DEPARTMENT OF CSE

LAB REPORT 05

Course Title: Statistics for Data Science

Course Code: CSE303

Section: 03

Semester: Spring 2022

Date: 24-03-2022

Submitted To

Md Al-Imran

Lecturer

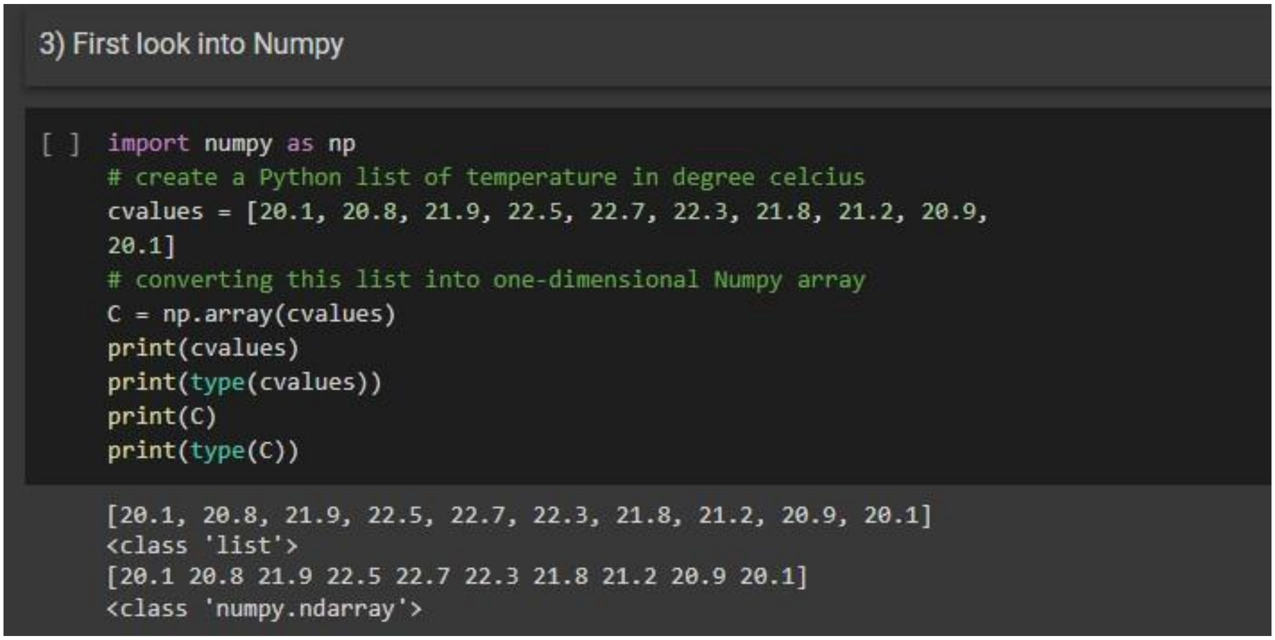
Department of Computer Science & Engineering

