GitHub:

Video Link:

Question a

**a. Using NumPy, create a random vector of size 15 with only Integers in the range 1-20.**

1. Reshape the array to 3 by 5
2. Print array shape.
3. Replace the max in each row by 0.

Create a 2-dimensional array of size 4 x 3 (composed of 4-byte integer elements), also print the shape, type and data type

of the array.

**Code :-**

import numpy as npy;

*# 1.a*

vector = npy.random.randint(1, 20, 15)

print ("1.a Vector: ", vector)

*# 1.a.1 Reshape the array to 3 by 5*

reshaped = vector.reshape(3, 5)

print("Array after reshaped :", reshaped)

print ("1.a.2 Reshaped array shape: ", reshaped.shape)

*# 1.a.3 Replace the max in each row by 0.*

for i in range(reshaped.shape[0]):

reshaped[i, npy.where(reshaped[i] == reshaped[i].max())] = 0

print ("1.a.3 Replaced max in each row by 0: \n", reshaped)

*# Create a 2-dimensional array of size 4x3 with 4-byte integer elements*

array = npy.array([[1, 2, 3],

[4, 5, 6],

[7, 8, 9],

[10, 11, 12]], dtype=npy.int32)

*# Print the array*

print("Array:")

print(array)

*# Print the shape of the array*

print("\nShape:", array.shape)

*# Print the type of the array*

print("\nType:", type(array))

*# Print the data type of the array*

print("\nData Type:", array.dtype)

**Output :-**

**A screenshot of a computer program

Description automatically generated**

Question b

b. Write a program to compute the eigenvalues and right eigenvectors of a given square array given below:

[[ 3 -2]

[ 1 0]]

Code :-

import numpy as npy;

*# 1.b compute the eigenvalues and right eigenvectors of a given square array*

array = npy.random.randint(1, 20, (4, 3), dtype=npy.int32)

print ("1.b Array: \n", array)

print ("1.b Array shape: ", array.shape)

print ("1.b Array type: ", type(array))

print ("1.b Array data type: ", array.dtype)

newArray = npy.array([[3, -2], [1, 0]])

eigenvalues, eigenvectors = npy.linalg.eig(newArray)

print ("1.b Eigenvalues: \n", eigenvalues)

print ("1.b Eigenvectors: \n", eigenvectors)

**Output :-**

**A computer screen shot of a number

Description automatically generated**

Question c

c. Compute the sum of the diagonal element of a given array.

[[0 1 2]

[3 4 5]]

**Code :-**import numpy as npy;

*# 1.c sum of the diagonal element of a given array:*

oneC = npy.array([[0, 1, 2], [3, 4, 5]])

print ("1.c Array: \n", oneC)

print ("1.c Sum of diagonal elements: ", npy.trace(oneC))

**Output :-**

**A black text on a white background

Description automatically generated**

Question d

d. Write a NumPy program to create a new shape to an array without changing its data. Reshape 3x2:

[[1 2]

[3 4]

[5 6]]

Reshape 2x3:

[[1 2 3]

[4 5 6]]

**Code :-**

import numpy as npy;

*# 1.d new shape to an array without changing its data. Reshape 3x2:*

oneD = npy.arange(1, 7)

print ("1.d Array: ", oneD)

*# reshape to 3x2*

oneD = oneD.reshape(3, 2)

print ("1.d Reshaped array 3x2: \n", oneD)

*# reshape to 2x3*

oneD = oneD.reshape(2, 3)

print ("1.d Reshaped array 2x3: \n", oneD)

**Output :-**

**A white background with black text

Description automatically generated**