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

import pandas as pd

df = pd.read\_csv(r"C:\Users\jampa\OneDrive\文档\student\_scores.csv")

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

average\_scores = df[subjects].mean()

highest\_avg\_subject = average\_scores.idxmax()

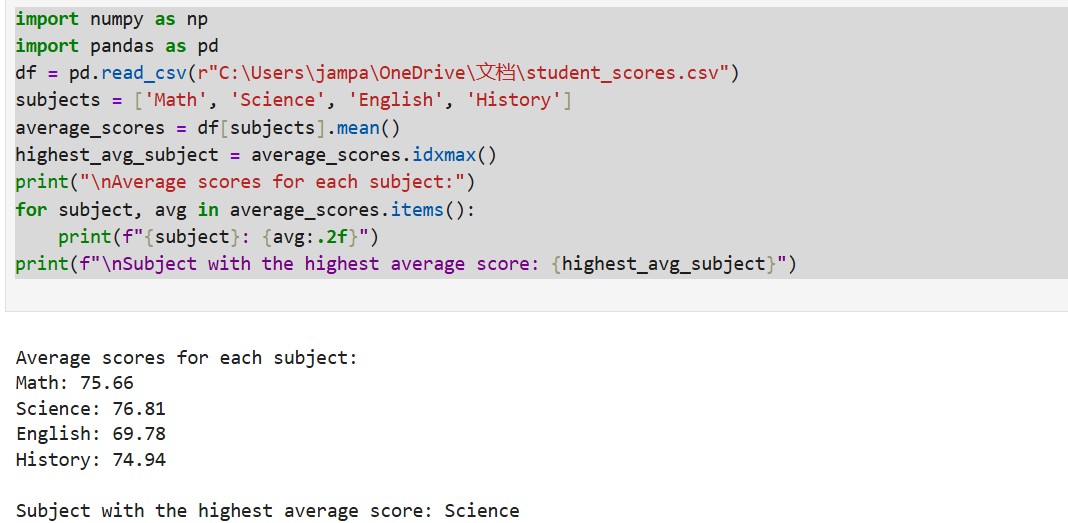
print("\nAverage scores for each subject:")

for subject, avg in average\_scores.items():

print(f"{subject}: {avg:.2f}")

print(f"\nSubject with the highest average score: {highest\_avg\_subject}")

output:



Dataset:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Math | Science | English | History |
| Student\_1 | 88 | 78 | 64 | 92 |
| Student\_2 | 57 | 70 | 88 | 68 |
| Student\_3 | 72 | 60 | 60 | 73 |
| Student\_4 | 85 | 89 | 73 | 52 |
| Student\_5 | 71 | 51 | 73 | 93 |
| Student\_6 | 79 | 87 | 51 | 70 |
| Student\_7 | 82 | 61 | 71 | 93 |
| Student\_8 | 74 | 98 | 76 | 91 |
| Student\_9 | 77 | 65 | 64 | 96 |
| Student\_10 | 100 | 93 | 52 | 86 |
| Student\_11 | 100 | 56 | 70 | 58 |
| Student\_12 | 88 | 67 | 53 | 74 |
| Student\_13 | 63 | 99 | 58 | 75 |
| Student\_14 | 51 | 69 | 77 | 96 |
| Student\_15 | 56 | 93 | 57 | 96 |
| Student\_16 | 84 | 63 | 66 | 85 |
| Student\_17 | 99 | 89 | 53 | 51 |
| Student\_18 | 55 | 91 | 53 | 78 |
| Student\_19 | 67 | 75 | 93 | 83 |
| Student\_20 | 59 | 85 | 63 | 80 |
| Student\_21 | 97 | 64 | 57 | 63 |
| Student\_22 | 72 | 89 | 70 | 65 |
| Student\_23 | 94 | 67 | 96 | 73 |
| Student\_24 | 75 | 74 | 94 | 90 |
| Student\_25 | 78 | 64 | 94 | 50 |
| Student\_26 | 74 | 56 | 58 | 73 |
| Student\_27 | 50 | 93 | 57 | 73 |
| Student\_28 | 60 | 100 | 66 | 57 |
| Student\_29 | 84 | 84 | 82 | 54 |
| Student\_30 | 91 | 88 | 90 | 77 |
| Student\_31 | 56 | 58 | 57 | 61 |
| Student\_32 | 83 | 82 | 97 | 72 |