Submitted by

Aahadul Islam Fardin

2019-1-60-224

Problem No: 03

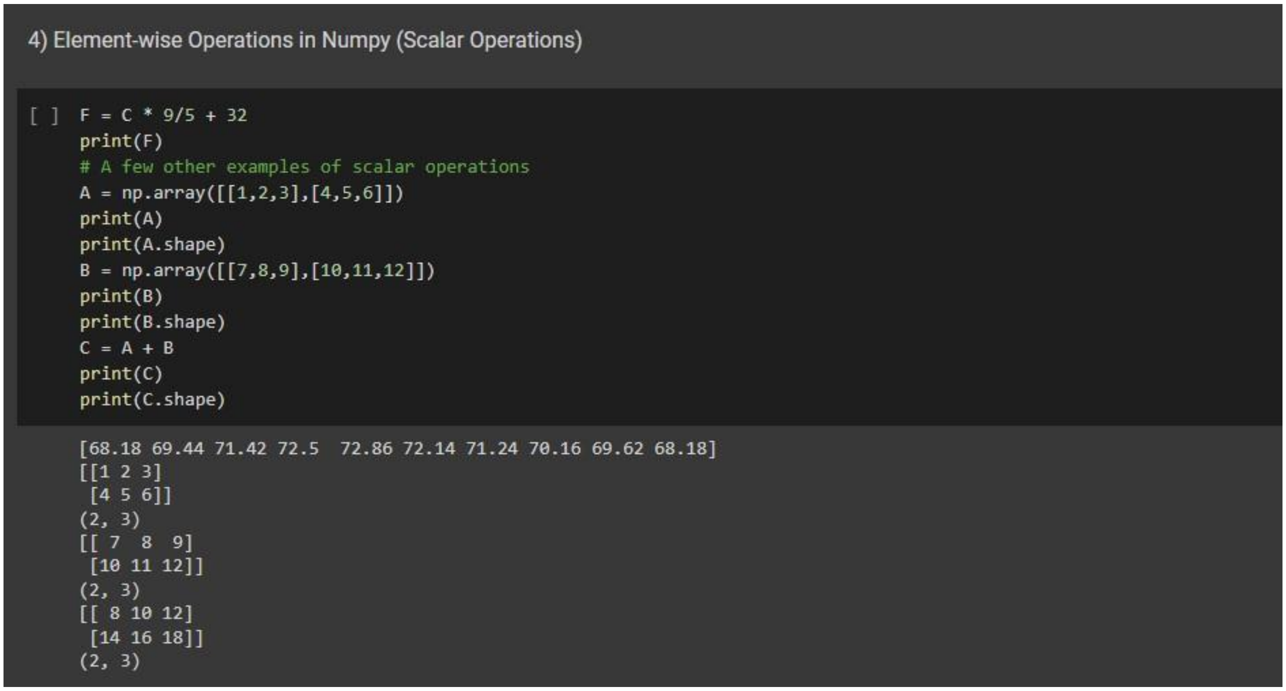


3) First look into Numpy

```
import numpy as np
# create a Python list of temperature in degree celcius
cvalues = [20.1, 20.8, 21.9, 22.5, 22.7, 22.3, 21.8, 21.2, 20.9,
20.1]
# converting this list into one-dimensional Numpy array
C = np.array(cvalues)
print(cvalues)
print(type(cvalues))
print(C)
print(type(C))
```

```
[20.1, 20.8, 21.9, 22.5, 22.7, 22.3, 21.8, 21.2, 20.9, 20.1]
<class 'list'>
[20.1 20.8 21.9 22.5 22.7 22.3 21.8 21.2 20.9 20.1]
<class 'numpy.ndarray'>
```

Problem No: 04

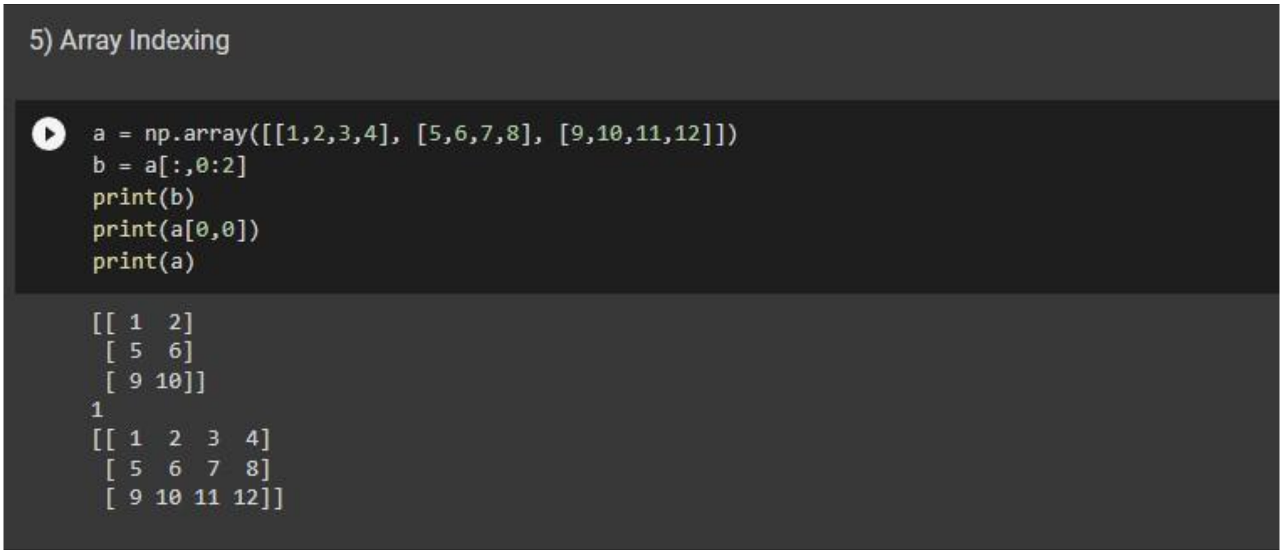


4) Element-wise Operations in Numpy (Scalar Operations)

```
F = C * 9/5 + 32
print(F)
# A few other examples of scalar operations
A = np.array([[1,2,3],[4,5,6]])
print(A)
print(A.shape)
B = np.array([[7,8,9],[10,11,12]])
print(B)
print(B.shape)
C = A + B
print(C)
print(C.shape)
```

```
[68.18 69.44 71.42 72.5  72.86 72.14 71.24 70.16 69.62 68.18]
[[1 2 3]
 [4 5 6]]
(2, 3)
[[ 7  8  9]
 [10 11 12]]
(2, 3)
[[ 8 10 12]
 [14 16 18]]
(2, 3)
```

