**Experiment - 3**

**Name:Rhythm Kachhwaha**

**SAP:500107218**

**Batch - 4 AIML**

**Q1. Array Creation and Manipulation:**

**Create different types of arrays (1D, 2D, 3D) using various methods (np.array, np.arange, np.linspace, etc.).**

**Perform basic operations on arrays (indexing, slicing, reshaping, concatenation).**

**Investigate array attributes like shape, size, dtype, ndim.**

**Use methods like reshape, resize, flatten,**

import numpy as np

# 1D Array

array\_1d = np.array([1, 2, 3, 4, 5])

# 2D Array cr

array\_2d = np.array([[1, 2, 3], [4, 5, 6]])

# 3D Array

array\_3d = np.array([[[1, 2], [3, 4]], [[5, 6], [7, 8]]])

# Using np.arange

array\_arange = np.arange(1, 10, 2)

# Using np.linspace (create an array of 5 equally spaced values between 0 and 1)

array\_linspace = np.linspace(0, 1, 5)

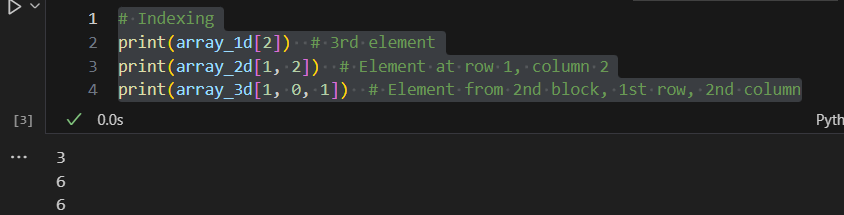
# Indexing

print(array\_1d[2])  # 3rd element

print(array\_2d[1, 2])  # Element at row 1, column 2

print(array\_3d[1, 0, 1])  # Element from 2nd block, 1st row, 2nd column

**Output:**



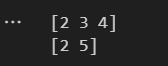
# 1D Array slicing

print(array\_1d[1:4])  # Elements from index 1 to 3

# 2D Array slicing

print(array\_2d[:, 1])  # All rows, 2nd column

**Output:**

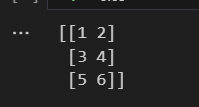


array\_arange = np.arange(1, 7)

reshaped\_array = array\_arange.reshape(3, 2)

print(reshaped\_array)

Output:



# Concatenate two arrays

concat\_array = np.concatenate((array\_1d, array\_arange))

print(concat\_array)

Output:



# Array attributes

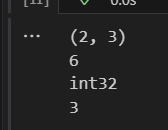
print(array\_2d.shape)  # Shape of the array

print(array\_2d.size)   # Total number of elements

print(array\_2d.dtype)  # Data type of the elements

print(array\_3d.ndim)   # Number of dimensions

Output:



# Reshape

reshaped = array\_2d.reshape(3, 2)

print(reshaped)

# Resize (modifies the original array)

resized = np.resize(array\_2d, (3, 3))

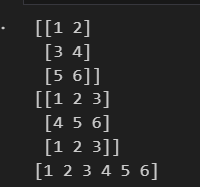
print(resized)

# Flatten (convert to 1D array)

flattened = array\_2d.flatten()

print(flattened)

Output:



**Q2. Data Loading and Preprocessing:**

**Load a dataset (e.g., CSV, Excel) using NumPy.**

**Clean and preprocess the data (handling missing values, normalization, standardization).**

**Calculate mean, median, standard deviation, variance, and other statistical measures.**

import numpy as np

# Step 1: Create a dataset and save it to CSV

data = np.array([[25, 180, 70],

                 [30, 175, np.nan],  # Missing weight

                 [22, 160, 50],

                 [35, np.nan, 90],   # Missing height

                 [28, 170, 65]])

np.savetxt('sample\_data.csv', data, delimiter=',', header="Age,Height,Weight", comments='', fmt='%10.2f')

# Step 2: Load the dataset

data = np.genfromtxt('sample\_data.csv', delimiter=',', skip\_header=1, filling\_values=np.nan)

# Step 3: Handle missing values by replacing NaN with the column mean

mean\_vals = np.nanmean(data, axis=0)

clean\_data = np.where(np.isnan(data), mean\_vals, data)

# Step 4: Normalize and Standardize the data

min\_vals = np.min(clean\_data, axis=0)

max\_vals = np.max(clean\_data, axis=0)

normalized\_data = (clean\_data - min\_vals) / (max\_vals - min\_vals)

mean\_vals = np.mean(clean\_data, axis=0)

std\_vals = np.std(clean\_data, axis=0)

standardized\_data = (clean\_data - mean\_vals) / std\_vals

# Step 5: Calculate statistical measures

mean = np.mean(clean\_data, axis=0)

median = np.median(clean\_data, axis=0)

std\_dev = np.std(clean\_data, axis=0)

variance = np.var(clean\_data, axis=0)

print("Mean:", mean)

print("Median:", median)

print("Standard Deviation:", std\_dev)

print("Variance:", variance)

Output:

