Name: Palak Khandelia
Email: palakkhandelia65@gmail.com
Assignment Title: Data Toolkit – Theoretical Questions
Subject: Python for Data Science

Q1: What is NumPy, and why is it widely used in Python?

Answer:
NumPy is a powerful Python library used for numerical computations. It provides support for multi-dimensional arrays and includes functions for mathematical, logical, statistical, and Fourier operations. It is widely used because it is fast, efficient, and integrates easily with other libraries like Pandas and Matplotlib.

Q2: How does broadcasting work in NumPy?

Answer:
Broadcasting allows NumPy to perform arithmetic operations on arrays of different shapes. It automatically stretches the smaller array to match the larger array without copying data, making operations more efficient.

Q3: What is a Pandas DataFrame?

Answer:
A Pandas DataFrame is a 2D labeled data structure with columns of different data types. It is similar to a table in a database or an Excel spreadsheet and is widely used for data analysis and manipulation in Python.

Q4: Explain the use of the groupby() method in Pandas.

Answer:
The `groupby()` method is used to split data into groups based on values in one or more columns. It allows aggregation and transformation of data within each group, making it easier to analyze grouped data.

Q5: Why is Seaborn preferred for statistical visualizations?

Answer:
Seaborn is preferred because it provides a high-level interface for drawing attractive and informative statistical graphics. It is built on top of Matplotlib and supports complex plots with simple code.

Q6: What are the differences between NumPy arrays and Python lists?

Answer:

- NumPy arrays are faster and more memory efficient.
- They support element-wise operations and broadcasting.
- NumPy arrays require all elements to be of the same type, unlike lists.

Q7: What is a heatmap, and when should it be used?

Answer:
A heatmap is a graphical representation of data where values are represented by color. It is useful for showing correlation matrices, or the magnitude of data across two dimensions.

Q8: What does the term "vectorized operation" mean in NumPy?

Answer:
Vectorized operations allow operations on entire arrays without writing loops. This results in faster and more readable code, leveraging internal optimizations.

Q9: How does Matplotlib differ from Plotly?

Answer:
Matplotlib is used for static plots, while Plotly supports interactive plots with features like zoom and hover. Plotly is better suited for dashboards and web-based applications.

Q10: What is the significance of hierarchical indexing in Pandas?

Answer:
Hierarchical indexing allows multiple levels of indexing on rows or columns. It enables the representation of higher-dimensional data and flexible data selection and grouping.

Q11: What is the role of Seaborn's pairplot() function?

Answer:
The `pairplot()` function creates a grid of scatter plots to visualize pairwise relationships in a dataset. It is useful for quickly exploring correlations and distributions between variables.

Q12: What is the purpose of the describe() function in Pandas?

Answer:
The `describe()` function provides summary statistics of numeric columns, such as count, mean, std deviation, min, max, and percentiles. It is used for quick data overview.

Q13: Why is handling missing data important in Pandas?

Answer:
Missing data can affect analysis accuracy. Pandas provides methods like `isnull()`, `dropna()`, and `fillna()` to handle missing values and maintain data quality.

Q14: What are the benefits of using Plotly for data visualization?

Answer:
Plotly creates interactive plots that can be embedded in web applications. It supports zooming, tooltips, 3D plots, and is easy to integrate with Dash for dashboards.

Q15: How does NumPy handle multidimensional arrays?

Answer:
NumPy uses the `ndarray` object to create and manipulate arrays of any dimension. It provides methods like `reshape()`, `transpose()`, and slicing for handling multi-dimensional data.

Q16: What is the role of Bokeh in data visualization?

Answer:
Bokeh is used for creating interactive visualizations for the web. It supports real-time streaming, linked plots, and dashboards, making it ideal for dynamic data applications.

Q17: Explain the difference between apply() and map() in Pandas.

Answer:

- `map()` works only on Series and is used for element-wise operations.
- `apply()` can be used on both Series and DataFrames to apply a function along an axis.