Problem No: 05

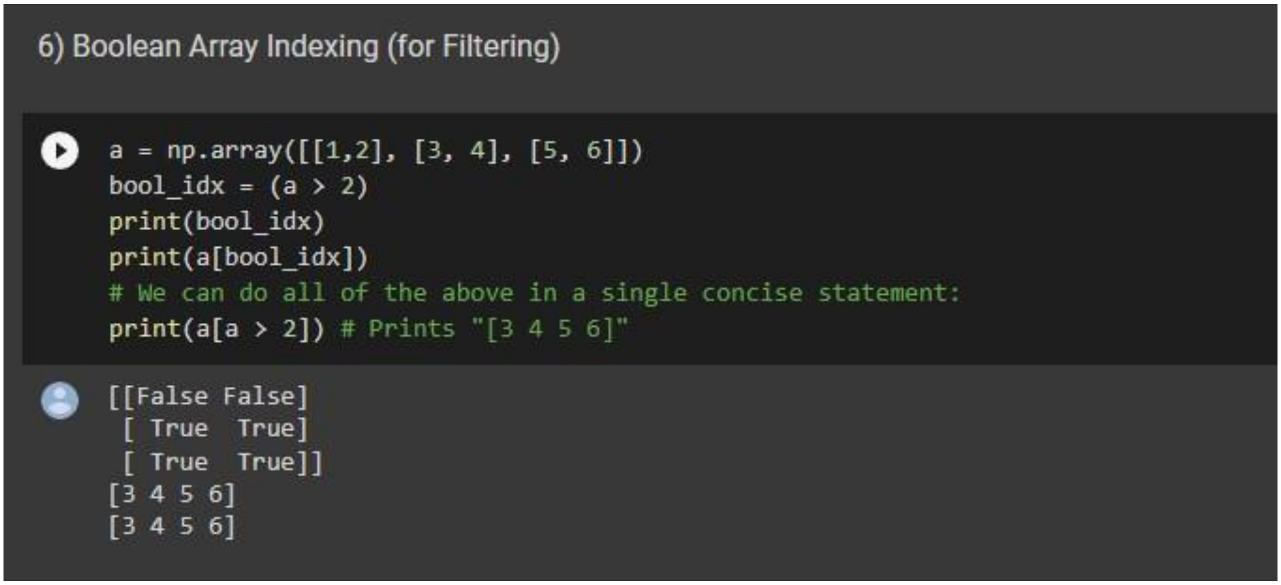


5) Array Indexing

```
a = np.array([[1,2,3,4], [5,6,7,8], [9,10,11,12]])
b = a[:,0:2]
print(b)
print(a[0,0])
print(a)
```

```
[[ 1  2]
 [ 5  6]
 [ 9 10]]
1
[[ 1  2  3  4]
 [ 5  6  7  8]
 [ 9 10 11 12]]
```

