Week 1

February 17, 2022

```
In [2]: import numpy as np

        # 1a. Make the array [0, 1, 2] with np.array

        arr = np.array([0,1,2])

        #note: array element dtype is not mentioned

In [4]: #checking data type of values

        arr.dtype

Out[4]: dtype('int64')

In [11]: #note: dtype can be mentioned during array creation

         arr = np.array([0.2,1.5,2.6], dtype='int')

In [12]: arr.dtype

Out[12]: dtype('int64')

In [14]: #float values are converted to int as dtype is int

         print(arr)

[0 1 2]


In [24]: # 1b. Make the array [0, 1, 2] with np.array
         arr = np.arange(0,3) #np.arange(starting value, end value+1, step-size)
         print(arr)

[0 1 2]


In [27]: arr = np.arange(0,10,3) #np.arange(starting value, end value+1, step-size)
         print(arr)
```

```
[0 3 6 9]
```

In [29]:
```
# 1c. Make the array [0, 0, 0] with np.zeros

arr = np.zeros(5)
print(arr)
#note that default zeros are of float type
```

```
[0. 0. 0. 0. 0.]
```

In [31]:
```
#making array of zeroes of int type

arr = np.zeros(5, dtype='int')
print(arr)
```

```
[0 0 0 0 0]
```

In [35]:
```
# 1d. Make the matrix [[0, 1], [2, 3]] with np.array

arr = np.array([[0,1],[2,3]], dtype='float')
print(arr)
```

```
[[0. 1.]
 [2. 3.]]
```

In [39]:
```
# 1e. Make the matrix [[0, 1], [2, 3]] with np.arange and np.reshape

a = np.arange(0,4) # arange generates array [0,1,2,3]
print(a)

b = a.reshape(2, 2) # reshape takes arguments are (no_of_rows, no_of_columns)
print(b)
```

```
[0 1 2 3]
array shape is (4,)
[[0 1]
 [2 3]]
```

In [37]:
```
#in a single command

arr = np.arange(0,4).reshape(2,2)
print(arr)
```

```
[[0 1]
 [2 3]]
```

In [38]:
```
# 1f. Print the shape of the matrix from the previous part with .shape

print("array shape is ", np.shape(arr))
```

```
array shape is  (2, 2)
```

In [41]:
```
# 1g. Print the number of rows in the matrix from previous part with .shape

#determine type of np.shape output

type(np.shape(arr))
```

Out[41]: tuple

In [45]:
```
#row is the first value of the tuple

print(np.shape(arr)[0])
```

```
2
```

In [46]:
```
#OR

row, col = np.shape(arr)
print(row)
```

```
2
```

In [49]:
```
# 1i. Take the average of elements in a vector (np.average)

arr = np.array([0,1,2,])
print(np.average(arr))
```

```
1.75
```

In [52]:
```
# 1j. Add two vectors element-wise, which means that the each element in the result v

arr1 = np.array([9, 8, 7])
arr2 = np.array([1, 2, 3])

sum_arr = np.add(arr1, arr2)

print(sum_arr)
```

```
[10 10 10]
```

In [54]:
```
# 1l. Take the sum of elements in a vector (np.sum)

arr = np.array([1,5,7])
sum_all = np.sum(arr)
print(sum_all)
```

```
13
```

In [57]:
```
# 1m. Take the square root of a number (np.sqrt)

num = int(input("Enter a number:"))
print("square root is:", np.sqrt(num))
```

```
Enter a number:6
square root is: 2.449489742783178
```