Q18: What are some advanced features of NumPy?

Answer:

- Broadcasting
- Vectorized operations
- Linear algebra
- Random number generation

- Fourier transforms
- Memory mapping

Q19: How does Pandas simplify time series analysis?

Answer:
Pandas provides powerful tools for time series like date parsing, resampling, shifting, and rolling window functions. It supports date-based indexing and frequency conversion.

Q20: What is the role of a pivot table in Pandas?

Answer:
A pivot table summarizes data by aggregating values based on categories. It allows for multi-dimensional grouping and analysis using the `pivot_table()` function.

Q21: Why is NumPy's array slicing faster than Python's list slicing?

Answer:
NumPy arrays use continuous memory blocks and are implemented in C, which allows faster slicing and better performance compared to regular Python lists.

Q22: What are some common use cases for Seaborn?

Answer:

- Visualizing distributions and relationships
- Drawing heatmaps and correlation plots
- Time series plots
- Statistical plots like boxplots and violin plots

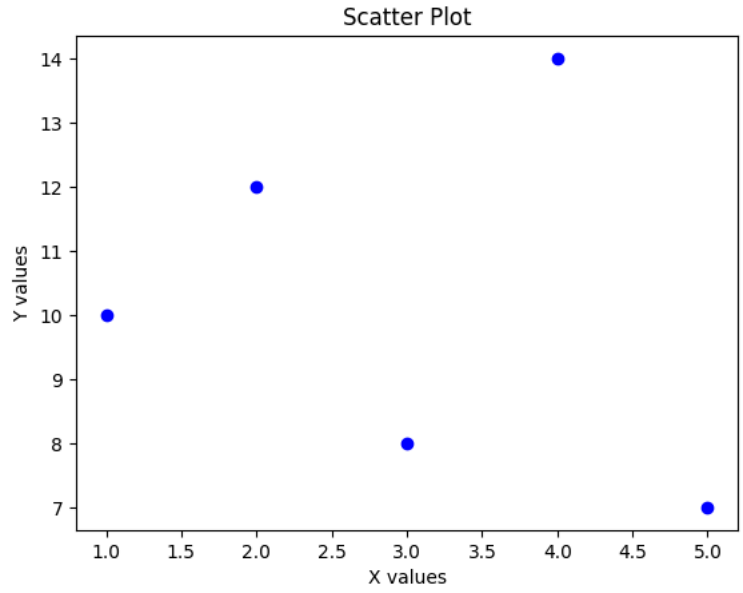
```
## Practical Questions


# Q1: How do you create a 2D NumPy array and calculate the sum of each row?

import numpy as np

arr = np.array([[1, 2, 3], [4, 5, 6], [7, 8, 9]])
row_sum = np.sum(arr, axis=1)
print("2D Array:\n", arr)
print("Sum of each row:", row_sum)
```

```
2D Array:
 [[1 2 3]
 [4 5 6]
 [7 8 9]]
Sum of each row: [ 6 15 24]
```

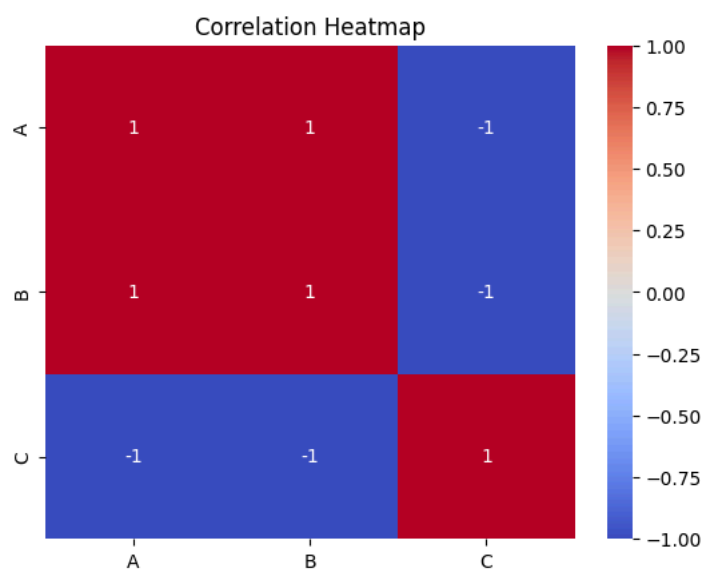
```
# Q2: Write a Pandas script to find the mean of a specific column in a DataFrame.

import pandas as pd

data = {'Math': [80, 75, 90, 85], 'Science': [88, 72, 95, 78]}
df = pd.DataFrame(data)
mean_math = df['Math'].mean()
print("Mean of Math column:", mean_math)
```

```
Mean of Math column: 82.5
```