Problem No: 06

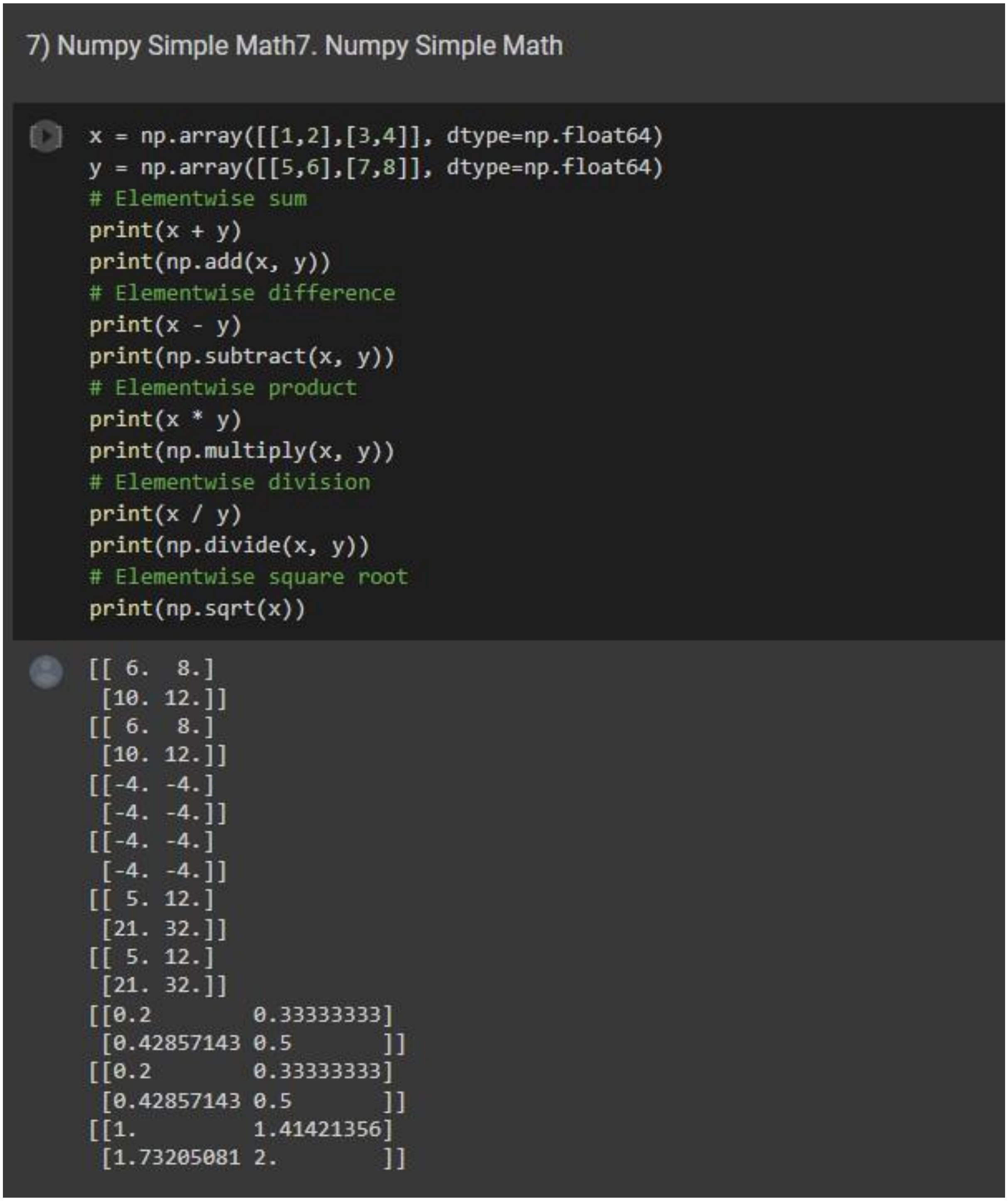


6) Boolean Array Indexing (for Filtering)

```
a = np.array([[1,2], [3, 4], [5, 6]])
bool_idx = (a > 2)
print(bool_idx)
print(a[bool_idx])
# We can do all of the above in a single concise statement:
print(a[a > 2]) # Prints "[3 4 5 6]"
```

```
[[False False]
 [ True  True]
 [ True  True]]
[3 4 5 6]
[3 4 5 6]
```

