**WEEK-4**

**NumPy Basics and Operations**

**DESCRIPTION**

* This unit provides a comprehensive introduction to **NumPy**, one of the most essential libraries in Python for numerical computing. NumPy offers efficient array operations and forms the foundation for libraries like Pandas, SciPy, and scikit-learn.
* Students will explore the creation of **NumPy arrays**, understand their attributes such as **shape, size, and dimension**, and learn the importance of **vectorization** and broadcasting over traditional looping methods. The unit also covers **array indexing, slicing, reshaping, and flattening**.
* Advanced mathematical operations like **dot product, matrix multiplication, transpose, and inversion** are explored, helping students grasp how NumPy simplifies linear algebra tasks. By the end of this unit, students will be proficient in performing a wide variety of data manipulation tasks using NumPy arrays efficiently and concisely.

**Write a Python program that demonstrates a wide range of NumPy operations including array creation, reshaping, indexing, slicing, mathematical operations, aggregation, concatenation, splitting, and working with structured arrays.**

**Description:**

This program covers a wide range of NumPy operations as follows:

* **Array creation** using array, arange, ones, zeros, and eye.
* **Reshaping** arrays to new dimensions using .reshape().
* **Indexing and slicing** to access individual and groups of elements.
* **Mathematical operations** (e.g., addition, multiplication, square root).
* **Aggregations** like sum, mean, and max for statistical insights.
* **Concatenation and splitting** to combine and divide arrays.
* **Structured arrays**, which allow storage of mixed-type records in tabular format (like a lightweight DataFrame).

**Program:**

import numpy as np

# 1. Array Creation

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

arr2 = np.arange(6, 11)

arr3 = np.ones((2, 3))

arr4 = np.zeros((2, 3))

arr5 = np.eye(3)

print("Array 1:", arr1)

print("Array 2:", arr2)

print("Ones:\n", arr3)

print("Zeros:\n", arr4)

print("Identity Matrix:\n", arr5)

# 2. Reshaping

reshaped = arr2.reshape(1, 5)

print("Reshaped Array (1x5):", reshaped)

# 3. Indexing and Slicing

print("First element of arr1:", arr1[0])

print("Last 3 elements of arr2:", arr2[-3:])

# 4. Mathematical Operations

print("arr1 + 10:", arr1 + 10)

print("arr1 \* 2:", arr1 \* 2)

print("Square root of arr1:", np.sqrt(arr1))

# 5. Aggregation

print("Sum of arr1:", np.sum(arr1))

print("Mean of arr1:", np.mean(arr1))

print("Max of arr1:", np.max(arr1))

# 6. Concatenation and Splitting

arr6 = np.array([[1, 2], [3, 4]])

arr7 = np.array([[5, 6]])

concatenated = np.concatenate((arr6, arr7), axis=0)

split1, split2 = np.split(arr1, [3])

print("Concatenated:\n", concatenated)

print("Split 1:", split1)

print("Split 2:", split2)

# 7. Structured Arrays

data = np.array([('Alice', 25, 55.5), ('Bob', 30, 60.0)],

 dtype=[('name', 'U10'), ('age', 'i4'), ('weight', 'f4')])

print("Structured Array:\n", data)

print("Names:", data['name'])

print("Ages:", data['age'])

**Output :**

****

**Viva Questions with Answers (Unit–2: NumPy)**

**1. What is NumPy and why is it used in Python?**

 NumPy (Numerical Python) is a powerful library used for numerical and scientific computing in Python. It provides efficient multi-dimensional array operations and tools for mathematical, logical, linear algebra, and statistical functions.

 **2. What is the difference between a Python list and a NumPy array?**

A Python list is a general-purpose container, while a NumPy array is more memory-efficient and supports element-wise mathematical operations. NumPy arrays also support vectorization and broadcasting, which lists do not.

 **3. What is broadcasting in NumPy?**

 Broadcasting is a feature in NumPy that allows arithmetic operations on arrays of different shapes by automatically expanding them to compatible shapes without making copies of data.