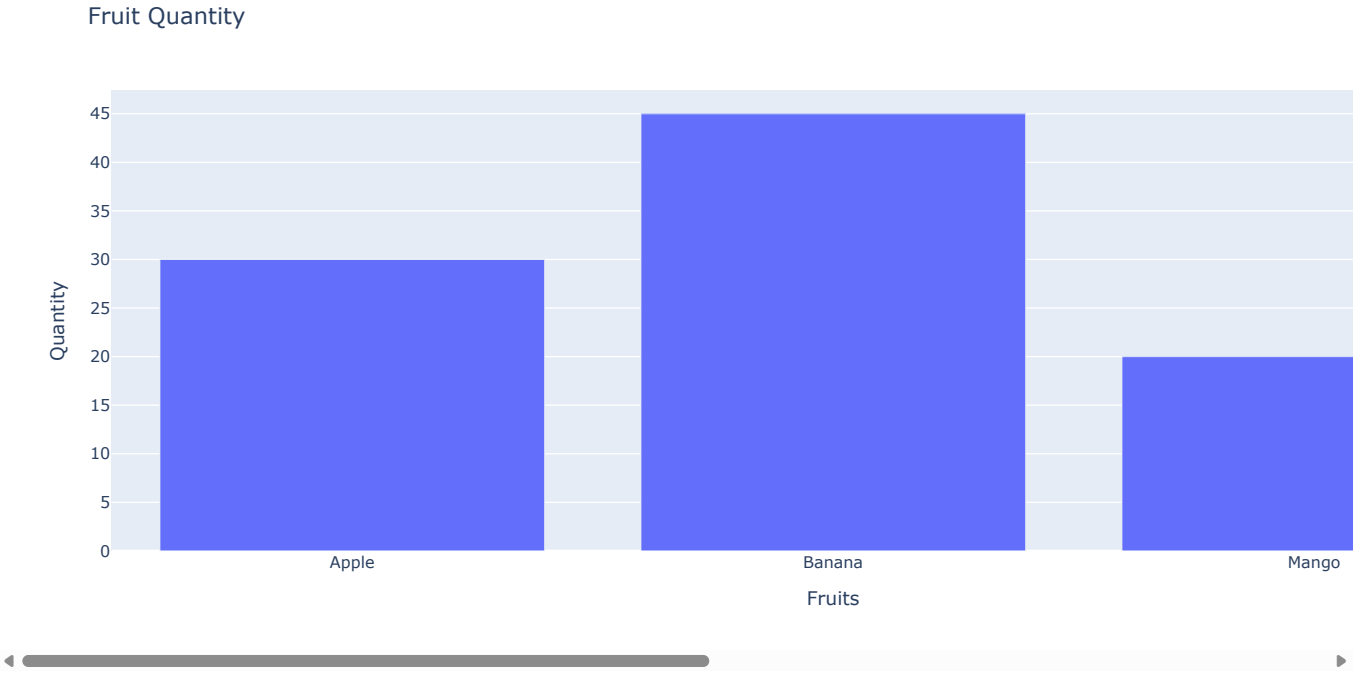
```
# Q3: Create a scatter plot using Matplotlib.

import matplotlib.pyplot as plt

x = [1, 2, 3, 4, 5]
y = [10, 12, 8, 14, 7]

plt.scatter(x, y, color='blue')
plt.title("Scatter Plot")
plt.xlabel("X values")
plt.ylabel("Y values")
plt.show()
```




```
# Q4: How do you calculate the correlation matrix using Seaborn and visualize it with a heatmap?

import seaborn as sns
import matplotlib.pyplot as plt

df = pd.DataFrame({
    'A': [1, 2, 3, 4],
    'B': [2, 3, 4, 5],
    'C': [5, 4, 3, 2]
})

corr_matrix = df.corr()
sns.heatmap(corr_matrix, annot=True, cmap='coolwarm')
plt.title("Correlation Heatmap")
plt.show()
```




```
# Q5: Generate a bar plot using Plotly.

import plotly.express as px

data = {'Fruits': ['Apple', 'Banana', 'Mango'], 'Quantity': [30, 45, 20]}
df = pd.DataFrame(data)

fig = px.bar(df, x='Fruits', y='Quantity', title="Fruit Quantity")
fig.show()
```