Problem No: 07

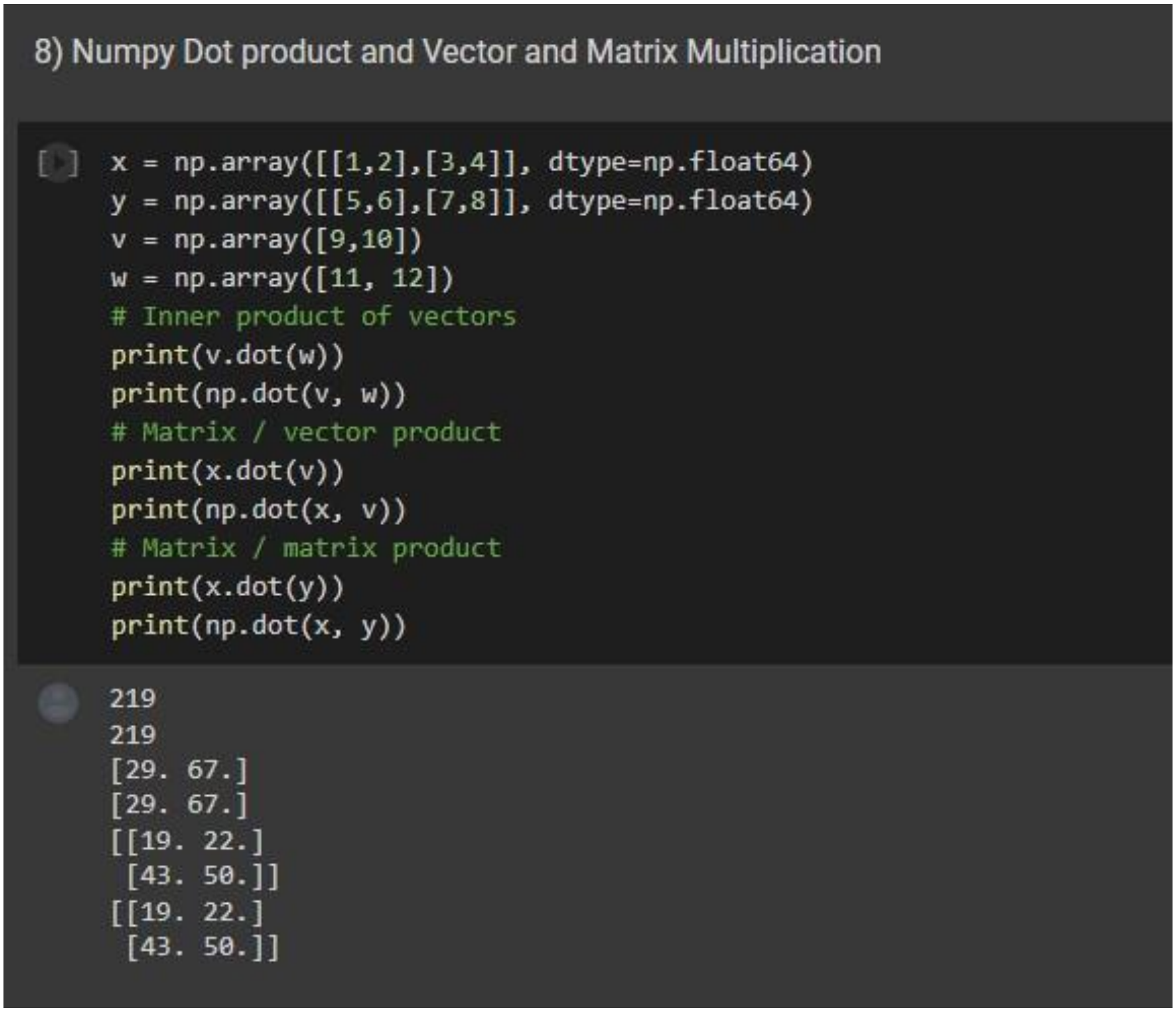


7) Numpy Simple Math7. Numpy Simple Math

```
x = np.array([[1,2],[3,4]], dtype=np.float64)
y = np.array([[5,6],[7,8]], dtype=np.float64)
# Elementwise sum
print(x + y)
print(np.add(x, y))
# Elementwise difference
print(x - y)
print(np.subtract(x, y))
# Elementwise product
print(x * y)
print(np.multiply(x, y))
# Elementwise division
print(x / y)
print(np.divide(x, y))
# Elementwise square root
print(np.sqrt(x))
```

```
[[ 6.  8.]
 [10. 12.]]
[[ 6.  8.]
 [10. 12.]]
[[-4. -4.]
 [-4. -4.]]
[[-4. -4.]
 [-4. -4.]]
[[ 5. 12.]
 [21. 32.]]
[[ 5. 12.]
 [21. 32.]]
[[0.2        0.33333333]
 [0.42857143 0.5       ]]
[[0.2        0.33333333]
 [0.42857143 0.5       ]]
[[1.         1.41421356]
 [1.73205081 2.        ]]
```

Problem No: 08

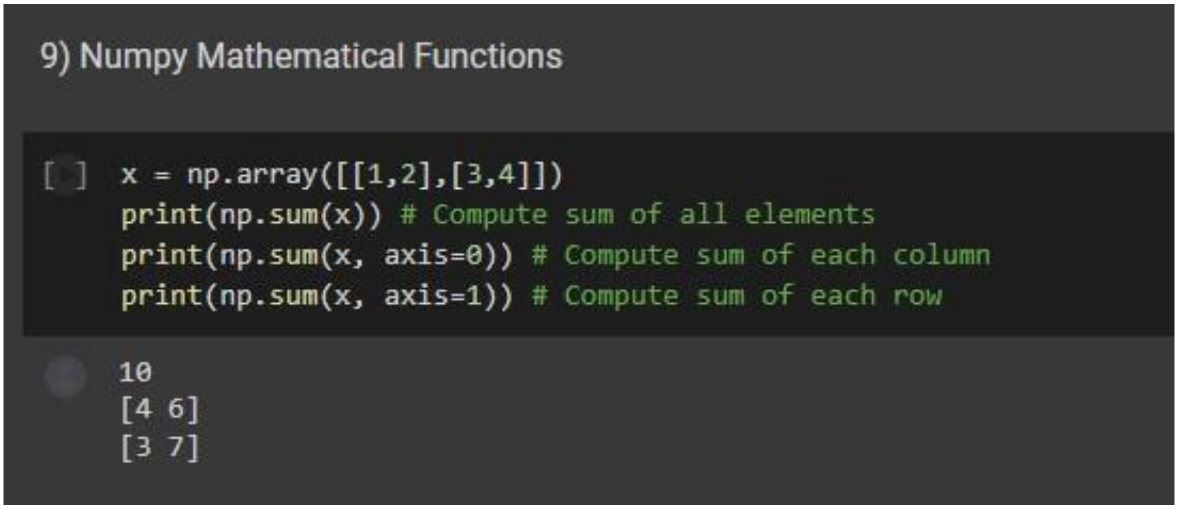


8) Numpy Dot product and Vector and Matrix Multiplication

```
x = np.array([[1,2],[3,4]], dtype=np.float64)
y = np.array([[5,6],[7,8]], dtype=np.float64)
v = np.array([9,10])
w = np.array([11, 12])
# Inner product of vectors
print(v.dot(w))
print(np.dot(v, w))
# Matrix / vector product
print(x.dot(v))
print(np.dot(x, v))
# Matrix / matrix product
print(x.dot(y))
print(np.dot(x, y))
```

```
219
219
[29. 67.]
[29. 67.]
[[19. 22.]
 [43. 50.]]
[[19. 22.]
 [43. 50.]]
```