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:

mport numpy as np

import pandas as pd

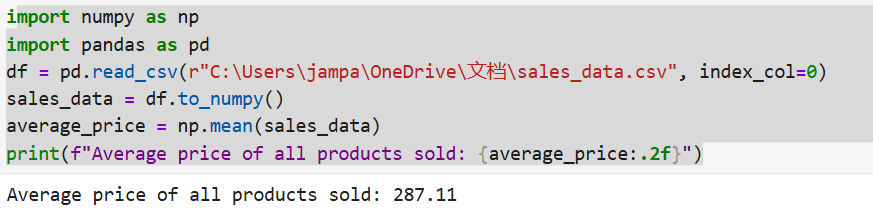
df = pd.read\_csv(r"C:\Users\jampa\OneDrive\文档\sales\_data.csv", index\_col=0)

sales\_data = df.to\_numpy()

average\_price = np.mean(sales\_data)

print(f"Average price of all products sold: {average\_price:.2f}")

output:



Dataset:

Sale\_1 Sale\_2 Sale\_3

Product\_A 428 397 202

Product\_B 353 411 155

Product\_C 139 366 133

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

import pandas as pd

try:

df = pd.read\_csv(r"C:\Users\jampa\OneDrive\文档\house\_data.csv")

house\_data = df.to\_numpy()

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

if houses\_more\_than\_4\_bedrooms.size > 0:

sale\_prices = houses\_more\_than\_4\_bedrooms[:, -1]

average\_sale\_price = np.mean(sale\_prices)

print(f"Average sale price of houses with more than 4 bedrooms: ${average\_sale\_price:,.2f}")

else:

print("No houses with more than 4 bedrooms found.")

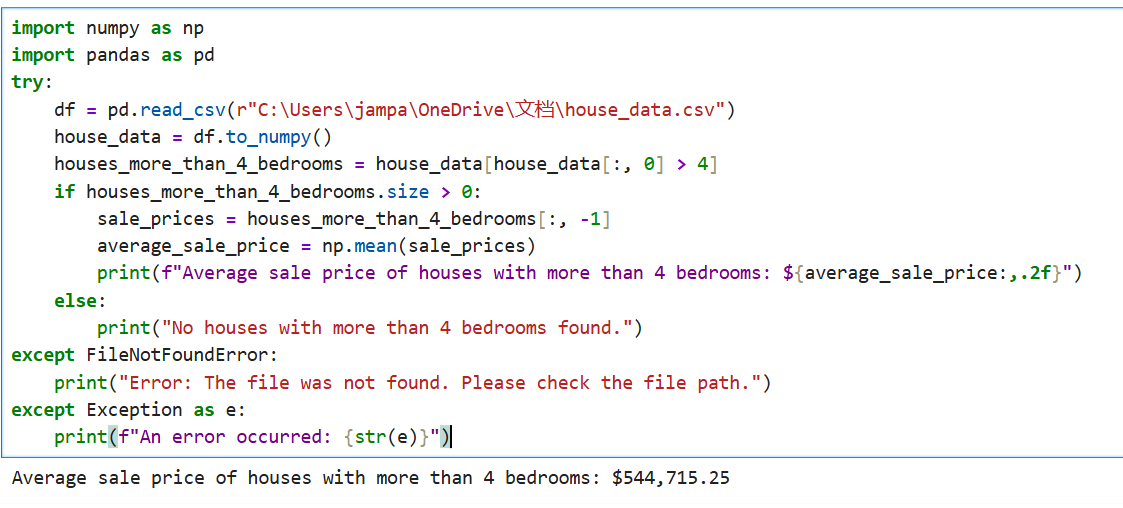
except FileNotFoundError:

print("Error: The file was not found. Please check the file path.")

except Exception as e:

print(f"An error occurred: {str(e)}")

output:



Dataset:

|  |  |  |  |
| --- | --- | --- | --- |
| Bedrooms | SquareFootage | SalePrice |  |
| 6 | 1704 | 950950 |  |
| 3 | 2822 | 828851 |  |
| 5 | 1892 | 889948 |  |
| 5 | 2256 | 127692 |  |
| 6 | 2719 | 210271 |  |
| 3 | 1191 | 464344 |  |
| 2 | 3788 | 330552 |  |
| 3 | 1899 | 669527 |  |
| 4 | 3671 | 230160 |  |
| 4 | 3098 | 989595 |  |
|  |  |  |  |

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 pandas as pd

import numpy as np

df = pd.read\_csv(r"C:\Users\jampa\OneDrive\文档\quarterly\_sales.csv")

sales\_data = df['Sales'].to\_numpy()

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:

A screenshot of a computer code

AI-generated content may be incorrect.

