Text

Description automatically generated

**CSE303: Lab 5 Report**

**SAZZAD HOSSEN**

**2019-1-60-063**

**LAB 5 CODE :**

import numpy as np

print("First look into Numpy")

cvalues = [20.1, 20.8, 21.9, 22.5, 22.7, 22.3, 21.8, 21.2, 20.9,

20.1]

C = np.array(cvalues)

print(cvalues)

print(type(cvalues))

print(C)

print(type(C))

print("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)

print("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)

print("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])

print("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))

print("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))

print("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))

print("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))

print("Broadcasting")

x = np.array([[1,2,3], [4,5,6], [7,8,9], [10, 11, 12]])

v = np.array([1, 0, 1])

y = np.empty\_like(x)

# Add the vector v to each row of the matrix x with an explicit loop

for i in range(4):

      y[i, :] = x[i, :] + v

##Better Solution

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)

print("Using Broadcasting")

x = np.array([[1,2,3], [4,5,6], [7,8,9], [10, 11, 12]])

v = np.array([1, 0, 1])

y = x + v # Add v to each row of x using broadcasting

print(y)

print("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)

print("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

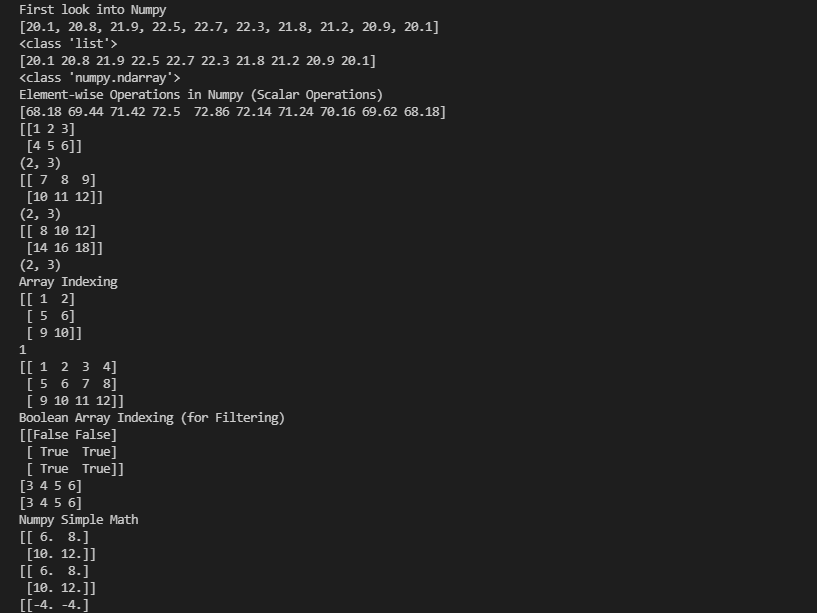
plt.title('My bar chart!')