Problem No: 09

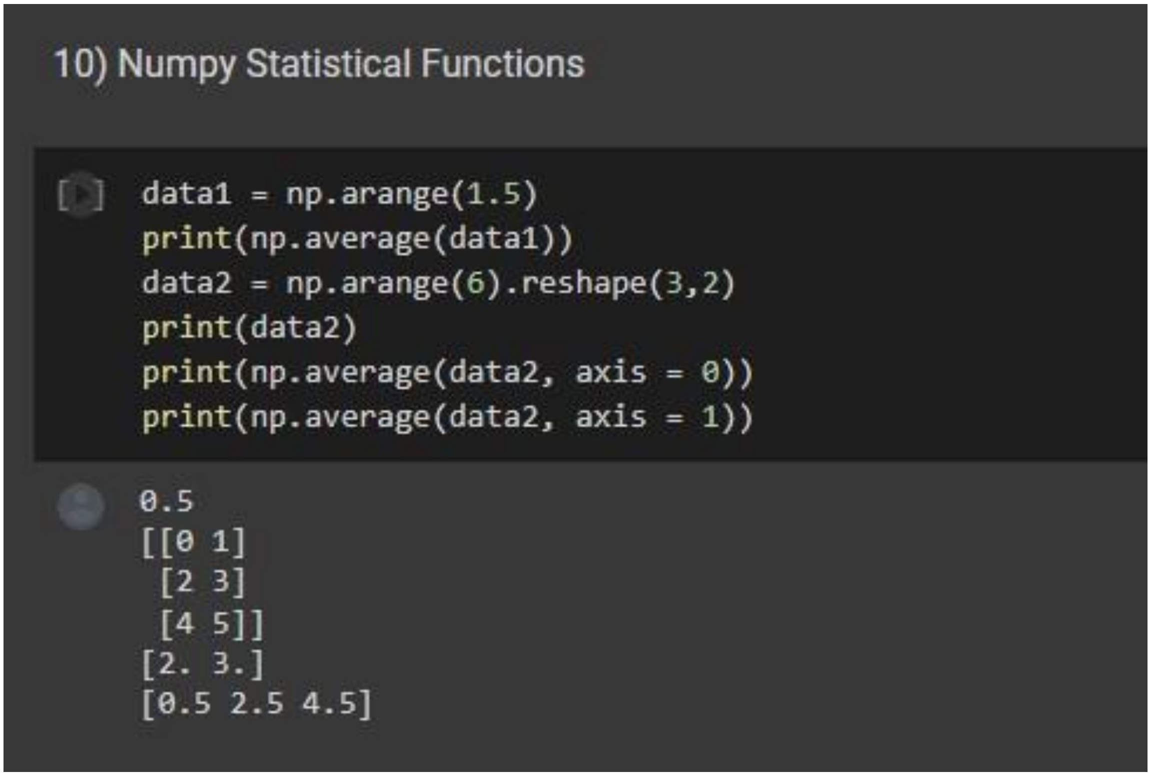


9) Numpy Mathematical Functions

```
x = np.array([[1,2],[3,4]])
print(np.sum(x)) # Compute sum of all elements
print(np.sum(x, axis=0)) # Compute sum of each column
print(np.sum(x, axis=1)) # Compute sum of each row
```

```
10
[4 6]
[3 7]
```

Problem No: 10

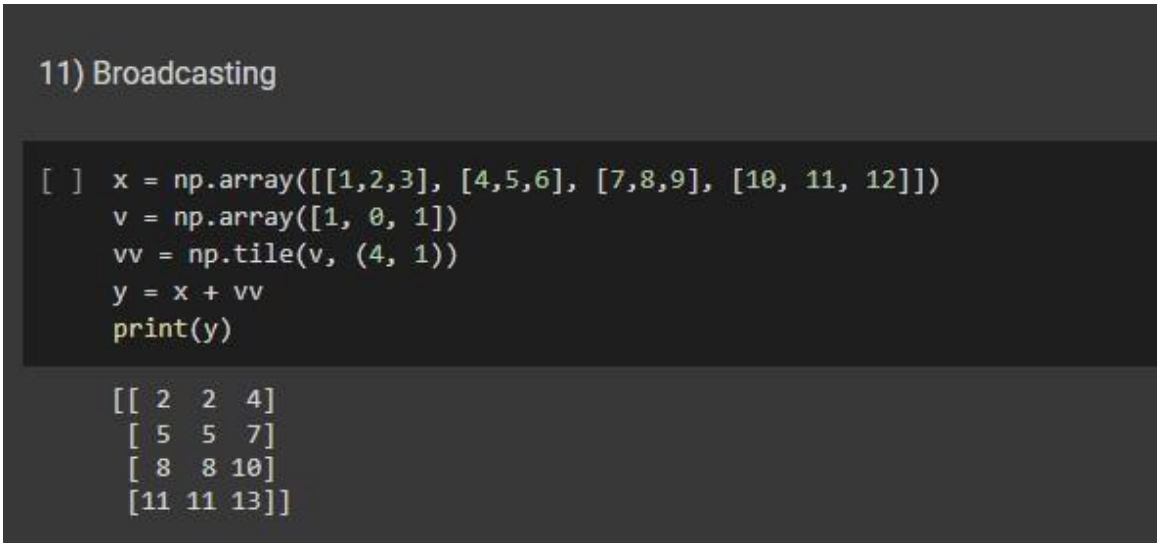


10) Numpy Statistical Functions

```
data1 = np.arange(1.5)
print(np.average(data1))
data2 = np.arange(6).reshape(3,2)
print(data2)
print(np.average(data2, axis = 0))
print(np.average(data2, axis = 1))
```

```
0.5
[[0 1]
 [2 3]
 [4 5]]
[2. 3.]
[0.5 2.5 4.5]
```