Dataset:

|  |  |  |
| --- | --- | --- |
| Quarter | Sales |  |
| Q1 | 25795 |  |
| Q2 | 10860 |  |
| Q3 | 86820 |  |
| Q4 | 64886 |  |
|  |  |  |
|  |  |  |

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 pandas as pd

import numpy as np

df = pd.read\_csv(r"C:\Users\jampa\OneDrive\文档\fuel\_efficiency\_data.csv")

fuel\_efficiency = df['FuelEfficiency'].to\_numpy()

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

efficiency\_model\_2 = df.loc[df['CarModel'] == 'Model\_2', 'FuelEfficiency'].values[0]

efficiency\_model\_7 = df.loc[df['CarModel'] == 'Model\_7', 'FuelEfficiency'].values[0]

percentage\_improvement = ((efficiency\_model\_7 - efficiency\_model\_2) / efficiency\_model\_2) \* 100

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

print(f"Fuel Efficiency of Model\_2: {efficiency\_model\_2:.2f} mpg")

print(f"Fuel Efficiency of Model\_7: {efficiency\_model\_7:.2f} mpg")

print(f"Percentage Improvement from Model\_2 to Model\_7: {percentage\_improvement:.2f}%")

output:

A screenshot of a computer program

AI-generated content may be incorrect.

Dataset:

|  |  |  |
| --- | --- | --- |
| CarModel | FuelEfficiency | |
| Model\_1 | 48.40069 |  |
| Model\_2 | 37.98048 |  |
| Model\_3 | 49.99469 |  |
| Model\_4 | 18.80269 |  |
| Model\_5 | 24.44225 |  |
| Model\_6 | 39.59364 |  |
| Model\_7 | 49.85823 |  |
| Model\_8 | 22.65388 |  |
| Model\_9 | 15.46178 |  |
| Model\_10 | 25.3592 |  |
|  |  |  |

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:

import numpy as np

import pandas as pd

df = pd.read\_csv(r"C:\Users\jampa\Downloads\customer\_purchase.csv")

prices = df['Price per Item'].to\_numpy()

quantities = df['Quantity'].to\_numpy()

items = df['Item'].to\_numpy()

subtotals = prices \* quantities

subtotal = np.sum(subtotals)

discount\_rate = 10

tax\_rate = 8

discount\_amount = subtotal \* (discount\_rate / 100)

after\_discount = subtotal - discount\_amount

tax\_amount = after\_discount \* (tax\_rate / 100)

total\_cost = after\_discount + tax\_amount

max\_index = np.argmax(subtotals)

top\_item = items[max\_index]

print("Subtotals per item:")

for item, value in zip(items, subtotals):

print(f"{item}: ${value:.2f}")

print(f"\nTotal before discount and tax: ${subtotal:.2f}")

print(f"Discount ({discount\_rate}%): -${discount\_amount:.2f}")

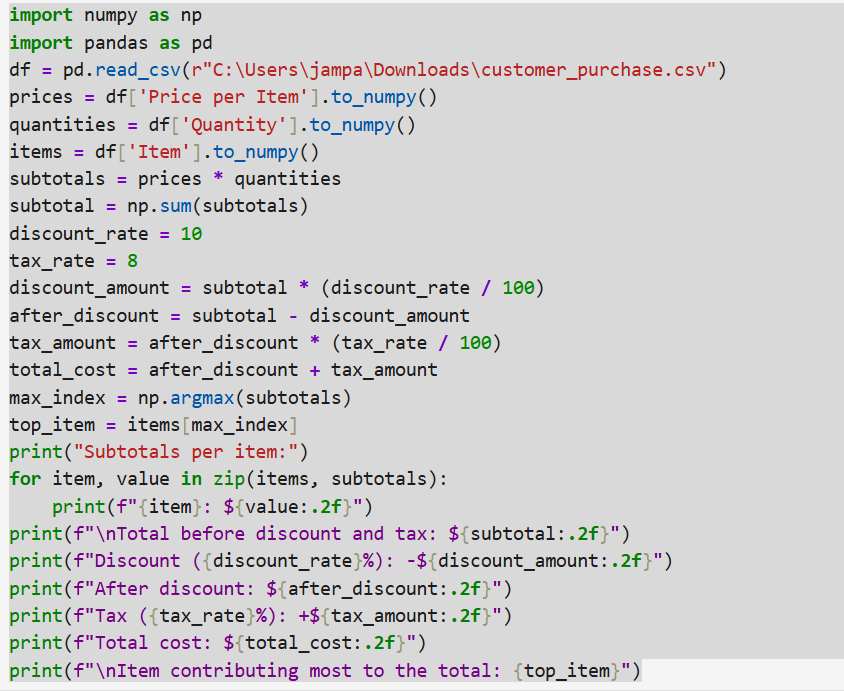
print(f"After discount: ${after\_discount:.2f}")

