

# Learning Numpy Library and practise

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

## Basics

```
In [ ]: One_DArray = np.array([12,3,4,3,5,3], dtype='float16')
One_DArray
```

```
Out[ ]: array([12., 3., 4., 3., 5., 3.], dtype=float16)
```

```
In [ ]: Two_DArray = np.array([[4.0,5.0,7.0],[4.0,5.0,7.0], [4.0,5.0,3.0],[4.0,5.0,3.0]])
Two_DArray
```

```
Out[ ]: array([[4., 5., 7.],
               [4., 5., 7.],
               [4., 5., 3.],
               [4., 5., 3.]])
```

```
In [ ]: Three_DArray = np.array([[[4.0,5.0,7.0],[4.0,5.0,7.0]],[[4.0,5.0,7.0],[4.0,5.0,7.0]],[[
Three_DArray
```

```
Out[ ]: array([[[4., 5., 7.],
                [4., 5., 7.]],

               [[4., 5., 7.],
                [4., 5., 7.]],

               [[4., 5., 7.],
                [4., 5., 7.]],

               [[4., 5., 7.],
                [4., 5., 7.]],

               [[4., 5., 7.],
                [4., 5., 7.]])
```

```
In [ ]: # what type is that array

print('Type of One_DArray : ', type(One_DArray))
print('Type of Two_DArray : ', type(Two_DArray))
print('Type of Three_DArray : ', type(Three_DArray))
```

```
Type of One_DArray : <class 'numpy.ndarray'>
Type of Two_DArray : <class 'numpy.ndarray'>
Type of Three_DArray : <class 'numpy.ndarray'>
```

```
In [ ]: # How to get the dimension of the array
print('One_DArray.ndim : ',One_DArray.ndim)
```

```
print('Two_DArray.ndim : ',Two_DArray.ndim)
print('Three_DArray.ndim : ',Three_DArray.ndim)
```

```
One_DArray.ndim : 1
Two_DArray.ndim : 2
Three_DArray.ndim : 3
```

```
In [ ]: # How to get the shape of the array
print('One_DArray.shape : ',One_DArray.shape)
print('Two_DArray.shape : ',Two_DArray.shape)
print('Three_DArray.shape : ',Three_DArray.shape)
```

```
One_DArray.shape : (6,)
Two_DArray.shape : (4, 3)
Three_DArray.shape : (5, 2, 3)
```

```
In [ ]: #how to get the type of the array

print('One_DArray.dtype : ', One_DArray.dtype)
print('Two_DArray.dtype : ', Two_DArray.dtype)
print('Three_DArray.dtype : ', Three_DArray.dtype)
```

```
One_DArray.dtype : float16
Two_DArray.dtype : float64
Three_DArray.dtype : float64
```

```
In [ ]: #print the size of the array

print('One_DArray.size : ',One_DArray.size)
print('Two_DArray.size : ',Two_DArray.size)
print('Three_DArray.size : ',Three_DArray.size)
```

```
One_DArray.size : 6
Two_DArray.size : 12
Three_DArray.size : 30
```

```
In [ ]: #print the size of the item array note the number is representing the size in bytes

print('One_DArray.itemsize : ',One_DArray.itemsize)
print('Two_DArray.itemsize : ',Two_DArray.itemsize)
print('Three_DArray.itemsize : ',Three_DArray.itemsize)
```

```
One_DArray.itemsize : 2
Two_DArray.itemsize : 8
Three_DArray.itemsize : 8
```