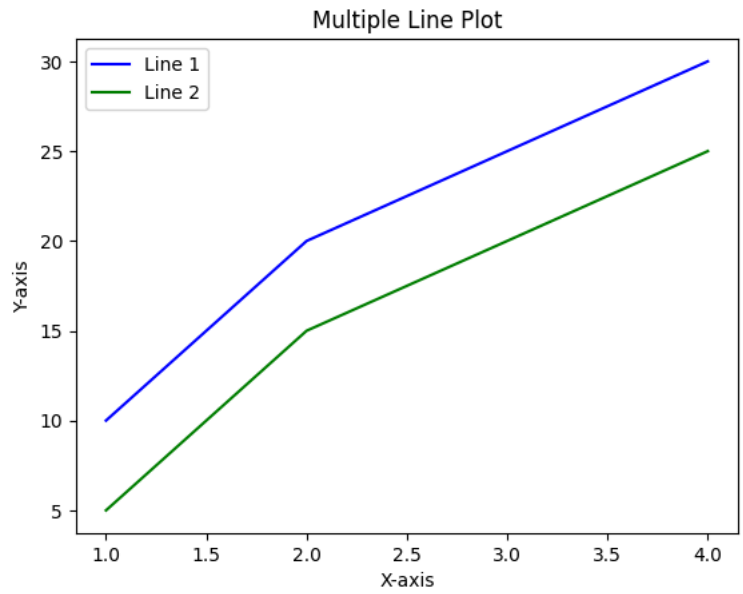

```
# Q6: Create a DataFrame and add a new column based on an existing column.

df = pd.DataFrame({'Marks': [50, 60, 70, 80]})
df['Grade'] = ['Fail' if m < 60 else 'Pass' for m in df['Marks']]
print(df)
```

```
   Marks Grade
0     50  Fail
1     60  Pass
2     70  Pass
3     80  Pass
```

```
# Q7: Write a program to perform element-wise multiplication of two NumPy arrays.

arr1 = np.array([1, 2, 3])
arr2 = np.array([4, 5, 6])

result = arr1 * arr2
print("Element-wise multiplication:", result)
```

```
Element-wise multiplication: [ 4 10 18]
```