Problem No: 11

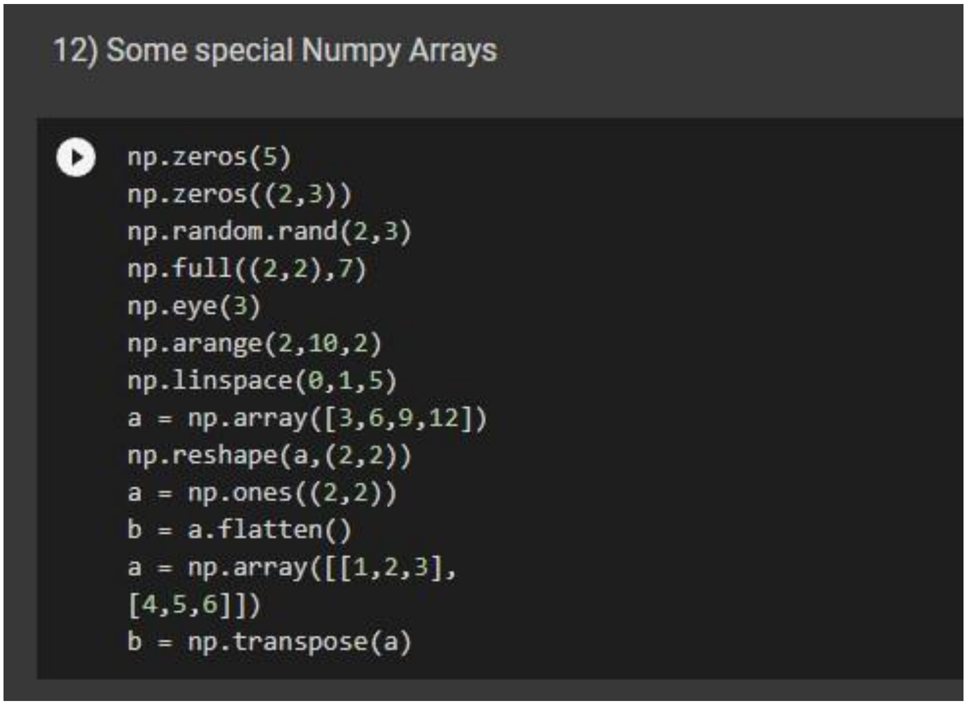


11) Broadcasting

```
x = np.array([[1,2,3], [4,5,6], [7,8,9], [10, 11, 12]])
v = np.array([1, 0, 1])
vv = np.tile(v, (4, 1))
y = x + vv
print(y)
```

```
[[ 2  2  4]
 [ 5  5  7]
 [ 8  8 10]
 [11 11 13]]
```