## Accessing/Changing elements, rows, columns etc

```
In [ ]: arr1 = np.array([[1,2,3,4,5,6,7],[8,9,10,11,12,13,14]])
arr1
```

```
Out [ ]: array([[ 1,  2,  3,  4,  5,  6,  7],
                [ 8,  9, 10, 11, 12, 13, 14]])
```

```
In [ ]: #get a specific element in this formate [row, column]  
arr1[1, -2]
```

```
Out[ ]: 13
```

```
In [ ]: #get a specific row  
  
arr1[1, :]
```

```
Out[ ]: array([ 8,  9, 10, 11, 12, 13, 14])
```

```
In [ ]: #get a specific column  
arr1[:, 3]
```

```
Out[ ]: array([ 4, 11])
```

```
In [ ]: #gettig a little more fancy [startIndex:endIndex:stepsize]  
  
print(arr1[0, 1:6:2])  
print(arr1[0, ::2])  
print(arr1[0, 1::2])
```

```
[2 4 6]  
[1 3 5 7]  
[2 4 6]
```

```
In [ ]: arr1[1,-1] = 50  
arr1
```

```
Out[ ]: array([[ 1,  2,  3,  4,  5,  6,  7],  
               [ 8,  9, 10, 11, 12, 13, 50]])
```

```
In [ ]: arr1[:,2] = 0  
arr1
```

```
Out[ ]: array([[ 1,  2,  0,  4,  5,  6,  7],  
               [ 8,  9,  0, 11, 12, 13, 50]])
```

```
In [ ]: arr1[1,:] = 0  
arr1
```

```
Out[ ]: array([[1, 2, 0, 4, 5, 6, 7],  
               [0, 0, 0, 0, 0, 0, 0]])
```

```
In [ ]: arr2 = np.array(  
    [[1,2],[3,4]],  
    [[5,6],[7,8]],  
    [[9,10],[11,12]],  
    [[13,14],[15,16]],  
    ])  
arr2
```

```
Out[ ]: array([[ 1,  2],
               [ 3,  4]],

              [[ 5,  6],
               [ 7,  8]],

              [[ 9, 10],
               [11, 12]],

              [[13, 14],
               [15, 16]]])
```

```
In [ ]: arr2[2,1,1]
```

```
Out[ ]: 12
```

```
In [ ]: arr2[:, :, 0]
```

```
Out[ ]: array([[ 1,  3],
               [ 5,  7],
               [ 9, 11],
               [13, 15]])
```

## Initializing all type of array

```
In [ ]: # ALL 0's arrays

arr3 = np.zeros((3,))
arr3
```

```
Out[ ]: array([0., 0., 0.])
```

```
In [ ]: np.ones(3, dtype='int32')
```

```
Out[ ]: array([1, 1, 1])
```

```
In [ ]: np.full((3,4), 45, dtype='int32')
```

```
Out[ ]: array([[45, 45, 45, 45],
               [45, 45, 45, 45],
               [45, 45, 45, 45]])
```

```
In [ ]: #random values array

np.random.rand(4,2)
```

```
Out[ ]: array([[0.2635247 , 0.98708036],
               [0.82023666, 0.55097689],
               [0.24441768, 0.32694855],
               [0.83970633, 0.68071648]])
```

```
In [ ]: #np identity matrix
```

```
np.identity(4)
```

```
Out[ ]: array([[1., 0., 0., 0.],
              [0., 1., 0., 0.],
              [0., 0., 1., 0.],
              [0., 0., 0., 1.]])
```

```
In [ ]: arr1 = np.ones((5,5))
arr1
```

```
Out[ ]: array([[1., 1., 1., 1., 1.],
              [1., 1., 1., 1., 1.],
              [1., 1., 1., 1., 1.],
              [1., 1., 1., 1., 1.],
              [1., 1., 1., 1., 1.]])
```

```
In [ ]: arr2 = np.zeros((3,3))
arr2
```

```
Out[ ]: array([[0., 0., 0.],
              [0., 0., 0.],
              [0., 0., 0.]])
```

```
In [ ]: arr2[1,1] = 9
arr2
```

```
Out[ ]: array([[0., 0., 0.],
              [0., 9., 0.],
              [0., 0., 0.]])
```

```
In [ ]: arr1[1:-1, 1:-1] = arr2
arr1
```

```
Out[ ]: array([[1., 1., 1., 1., 1.],
              [1., 0., 0., 0., 1.],
              [1., 0., 9., 0., 1.],
              [1., 0., 0., 0., 1.],
              [1., 1., 1., 1., 1.]])
```

```
In [ ]:
```