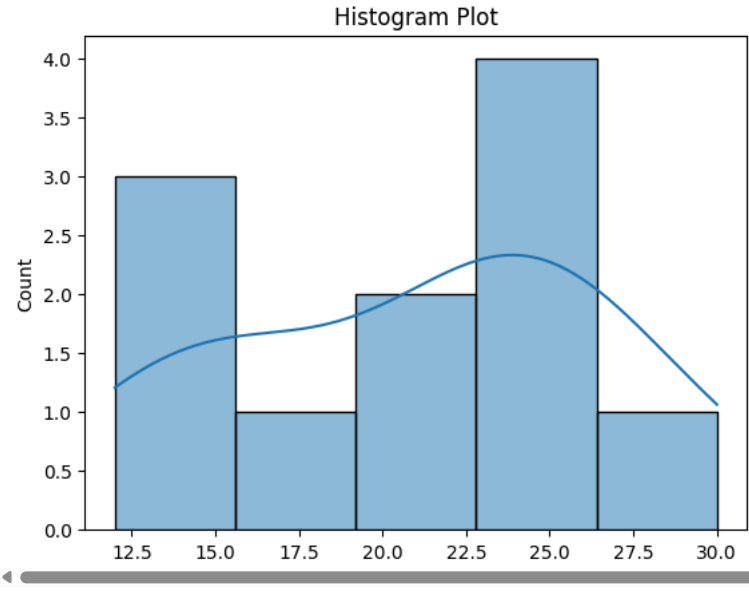
```
# Q8: Create a line plot with multiple lines using Matplotlib.

x = [1, 2, 3, 4]
y1 = [10, 20, 25, 30]
y2 = [5, 15, 20, 25]

plt.plot(x, y1, label='Line 1', color='blue')
plt.plot(x, y2, label='Line 2', color='green')
plt.xlabel("X-axis")
plt.ylabel("Y-axis")
plt.title("Multiple Line Plot")
plt.legend()
plt.show()
```




```
# Q9: Generate a Pandas DataFrame and filter rows where a column value is greater than a threshold.

df = pd.DataFrame({'Score': [45, 78, 50, 89, 60]})
filtered_df = df[df['Score'] > 60]
print("Filtered DataFrame:\n", filtered_df)
```

```
Filtered DataFrame:
    Score
1     78
3     89
```

```
# Q10: Create a histogram using Seaborn to visualize a distribution.

data = [12, 14, 15, 17, 21, 22, 24, 25, 25, 26, 30]

sns.histplot(data, bins=5, kde=True)
plt.title("Histogram Plot")
plt.show()
```




```
# Q11: Perform matrix multiplication using NumPy.

a = np.array([[1, 2], [3, 4]])
b = np.array([[5, 6], [7, 8]])

result = np.dot(a, b)
print("Matrix Multiplication Result:\n", result)
```

```
Matrix Multiplication Result:
 [[19 22]
 [43 50]]
```

```
# Q12: Use Pandas to load a CSV file and display its first 5 rows.
```

```
# Q12: Use Pandas to load a CSV file and display its first 5 rows
# (No upload needed – CSV will be created automatically)

import pandas as pd

# Step 1: Create and save a sample CSV file
data = {
    'Name': ['Palak', 'Riya', 'Neha', 'Tanya', 'Meera'],
    'Subject': ['Math', 'Science', 'English', 'Math', 'History'],
    'Marks': [85, 92, 88, 79, 95]
}

df = pd.DataFrame(data)
df.to_csv('sample.csv', index=False)
print("✅ Sample CSV file 'sample.csv' created.\n")

# Step 2: Read the CSV file and display first 5 rows
df_loaded = pd.read_csv('sample.csv')
print("First 5 rows of the CSV file:")
print(df_loaded.head())
```

```
✅ Sample CSV file 'sample.csv' created.

First 5 rows of the CSV file:
    Name  Subject  Marks
0  Palak     Math     85
1   Riya  Science     92
2   Neha  English     88
3  Tanya     Math     79
4  Meera  History     95
```

```
# Q13: Create a 3D scatter plot using Plotly.

import plotly.graph_objects as go

fig = go.Figure(data=[go.Scatter3d(
    x=[1, 2, 3, 4],
    y=[10, 20, 25, 30],
    z=[5, 15, 10, 20],
    mode='markers',
    marker=dict(size=5, color='blue')
)])
fig.update_layout(title="3D Scatter Plot")
fig.show()
```