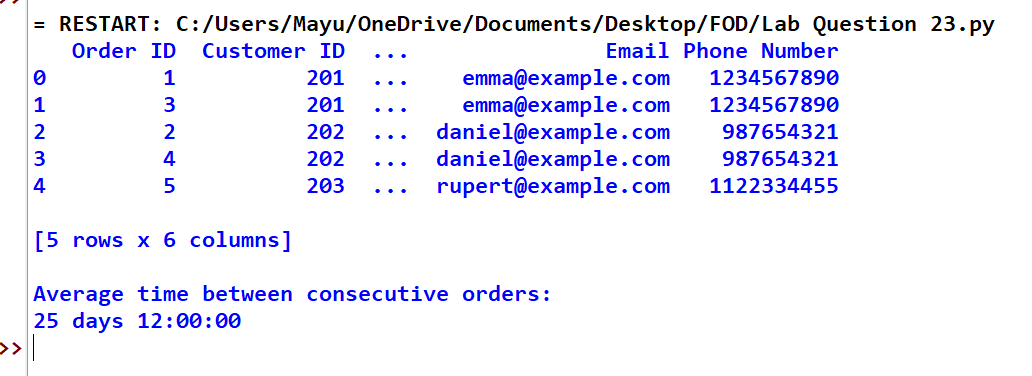
merged\_df['Time Diff'] = merged\_df.groupby('Customer ID')['Order Date'].diff()

avg\_time\_diff = merged\_df['Time Diff'].dropna().mean()

print("\nAverage time between consecutive orders:")

print(avg\_time\_diff)

**OUTPUT:**

****