# Machine Learning Assignment-03

Praveena Goli

700743010

1. Numpy:

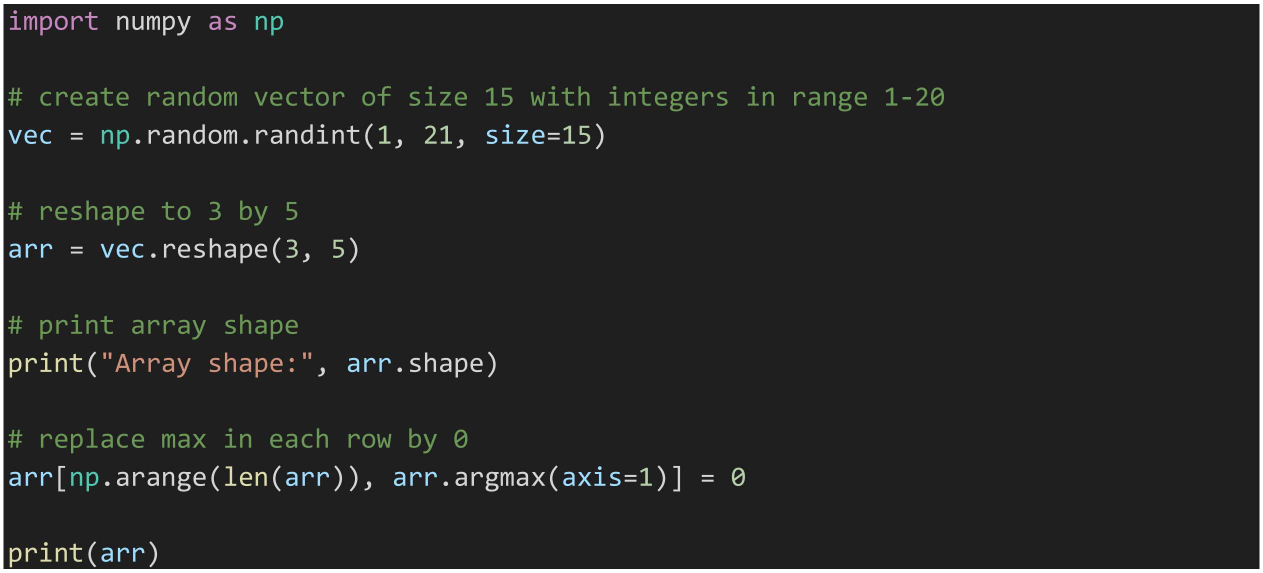
a. Using NumPy create random vector of size 15 having only Integers in the range 1-20.

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.

b. Write a program to compute the eigenvalues and right eigenvectors of a given square array given below: [[ 3 -2] [ 1 0

c. Compute the sum of the diagonal element of a given array. [[0 1 2] [3 4 5]]



```
import numpy as np

# create random vector of size 15 with integers in range 1-20
vec = np.random.randint(1, 21, size=15)

# reshape to 3 by 5
arr = vec.reshape(3, 5)

# print array shape
print("Array shape:", arr.shape)

# replace max in each row by 0
arr[np.arange(len(arr)), arr.argmax(axis=1)] = 0

print(arr)
```

```
Array shape: (3, 5)
[[15  2  0 16  8]
 [ 1 13  0  7  7]
 [ 2  9 10  0 11]]
```

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.

b. Write a program to compute the eigenvalues and right eigenvectors of a given square array given below: [[ 3 -2] [ 1 0]]

c. Compute the sum of the diagonal element of a given array. [[0 1 2] [3 4 5]]

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]]

```python
import numpy as np

# create square array
arr = np.array([[3, -2], [1, 0]])

# compute eigenvalues and right eigenvectors
eig_vals, eig_vecs = np.linalg.eig(arr)

print("Eigenvalues:", eig_vals)
print("Right eigenvectors:\n", eig_vecs)
```

```
Eigenvalues: [2. 1.]
Right eigenvectors:
 [[0.89442719 0.70710678]
 [0.4472136  0.70710678]]
```

```python
import numpy as np

# create array
arr = np.array([[0, 1, 2], [3, 4, 5]])

# compute sum of diagonal elements
diag_sum = np.trace(arr)

print("Sum of diagonal elements:", diag_sum)
```

