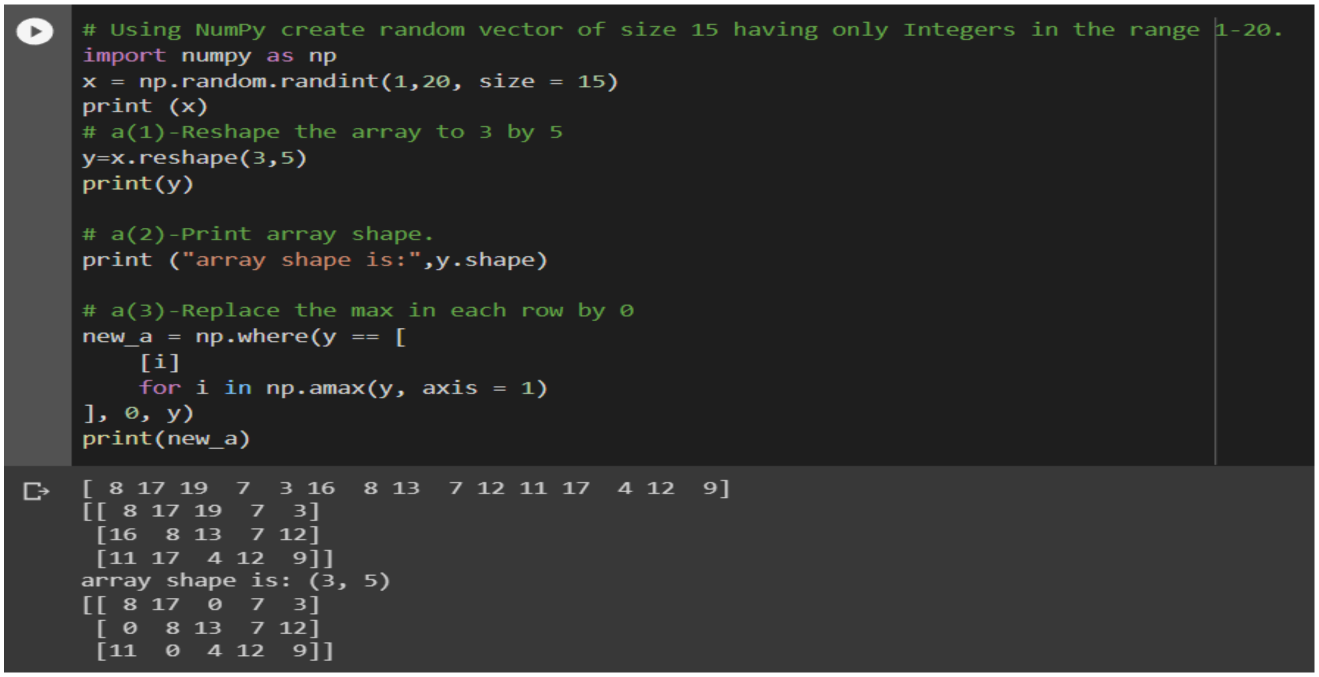
In-Class Programming Assignment-3

Name: Sammeta Jaya Sai Charan
Student ID: 700739775

Github link: https://github.com/saicharan255/ICP_ML_3.git
Video link:
https://drive.google.com/file/d/1ZeFyWcayL9fUEamvhLhEMxH5d5gRVA-N/view?usp=share_link

1.Numpy

```
# Using NumPy create random vector of size 15 having only Integers in the range 1-20.
import numpy as np
x = np.random.randint(1,20, size = 15)
print (x)
# a(1)-Reshape the array to 3 by 5
y=x.reshape(3,5)
print(y)

# a(2)-Print array shape.
print ("array shape is:",y.shape)

# a(3)-Replace the max in each row by 0
new_a = np.where(y == [
   [i]
   for i in np.amax(y, axis = 1)
], 0, y)
print(new_a)
```



```
# Using NumPy create random vector of size 15 having only Integers in the range 1-20.
import numpy as np
x = np.random.randint(1,20, size = 15)
print (x)
# a(1)-Reshape the array to 3 by 5
y=x.reshape(3,5)
print(y)

# a(2)-Print array shape.
print ("array shape is:",y.shape)

# a(3)-Replace the max in each row by 0
new_a = np.where(y == [
    [i]
    for i in np.amax(y, axis = 1)
], 0, y)
print(new_a)
```