print(f"Tax ({tax\_rate}%): +${tax\_amount:.2f}")

print(f"Total cost: ${total\_cost:.2f}")

print(f"\nItem contributing most to the total: {top\_item}")

output:



A screenshot of a computer screen

AI-generated content may be incorrect.

Data set:

|  |  |  |  |
| --- | --- | --- | --- |
| Item | Price per Item | Quantity |  |
| Apples | 2.5 | 4 |  |
| Bread | 1.75 | 2 |  |
| Milk | 3 | 1 |  |
| Eggs | 2.25 | 3 |  |

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

df = pd.read\_csv(r"C:\Users\jampa\Downloads\customer\_orders.csv")

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

total\_orders\_per\_customer = df.groupby('Customer ID').size()

average\_quantity\_per\_product = df.groupby('Product Name')['Order Quantity'].mean()

earliest\_order\_date = df['Order Date'].min()

latest\_order\_date = df['Order Date'].max()

print("1. Total number of orders per customer:")

print(total\_orders\_per\_customer)

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

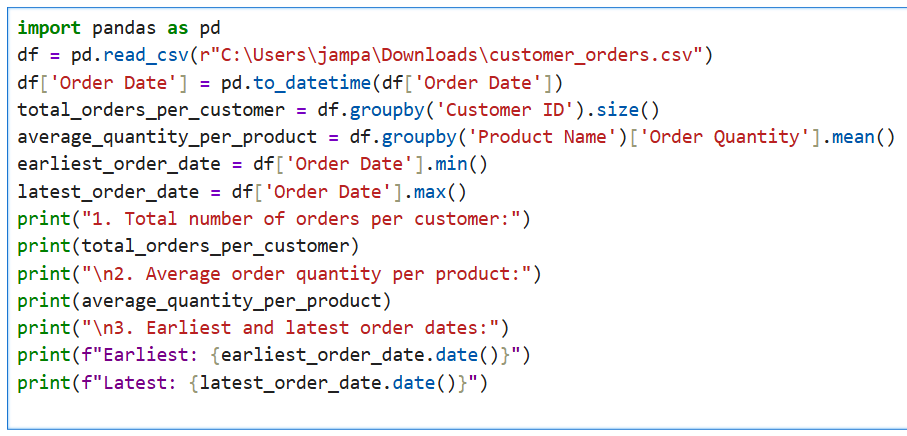
print(average\_quantity\_per\_product)

print("\n3. Earliest and latest order dates:")

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

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

output:



A screenshot of a computer

AI-generated content may be incorrect.

Dataset:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Customer ID | Order Date | Product Name | Order Quantity | |
| 101 | 1/10/2024 | Laptop | 1 |  |
| 102 | 1/12/2024 | Mouse | 2 |  |
| 103 | 2/1/2024 | Keyboard | 1 |  |
| 104 | 2/10/2024 | Laptop | 1 |  |
| 105 | 2/15/2024 | Mouse | 3 |  |
| 105 | 3/1/2024 | Monitor | 2 |  |
| 106 | 3/5/2024 | Laptop | 1 |  |
| 2 | 2 | 2 | 2 |  |
|  |  |  |  |  |
|  |  |  |  |  |

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

df = pd.read\_csv(r"C:\Users\jampa\Downloads\monthly\_sales\_data.csv")

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

recent\_sales = df[df['Order Date'] >= pd.Timestamp.today() - pd.Timedelta(days=30)]

