**Module 7 - Advanced Python Tools**

**Assignment 1 - Working with NumPy**

Answer

import numpy as np

Problem 1: 1D Array Operations

print("--- Problem 1: 1D Array Operations ---")

a.Create a 1D NumPy array with integers from 1 to 20

arr1d = np.arange(1, 21)

print(f"1D Array:\n{arr1d}\n")

b. Calculate the sum, mean, median, and standard deviation

arr\_sum = np.sum(arr1d)

arr\_mean = np.mean(arr1d)

arr\_median = np.median(arr1d)

arr\_std\_dev = np.std(arr1d)

print(f"Sum of elements: {arr\_sum}")

print(f"Mean of elements: {arr\_mean}")

print(f"Median of elements: {arr\_median}")

print(f"Standard deviation of elements: {arr\_std\_dev:.2f}\n")

c. Find the indices of elements greater than 10

indices\_greater\_than\_10 = np.where(arr1d > 10)

print(f"Indices of elements greater than 10:\n{indices\_greater\_than\_10}\n") print(f"Elements greater than 10: {arr1d[indices\_greater\_than\_10]}\n")

2: 2D Array Operations ---

print("--- Problem 2: 2D Array Operations ---")

1. Create a 2D NumPy array of shape 4x4

arr2d = np.arange(1, 17).reshape(4, 4)

print(f"2D Array (4x4):\n{arr2d}\n")

1. Find the transpose of the array

arr2d\_transposed = arr2d.T

print(f"Transpose of the array:\n{arr2d\_transposed}\n")

1. Calculate the row-wise and column-wise sums

row\_sums = np.sum(arr2d, axis=1)

col\_sums = np.sum(arr2d, axis=0)

print(f"Row-wise sums: {row\_sums}")

print(f"Column-wise sums: {col\_sums}\n")

Problem 3: Array Arithmetic ---

print("--- Problem 3: Array Arithmetic ---")

1. Create two 3x3 arrays filled with random integers

arr\_A = np.random.randint(1, 21, size=(3, 3))

arr\_B = np.random.randint(1, 21, size=(3, 3))

print(f"Array A:\n{arr\_A}\n")

print(f"Array B:\n{arr\_B}\n")

1. Perform element-wise addition, subtraction, and multiplication

print(f"Element-wise Addition (A + B):\n{arr\_A + arr\_B}\n")

print(f"Element-wise Subtraction (A - B):\n{arr\_A - arr\_B}\n")

print(f"Element-wise Multiplication (A \* B):\n{arr\_A \* arr\_B}\n")

1. Compute the dot product

dot\_product = arr\_A @ arr\_B

print(f"Dot product of A and B:\n{dot\_product}\n")

Problem 4: Reshaping and Slicing ---

print("--- Problem 4: Reshaping and Slicing ---")

1. Create a 1D array of size 12 and reshape it

arr1d\_size12 = np.arange(1, 13)

reshaped\_arr = arr1d\_size12.reshape(3, 4)

print(f"Reshaped 3x4 array:\n{reshaped\_arr}\n")

1. Slice the first two rows and last two columns

sliced\_arr = reshaped\_arr[:2, 2:]

print(f"Sliced array (first 2 rows, last 2 columns):\n{sliced\_arr}")

**Assignment 2 - Working with Pandas**

import pandas as pd

Provided data for the DataFrame

data = {

'Name': ['Alice', 'Bob', 'Charlie', 'Diana', 'Eve'],

'Age': [24, 27, 22, 32, 29],

'Department': ['HR', 'Finance', 'IT', 'Marketing', 'HR'],

'Salary': [45000, 54000, 50000, 62000, 47000] }

Problem 1: Create a DataFrame with the following data

Create a DataFrame

df = pd.DataFrame(data)

a.Print the first five rows of the DataFrame.

print("a. First 5 rows of the DataFrame:")

print(df.head()) print("\n" + "="\*30 + "\n")

b. Get the summary statistics of the 'Age' and 'Salary' columns.

print("b. Summary statistics of 'Age' and 'Salary':")

print(df[['Age', 'Salary']].describe())

print("\n" + "="\*30 + "\n")

1. Calculate the average salary of employees in the 'HR' department.

hr\_salary\_avg = df[df['Department'] == 'HR']['Salary'].mean()

print(f"c. Average salary of employees in the 'HR' department: ${hr\_salary\_avg:,.2f}") print("\n" + "="\*30 + "\n")

2. Add a new column, 'Bonus', which is 10% of the salary.

df['Bonus'] = df['Salary'] \* 0.10

print("DataFrame with new 'Bonus' column:")

print(df) print("\n" + "="\*30 + "\n")

3. Filter the DataFrame to show employees aged between 25 and 30.

filtered\_df = df[(df['Age'] >= 25) & (df['Age'] <= 30)]

print("Employees aged between 25 and 30:")

print(filtered\_df) print("\n" + "="\*30 + "\n")

4. Group the data by 'Department' and calculate the average salary for each department.

department\_salary\_avg = df.groupby('Department')['Salary'].mean()

print("Average salary for each department:")

print(department\_salary\_avg)

print("\n" + "="\*30 + "\n")

5. Sort the DataFrame by 'Salary' in ascending order and save the result to a new CSV file.

sorted\_df = df.sort\_values(by='Salary', ascending=True)

output\_file = 'sorted\_employees.csv'

sorted\_df.to\_csv(output\_file, index=False)

print(f"DataFrame sorted by Salary (ascending):\n{sorted\_df}\n")

print(f"Sorted data has been saved to '{output\_file}'.")

**Assignment 3 - Working with Matplotlib**

import matplotlib.pyplot as plt

import numpy as np

**1. Create a simple line plot for the following data:**

**x = [1, 2, 3, 4, 5]**

**y = [10, 15, 25, 30, 50]**

1. **Plot the data.**

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

plt.plot(x, y, marker='o', linestyle='-')

1. **Customize the plot by adding a title, axis labels, and a grid.**

plt.title('Line Plot of Sample Data')

plt.xlabel('X-axis')

plt.ylabel('Y-axis')

plt.grid(True)

plt.show()

**2.Create a bar graph to represent the marks scored by students in a subject:**

**students = ['John', 'Jane', 'Alice', 'Bob']**

**marks = [75, 85, 60, 90]**

1. **Plot the data as a bar graph.**

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

1. **Customize the colors and add a title.**

colors = ['blue', 'green', 'red', 'purple']

plt.bar(students, marks, color=colors)

plt.title('Student Marks in a Subject')

plt.xlabel('Students')

plt.ylabel('Marks')

plt.show()

**3. Create a pie chart to represent the percentage distribution of a company’s revenue from different regions:**

**regions = ['North America', 'Europe', 'Asia', 'Others']**

**revenue = [45, 25, 20, 10]**

**a. Create a pie chart with the region names as labels.**

**b. Highlight the region with the highest revenue.**

explode\_idx = revenue.index(max(revenue))

explode = [0] \* len(regions)

explode[explode\_idx] = 0.1

Separate the highest value slice

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

plt.pie(revenue, labels=regions, autopct='%1.1f%%', startangle=90, explode=explode, shadow=True)

plt.title('Company Revenue Distribution by Region')

plt.show()

**4. Generate a histogram to show the frequency distribution of randomly generated integers between 1 and 100 (sample size = 1000).**

random\_data = np.random.randint(1, 101, size=1000)

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

plt.hist(random\_data, bins=20, edgecolor='black', alpha=0.7)

plt.title('Frequency Distribution of Random Integers')

plt.xlabel('Value')

plt.ylabel('Frequency')

plt.grid(axis='y', alpha=0.75)

plt.show()