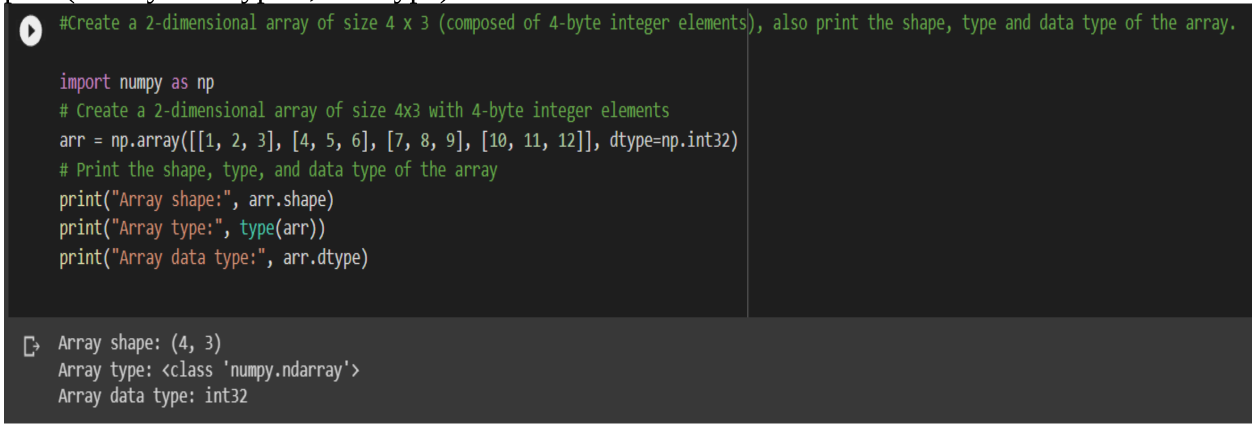
```
[ 8 17 19  7  3 16  8 13  7 12 11 17  4 12  9]
[[ 8 17 19  7  3]
 [16  8 13  7 12]
 [11 17  4 12  9]]
array shape is: (3, 5)
[[ 8 17  0  7  3]
 [ 0  8 13  7 12]
 [11  0  4 12  9]]
```

Description:

we used Numpy libarariesc created a random vector with size 15 ranging from 1-20 printed the vector, then the vector array is reshaped into 3 by 5 and printed the array shape. From the obtained array we replaced the maximum number with o in each row and printed the new array.

#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.

import numpy as np
Create a 2-dimensional array of size 4x3 with 4-byte integer elements
arr = np.array([[1, 2, 3], [4, 5, 6], [7, 8, 9], [10, 11, 12]], dtype=np.int32)
Print the shape, type, and data type of the array
print("Array shape:", arr.shape)
print("Array type:", type(arr))
print("Array data type:", arr.dtype)



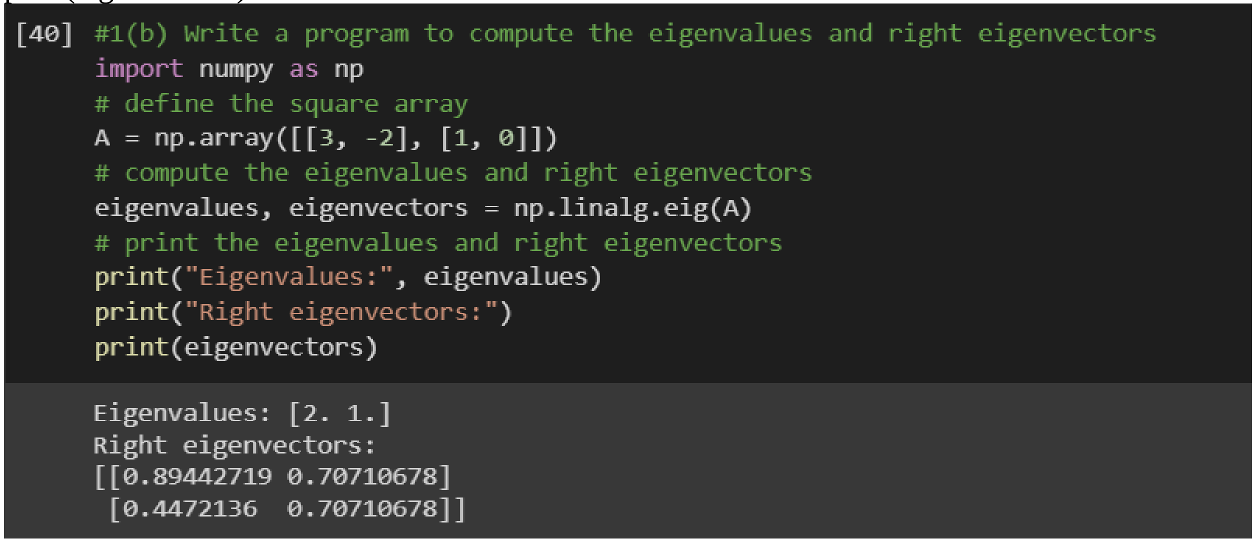
```
#Create a 2-dimensional array of size 4 x 3 (composed of 4-byte integer elements), also print the shape, type and data type of the array.

import numpy as np
# Create a 2-dimensional array of size 4x3 with 4-byte integer elements
arr = np.array([[1, 2, 3], [4, 5, 6], [7, 8, 9], [10, 11, 12]], dtype=np.int32)
# Print the shape, type, and data type of the array
print("Array shape:", arr.shape)
print("Array type:", type(arr))
print("Array data type:", arr.dtype)
```

```
Array shape: (4, 3)
Array type: <class 'numpy.ndarray'>
Array data type: int32
```