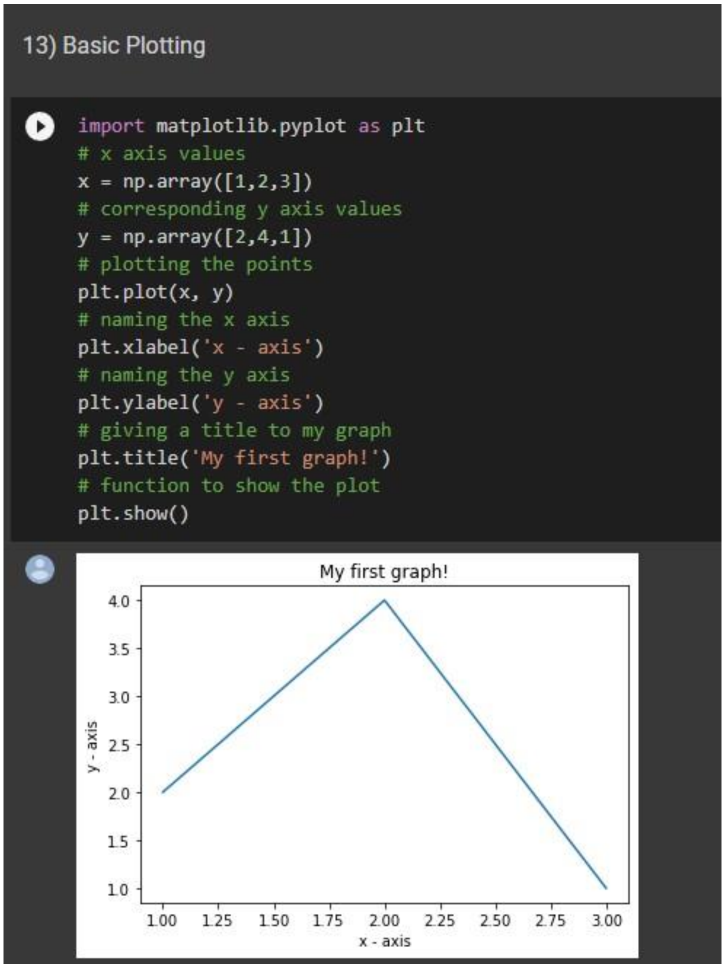
Problem No:12



12) Some special Numpy Arrays

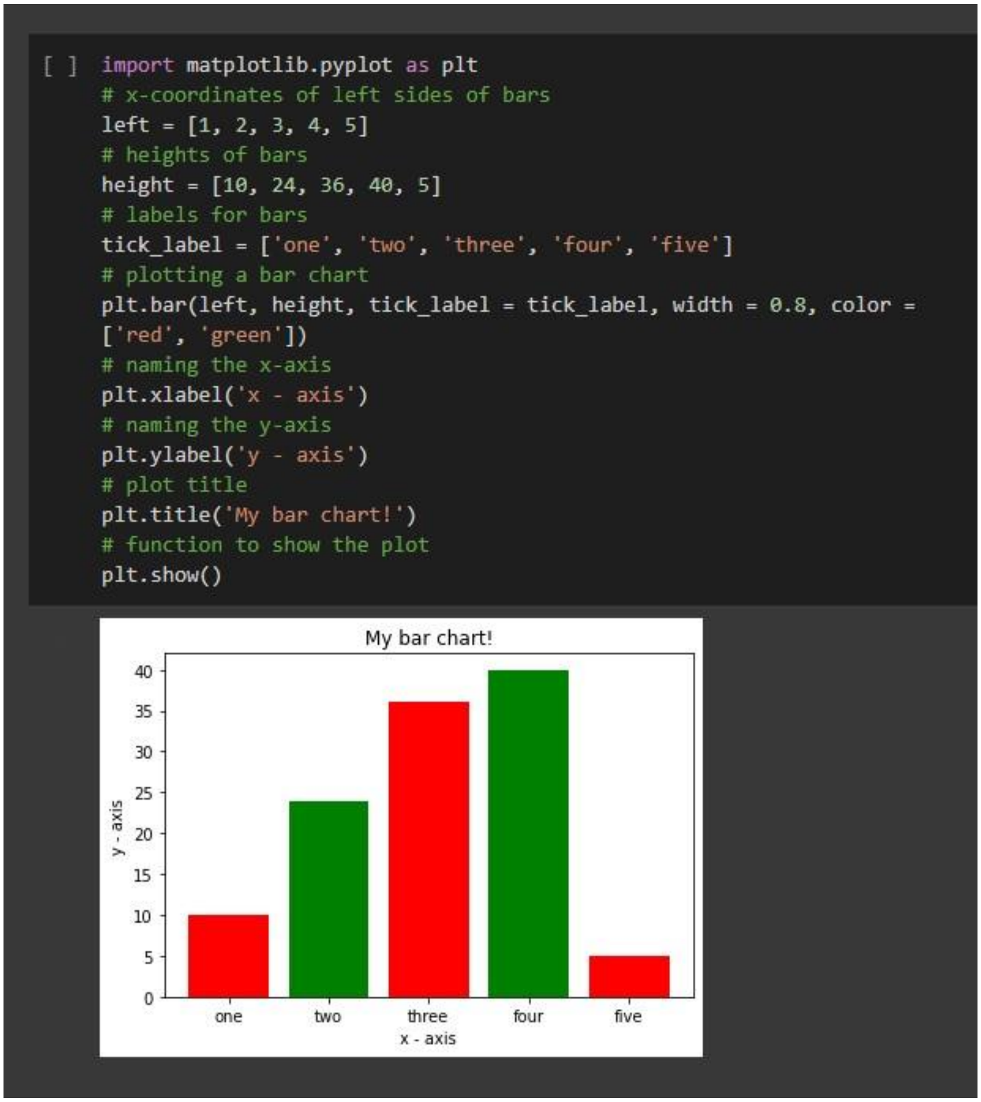
```
np.zeros(5)
np.zeros((2,3))
np.random.rand(2,3)
np.full((2,2),7)
np.eye(3)
np.arange(2,10,2)
np.linspace(0,1,5)
a = np.array([3,6,9,12])
np.reshape(a,(2,2))
a = np.ones((2,2))
b = a.flatten()
a = np.array([[1,2,3],
[4,5,6]])
b = np.transpose(a)
```

Problem No: 13



13) Basic Plotting

```
import matplotlib.pyplot as plt
# x axis values
x = np.array([1,2,3])
# corresponding y axis values
y = np.array([2,4,1])
# plotting the points
plt.plot(x, y)
# naming the x axis
plt.xlabel('x - axis')
# naming the y axis
plt.ylabel('y - axis')
# giving a title to my graph
plt.title('My first graph!')
# function to show the plot
plt.show()
```



```
[ ]  import matplotlib.pyplot as plt
     # x-coordinates of left sides of bars
     left = [1, 2, 3, 4, 5]
     # heights of bars
     height = [10, 24, 36, 40, 5]
     # labels for bars
     tick_label = ['one', 'two', 'three', 'four', 'five']
     # plotting a bar chart
     plt.bar(left, height, tick_label = tick_label, width = 0.8, color =
     ['red', 'green'])
     # naming the x-axis
     plt.xlabel('x - axis')
     # naming the y-axis
     plt.ylabel('y - axis')
     # plot title
     plt.title('My bar chart!')
     # function to show the plot
     plt.show()
```