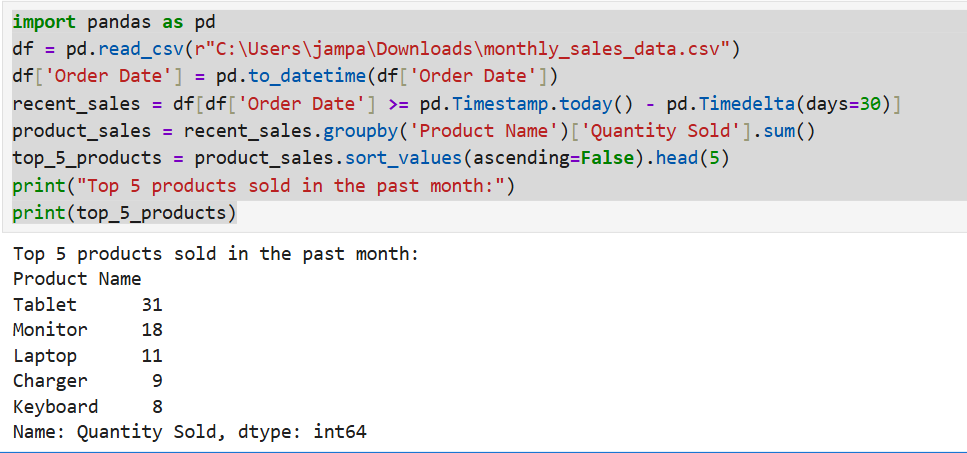
product\_sales = recent\_sales.groupby('Product Name')['Quantity Sold'].sum()

top\_5\_products = product\_sales.sort\_values(ascending=False).head(5)

print("Top 5 products sold in the past month:")

print(top\_5\_products)

output:



Dataset:

|  |  |  |
| --- | --- | --- |
| Product Name | Order Date | Quantity Sold |
| Keyboard | 3/30/2025 | 3 |
| Monitor | 4/5/2025 | 5 |
| Monitor | 3/26/2025 | 4 |
| Monitor | 3/21/2025 | 1 |
| Tablet | 4/17/2025 | 5 |
| USB Cable | 4/1/2025 | 2 |
| Keyboard | 4/15/2025 | 2 |
| Monitor | 4/24/2025 | 1 |
| USB Cable | 3/25/2025 | 3 |
| Mouse | 4/11/2025 | 2 |
| Tablet | 4/19/2025 | 1 |
| Tablet | 4/23/2025 | 1 |
| Tablet | 4/4/2025 | 2 |
| Charger | 3/22/2025 | 3 |
| Monitor | 4/26/2025 | 3 |
| Tablet | 4/14/2025 | 3 |
| Charger | 4/25/2025 | 3 |
| Tablet | 4/10/2025 | 5 |
| USB Cable | 3/29/2025 | 2 |
| Charger | 4/4/2025 | 2 |
| Tablet | 4/23/2025 | 4 |
| Keyboard | 3/26/2025 | 5 |
| Mouse | 3/30/2025 | 4 |
| Charger | 4/5/2025 | 3 |
| USB Cable | 4/29/2025 | 5 |
| Keyboard | 4/9/2025 | 5 |
| Smartphone | 3/25/2025 | 4 |
| Tablet | 3/26/2025 | 4 |
| USB Cable | 4/10/2025 | 1 |
| Keyboard | 4/20/2025 | 1 |
| Smartphone | 3/27/2025 | 5 |
| Charger | 3/20/2025 | 1 |
| Smartphone | 4/29/2025 | 2 |
| Monitor | 4/10/2025 | 1 |
| Laptop | 4/15/2025 | 1 |
| Laptop | 4/17/2025 | 5 |
| Laptop | 3/28/2025 | 5 |
| Tablet | 4/25/2025 | 1 |
| Smartphone | 3/24/2025 | 2 |
| Tablet | 4/26/2025 | 4 |
| Mouse | 3/26/2025 | 5 |
| Laptop | 4/20/2025 | 5 |
| Mouse | 4/25/2025 | 4 |
| USB Cable | 3/22/2025 | 4 |
| Tablet | 3/20/2025 | 2 |
| Monitor | 4/16/2025 | 5 |
| Monitor | 4/19/2025 | 3 |
| Smartphone | 3/22/2025 | 4 |
| Tablet | 4/7/2025 | 5 |
| Charger | 4/10/2025 | 1 |
|  |  |  |
|  |  |  |

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

df = pd.read\_csv(r"C:\Users\jampa\Downloads\property\_data.csv")

avg\_price\_per\_location = df.groupby('Location')['Listing Price'].mean()

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

print(avg\_price\_per\_location)

properties\_gt4\_bedrooms = df[df['Bedrooms'] > 4].shape[0]

print(f"\n2. Number of properties with more than four bedrooms: {properties\_gt4\_bedrooms}")

largest\_property = df.loc[df['Area (sqft)'].idxmax()]

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

print(largest\_property)

output:



A screenshot of a computer

AI-generated content may be incorrect.