Description:

Created a 2D array with size 4*3 with 4-byte integer elements and printed the shape, type, and data type of the array

#**1(b)** Write a program to compute the eigenvalues and right eigenvectors
import numpy as np
# define the square array
A = np.array([[3, -2], [1, 0]])
# compute the eigenvalues and right eigenvectors
eigenvalues, eigenvectors = np.linalg.eig(A)
# print the eigenvalues and right eigenvectors
print("Eigenvalues:", eigenvalues)
print("Right eigenvectors:")
print(eigenvectors)



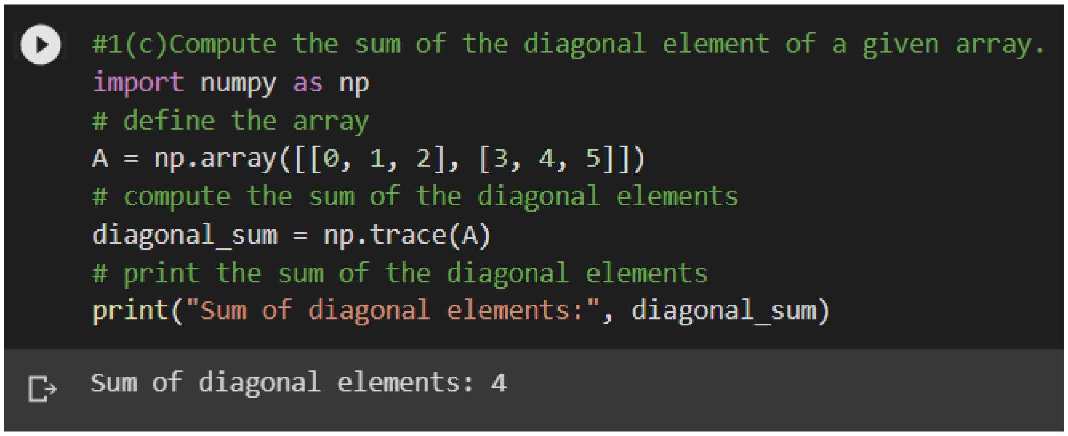
```
[40] #1(b) Write a program to compute the eigenvalues and right eigenvectors
     import numpy as np
     # define the square array
     A = np.array([[3, -2], [1, 0]])
     # compute the eigenvalues and right eigenvectors
     eigenvalues, eigenvectors = np.linalg.eig(A)
     # print the eigenvalues and right eigenvectors
     print("Eigenvalues:", eigenvalues)
     print("Right eigenvectors:")
     print(eigenvectors)
```

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

Description:

Imported Numpy libraries created a square array, computed the eigen values and right eigen vectors using the function “np.linalg.eig(A)” and printed those respectively.