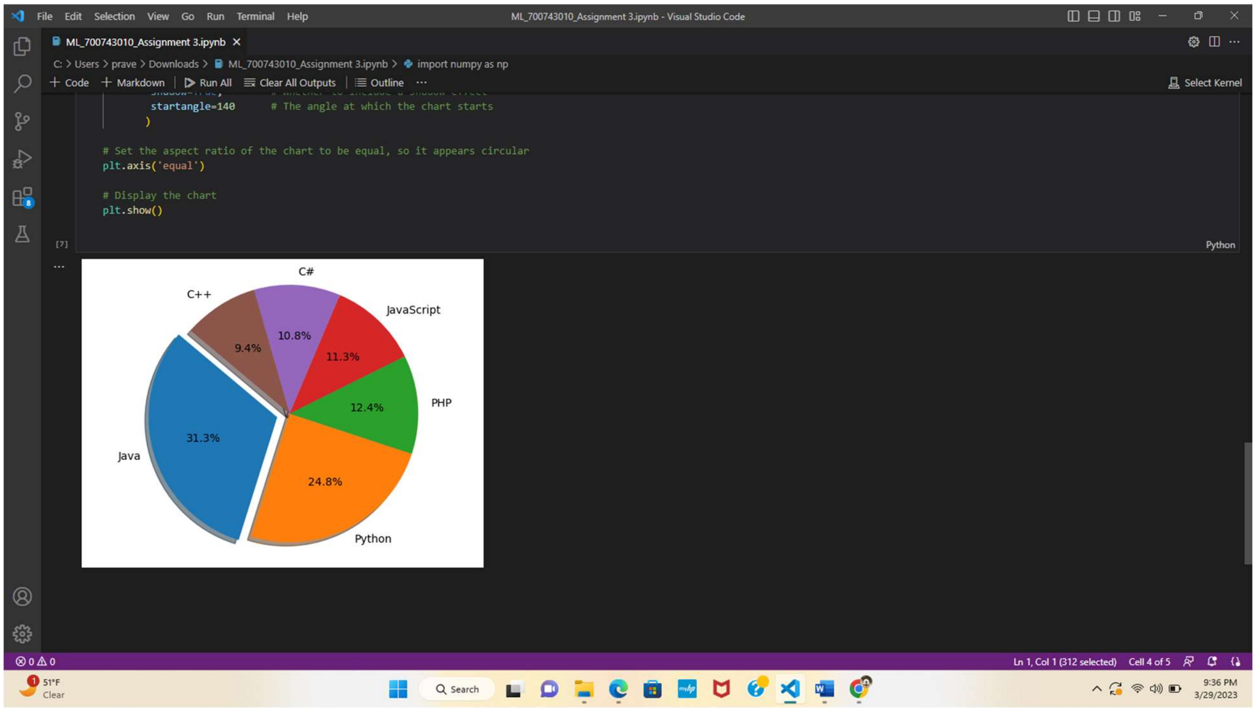
```
Sum of diagonal elements: 4
```

```
import numpy as np

# create original array
arr = np.array([[1, 2], [3, 4], [5, 6]])

# reshape to 3 by 2
arr_3by2 = arr.reshape(3, 2)

# reshape to 2 by 3
arr_2by3 = arr.reshape(2, 3)

print("Original array:\n", arr)
print("Reshaped to 3 by 2:\n", arr_3by2)
print("Reshaped to 2 by 3:\n", arr_2by3)
```

```
Original array:
 [[1 2]
 [3 4]
 [5 6]]
Reshaped to 3 by 2:
 [[1 2]
 [3 4]
 [5 6]]
Reshaped to 2 by 3:
 [[1 2 3]
 [4 5 6]]
```

```
# Import the matplotlib.pyplot module, which allows us to create plots
import matplotlib.pyplot as plt

# Define the data we want to plot
languages = 'Java', 'Python', 'PHP', 'JavaScript', 'C#', 'C++'
popularity = [22.2, 17.6, 8.8, 8, 7.7, 6.7]
colors = ["#1f77b4", "#ff7f0e", "#2ca02c", "#d62728", "#9467bd", "#8c564b"]

# Define how much we want to "explode" each slice of the pie chart
explode = (0.1, 0, 0, 0, 0, 0)

# Use the pie function to create the pie chart
plt.pie(popularity,    # The data to plot (popularity percentages)
```

```python
          explode=explode,     # How much to "explode" each slice
          labels=languages,    # Labels for each slice (the language names)
          colors=colors,       # Colors for each slice
          autopct='%1.1f%%',   # Format for the percentage labels
          shadow=True,         # Whether to include a shadow effect
          startangle=140       # The angle at which the chart starts
         )

# Set the aspect ratio of the chart to be equal, so it appears circular
plt.axis('equal')

# Display the chart
plt.show()
```



```python
          startangle=140       # The angle at which the chart starts
         )

# Set the aspect ratio of the chart to be equal, so it appears circular
plt.axis('equal')

# Display the chart
plt.show()
```

Git Hub Link : https://github.com/Goli18/ML_-ASS3.git

Video Link : https://github.com/Goli18/ML_-ASS3/blob/main/ML_ASS3.mp4