Dataset:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Property ID | Location | Bedrooms | Area (sqft) | Listing Price |
| P1000 | Uptown | 1 | 4384 | 373030 |
| P1001 | Suburb | 2 | 3161 | 111734 |
| P1002 | Suburb | 5 | 3744 | 885619 |
| P1003 | Countryside | 2 | 755 | 122025 |
| P1004 | Suburb | 5 | 4989 | 971361 |
| P1005 | Countryside | 1 | 1239 | 415681 |
| P1006 | Suburb | 3 | 4624 | 805268 |
| P1007 | Suburb | 3 | 1147 | 538995 |
| P1008 | Uptown | 1 | 4990 | 370373 |
| P1009 | Suburb | 6 | 3047 | 918661 |
| P1010 | Downtown | 3 | 1770 | 632275 |
| P1011 | Countryside | 2 | 4038 | 102225 |
| P1012 | Downtown | 3 | 4610 | 689011 |
| P1013 | Suburb | 2 | 2565 | 686207 |
| P1014 | Uptown | 5 | 4454 | 802110 |
| P1015 | Uptown | 4 | 4593 | 424878 |
| P1016 | Downtown | 4 | 3906 | 578262 |
| P1017 | Downtown | 4 | 2946 | 747357 |
| P1018 | Downtown | 6 | 3877 | 570701 |
| P1019 | Beachside | 6 | 4166 | 595284 |
| P1020 | Beachside | 6 | 4484 | 384225 |
| P1021 | Beachside | 4 | 3940 | 164284 |
| P1022 | Beachside | 4 | 1129 | 905991 |
| P1023 | Beachside | 6 | 3713 | 959953 |
| P1024 | Countryside | 4 | 916 | 758676 |
| P1025 | Countryside | 5 | 801 | 718583 |
| P1026 | Uptown | 5 | 2899 | 674186 |
| P1027 | Downtown | 1 | 1684 | 392289 |
| P1028 | Beachside | 3 | 1040 | 234617 |
| P1029 | Uptown | 2 | 2392 | 289949 |
| P1030 | Beachside | 1 | 3913 | 720755 |
| P1031 | Downtown | 4 | 3245 | 392006 |
| P1032 | Downtown | 1 | 4877 | 398314 |
| P1033 | Beachside | 4 | 3286 | 410913 |
| P1034 | Uptown | 4 | 1828 | 709252 |
| P1035 | Suburb | 1 | 1232 | 943908 |
| P1036 | Downtown | 2 | 1275 | 606295 |
| P1037 | Uptown | 2 | 3131 | 393130 |
| P1038 | Beachside | 4 | 4270 | 993872 |
| P1039 | Downtown | 3 | 2781 | 771738 |
| P1040 | Downtown | 5 | 843 | 786022 |
| P1041 | Downtown | 6 | 637 | 561131 |
| P1042 | Beachside | 4 | 2630 | 168340 |
| P1043 | Beachside | 5 | 2794 | 728638 |
| P1044 | Suburb | 3 | 3188 | 776262 |
| P1045 | Beachside | 3 | 1955 | 469456 |
| P1046 | Downtown | 1 | 3912 | 408043 |
| P1047 | Uptown | 3 | 2167 | 102899 |
| P1048 | Countryside | 5 | 3106 | 608864 |
| P1049 | Uptown | 6 | 777 | 153475 |
|  |  |  |  |  |

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 pandas as pd

import matplotlib.pyplot as plt

df = pd.read\_csv(r"C:\Users\jampa\Downloads\monthly\_sales\_data.csv")

print("Columns in the DataFrame:", df.columns)

print(df.head())

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

df['Month'] = df['Order Date'].dt.month\_name()

monthly\_sales = df.groupby('Month')['Quantity Sold'].sum().reset\_index()

# 1. Line Plot of Monthly Sales

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

plt.plot(monthly\_sales['Month'], monthly\_sales['Quantity Sold'], marker='o', color='b', linestyle='-', linewidth=2, markersize=6)

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

plt.xlabel('Month')

plt.ylabel('Quantity Sold')

plt.grid(True)

plt.xticks(rotation=45)

plt.tight\_layout()

plt.show()

# 2. Bar Plot of Monthly Sales

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

plt.bar(monthly\_sales['Month'], monthly\_sales['Quantity Sold'], color='c')

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

plt.xlabel('Month')

plt.ylabel('Quantity Sold')

plt.xticks(rotation=45)