#**1(c)**Compute the sum of the diagonal element of a given array.
import numpy as np
# define the array
A = np.array([[0, 1, 2], [3, 4, 5]])
# compute the sum of the diagonal elements
diagonal_sum = np.trace(A)
# print the sum of the diagonal elements
print("Sum of diagonal elements:", diagonal_sum)



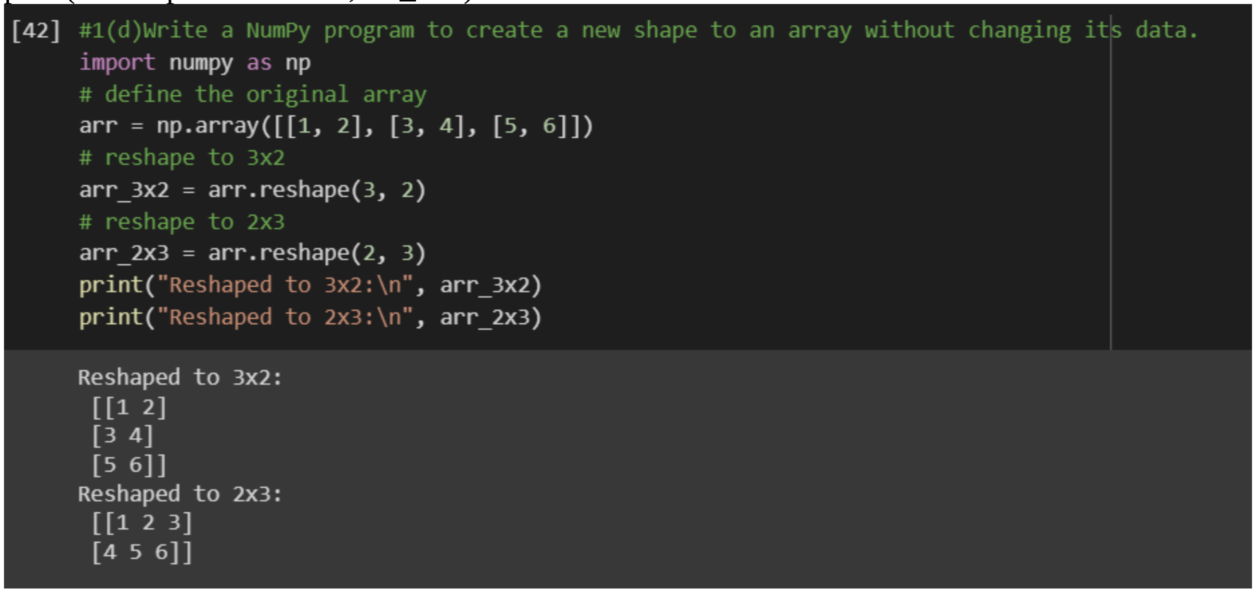
```
#1(c)Compute the sum of the diagonal element of a given array.
import numpy as np
# define the array
A = np.array([[0, 1, 2], [3, 4, 5]])
# compute the sum of the diagonal elements
diagonal_sum = np.trace(A)
# print the sum of the diagonal elements
print("Sum of diagonal elements:", diagonal_sum)
```

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

Description:

Numpy libraries is imported and created an array and found the sum of the diagonal elements and printed that sum.

#**1(d)**Write a NumPy program to create a new shape to an array without changing its data.
import numpy as np
# define the original array
arr = np.array([[1, 2], [3, 4], [5, 6]])
# reshape to 3x2
arr_3x2 = arr.reshape(3, 2)
# reshape to 2x3
arr_2x3 = arr.reshape(2, 3)
print("Reshaped to 3x2:\n", arr_3x2)
print("Reshaped to 2x3:\n", arr_2x3)



```
[42] #1(d)Write a NumPy program to create a new shape to an array without changing its data.
     import numpy as np
     # define the original array
     arr = np.array([[1, 2], [3, 4], [5, 6]])
     # reshape to 3x2
     arr_3x2 = arr.reshape(3, 2)
     # reshape to 2x3
     arr_2x3 = arr.reshape(2, 3)
     print("Reshaped to 3x2:\n", arr_3x2)
     print("Reshaped to 2x3:\n", arr_2x3)
```

```
Reshaped to 3x2:
 [[1 2]
 [3 4]
 [5 6]]
Reshaped to 2x3:
 [[1 2 3]
 [4 5 6]]
```