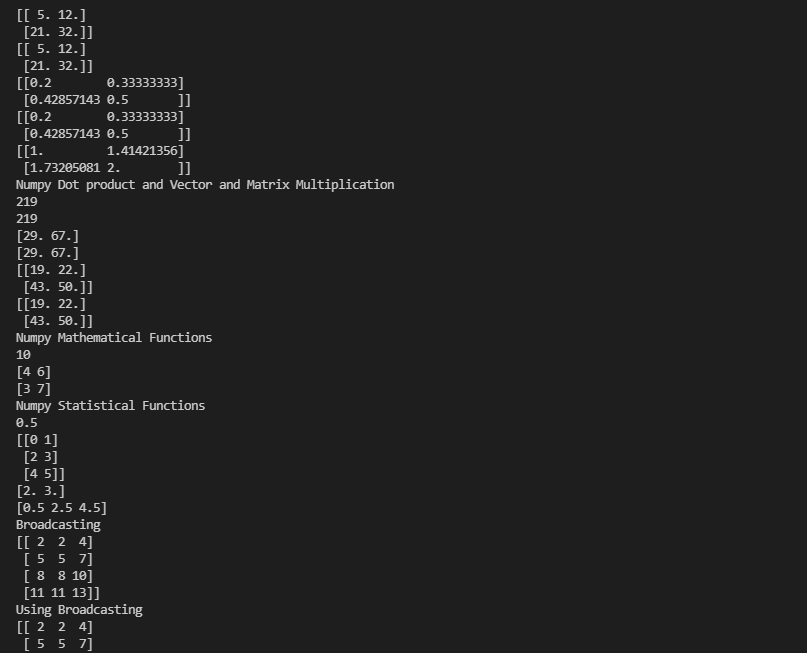
# function to show the plot

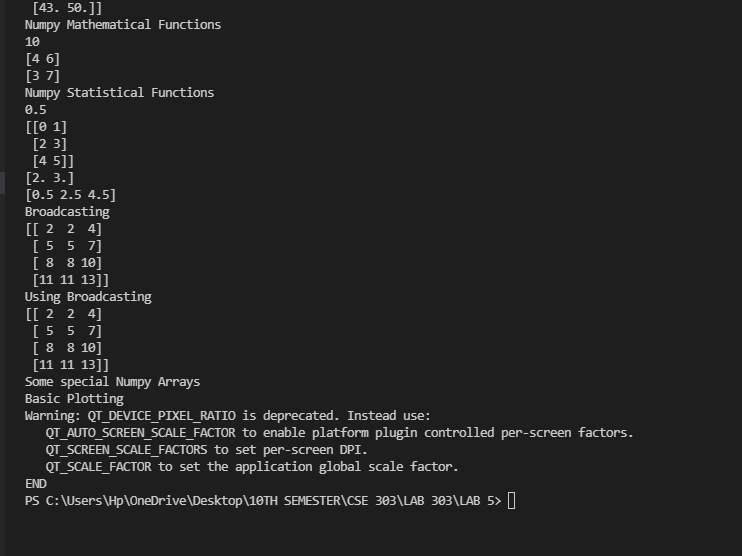
plt.show()

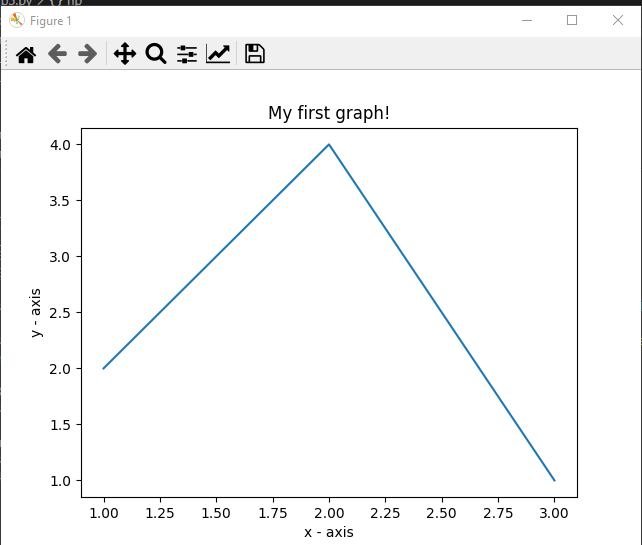
print("END")

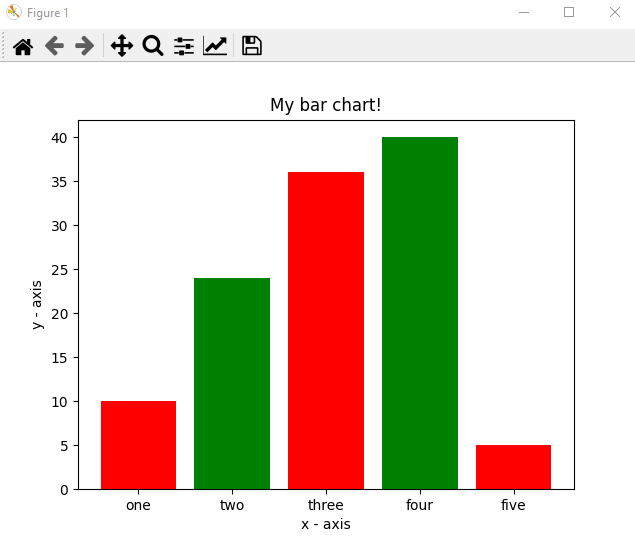
**OUTPUT**

****

****

****

****

****