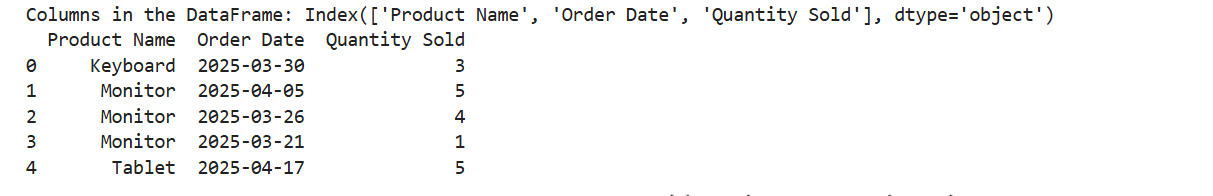
plt.tight\_layout()

plt.show()

output:

A white background with text

AI-generated content may be incorrect.



A graph with a line

AI-generated content may be incorrect.

A screenshot of a graph

AI-generated content may be incorrect.

Dataset:

|  |  |  |
| --- | --- | --- |
| **Month** | **Sales** |  |
| January | 35000 |  |
| February | 42000 |  |
| March | 39000 |  |
| April | 45000 |  |
| May | 47000 |  |
| June | 48000 |  |
| July | 46000 |  |
| August | 50000 |  |
| September | 44000 |  |
| October | 41000 |  |
| November | 43000 |  |
| December | 46000 |  |

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.

Code:

import pandas as pd

import matplotlib.pyplot as plt

df = pd.read\_csv(r"C:\Users\jampa\Downloads\monthly\_sales\_data (1).csv")

plt.figure(figsize=(15, 12))

# 1.Line Plot

plt.subplot(3, 1, 1)

plt.plot(df['Month'], df['Sales'], marker='o', linestyle='-', color='blue')

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

plt.xlabel('Month')

plt.ylabel('Sales')

plt.xticks(rotation=45)

plt.grid(True)

# 2.Scatter Plot

plt.subplot(3, 1, 2)

plt.scatter(df['Month'], df['Sales'], color='red')

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

plt.xlabel('Month')

plt.ylabel('Sales')

plt.xticks(rotation=45)

plt.grid(True)

# 3.Bar Plot

plt.subplot(3, 1, 3)

plt.bar(df['Month'], df['Sales'], color='green')

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

plt.xlabel('Month')

plt.ylabel('Sales')

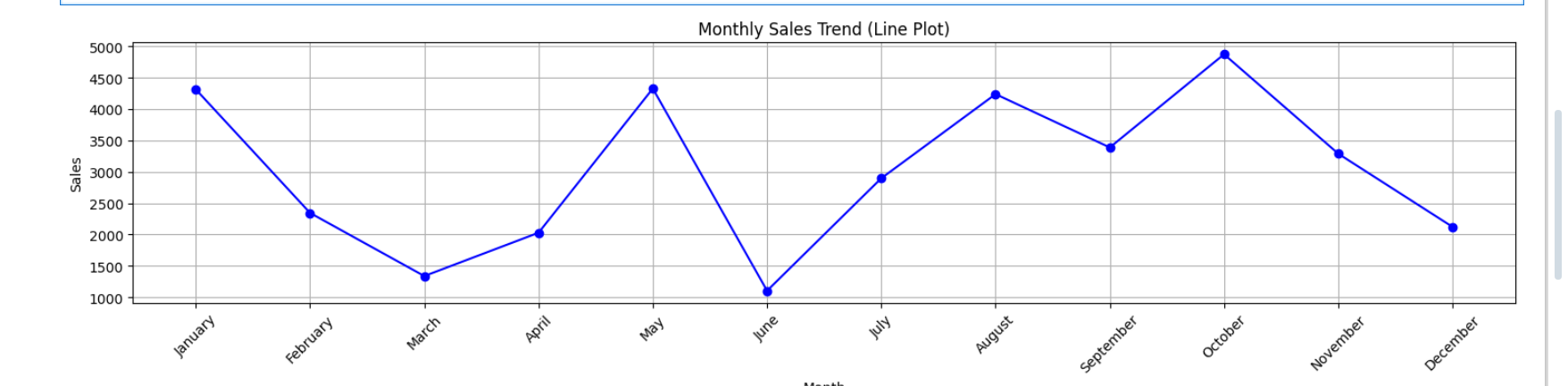
plt.xticks(rotation=45)

plt.tight\_layout()