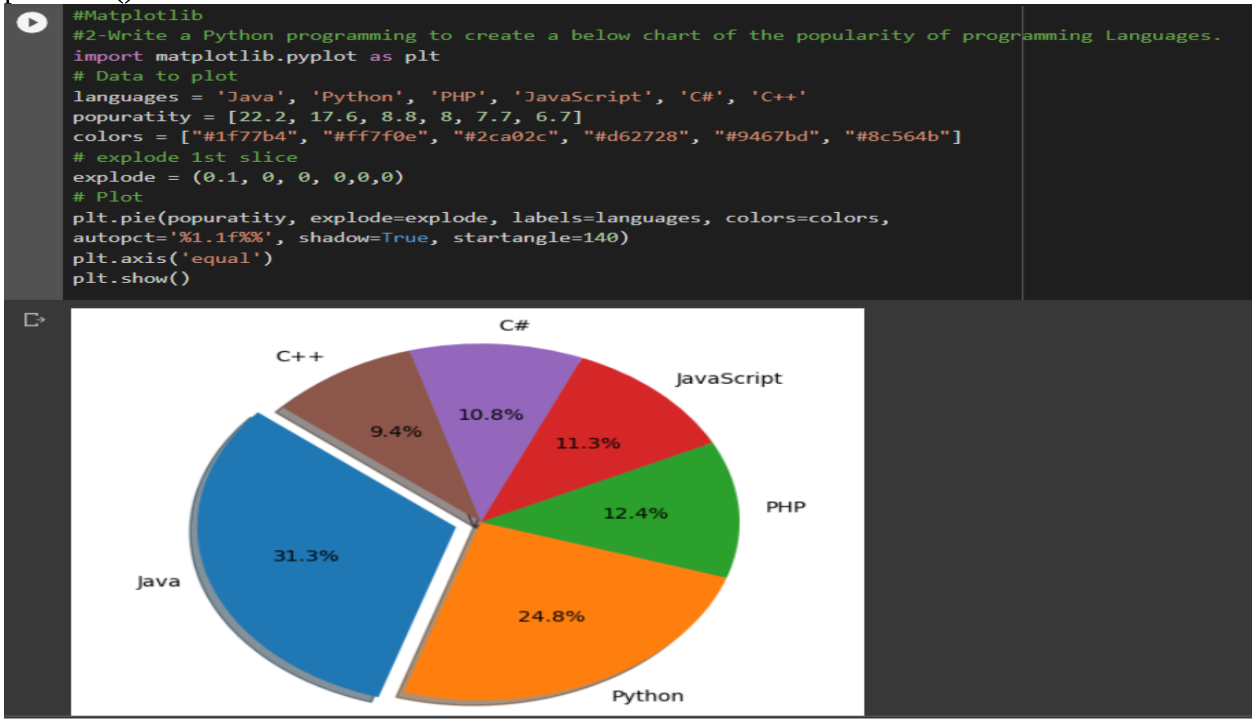
Description:

Numpy libraries are imported then defined the array with the given dat, by using the **arr.reshape()** function we have changed size of the array to 3*2 and 2*3 with out changing the original data and printed those arrays respectively.

#Matplotlib

#2-Write a Python programming to create a below chart of the popularity of programming Languages.

```
import matplotlib.pyplot as plt
# Data to plot
languages = 'Java', 'Python', 'PHP', 'JavaScript', 'C#', 'C++'
popuratity = [22.2, 17.6, 8.8, 8, 7.7, 6.7]
colors = ["#1f77b4", "#ff7f0e", "#2ca02c", "#d62728", "#9467bd", "#8c564b"]
# explode 1st slice
explode = (0.1, 0, 0, 0,0,0)
# Plot
plt.pie(popuratity, explode=explode, labels=languages, colors=colors,
autopct='%1.1f%%', shadow=True, startangle=140)
plt.axis('equal')
plt.show()
```



```
#Matplotlib
#2-Write a Python programming to create a below chart of the popularity of programming Languages.
import matplotlib.pyplot as plt
# Data to plot
languages = 'Java', 'Python', 'PHP', 'JavaScript', 'C#', 'C++'
popuratity = [22.2, 17.6, 8.8, 8, 7.7, 6.7]
colors = ["#1f77b4", "#ff7f0e", "#2ca02c", "#d62728", "#9467bd", "#8c564b"]
# explode 1st slice
explode = (0.1, 0, 0, 0,0,0)
# Plot
plt.pie(popuratity, explode=explode, labels=languages, colors=colors,
autopct='%1.1f%%', shadow=True, startangle=140)
plt.axis('equal')
plt.show()
```

Description:

Imported Matplotlib libraries, languages , popularity, colors were assigned. To plot the data we used different arguments to get a pie chart.