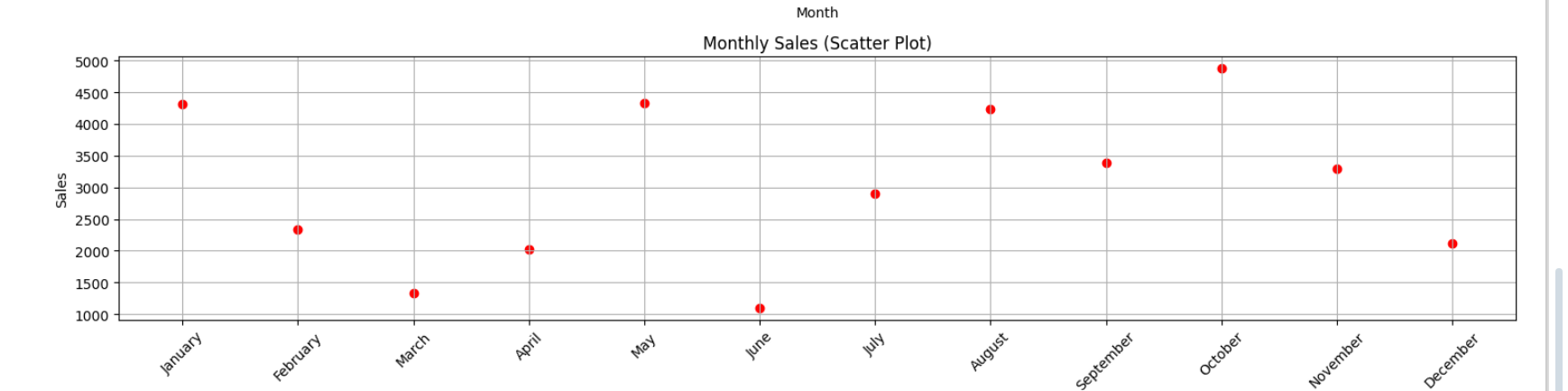
plt.show()

output:

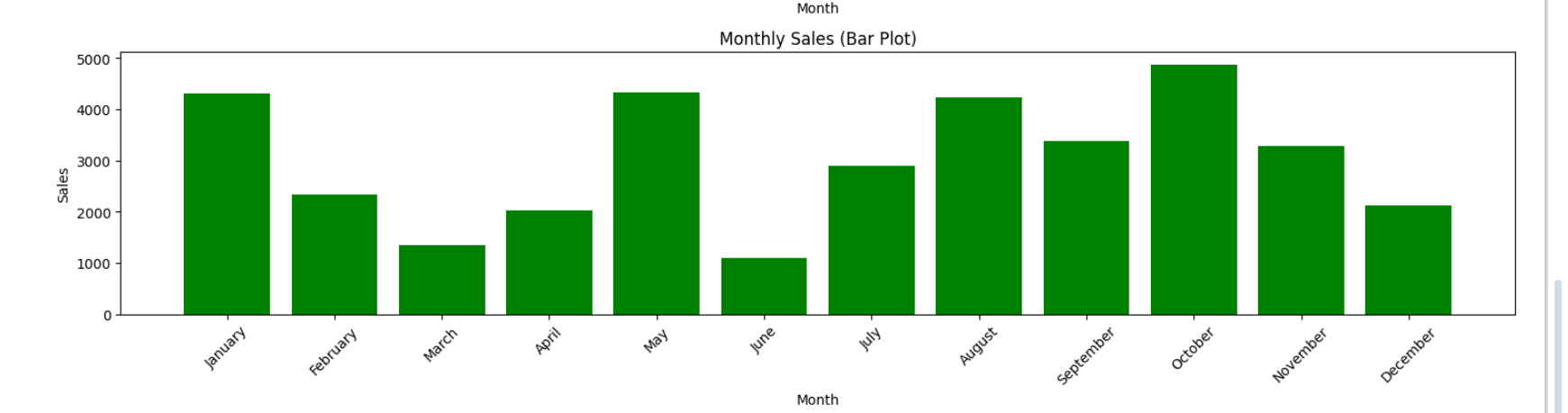
1.line plot



2.scatter plot



3.bar plot



Dataset:

|  |  |
| --- | --- |
| Month | Sales |
| January | 4313 |
| February | 2344 |
| March | 1338 |
| April | 2029 |
| May | 4327 |
| June | 1104 |
| July | 2897 |
| August | 4238 |
| September | 3388 |
| October | 4873 |
| November | 3288 |
| December | 2122 |