In [1]:
```
import numpy as np
```

In [2]:
```
# 1D array
a = np.array([1, 2, 3, 4, 5, 6])
a
```

Out[2]:
```
array([1, 2, 3, 4, 5, 6])
```

In [3]:
```
a.ndim
```

Out[3]:
```
1
```

In [4]:
```
# Zeros Method
np.zeros(4)
```

Out[4]:
```
array([0., 0., 0., 0.])
```

In [5]:
```
# Ones Method
np.ones(5)
```

Out[5]:
```
array([1., 1., 1., 1., 1.])
```

In [6]:
```
# empty cells method
np.empty(6) # May vary the value
```

Out[6]:
```
array([6.23042070e-307, 4.67296746e-307, 1.69121096e-306, 2.67023293e-307,
       2.67019320e-306, 2.42092166e-322])
```

In [7]:
```
# arange method
b=np.arange(20)
b
```

Out[7]:
```
array([ 0,  1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 12, 13, 14, 15, 16,
       17, 18, 19])
```

In [8]:
```
b.mean()
```

Out[8]:
```
9.5
```

In [9]:
```
# Specific range
np.arange(3,30)
```

Out[9]:
```
array([ 3,  4,  5,  6,  7,  8,  9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19,
       20, 21, 22, 23, 24, 25, 26, 27, 28, 29])
```

In [10]:
```
# Specific interval
np.arange(3,30,3)
```

Out[10]:
```
array([ 3,  6,  9, 12, 15, 18, 21, 24, 27])
```

In [11]:
```
# table
np.arange(0,55,5)
```

Out[11]:
```
array([ 0,  5, 10, 15, 20, 25, 30, 35, 40, 45, 50])
```

In [12]:
```
# Linear space
np.linspace(0,30, num= 6)
```

Out[12]:
```
array([ 0.,  6., 12., 18., 24., 30.])
```

In [13]:
```
# specifiy ur data type
np.ones(5,dtype = np.float64)
```

Out[13]:
```
array([1., 1., 1., 1., 1.])
```

Array Functions

In [14]:
```
# Sorting an array
c = np.array([10,7,37.5,48,3.3,44.5,1,3,5,0,6,4])
c.sort()
c
```

Out[14]:
```
array([ 0. ,  1. ,  3. ,  3.3,  4. ,  5. ,  6. ,  7. , 10. , 37.5, 44.5,
       48. ])
```

In [15]:
```
# Reperents the values in array
c.all()
c
```

Out[15]:
```
array([ 0. ,  1. ,  3. ,  3.3,  4. ,  5. ,  6. ,  7. , 10. , 37.5, 44.5,
       48. ])
```

In [16]:
```
# check wether array has values or not
np.any(c)
```

Out[16]:
```
True
```

In [17]:
```
d = np.array([])
np.any(d)
```

Out[17]:
```
False
```

In [18]:
```
# Returns the indices of the maximum values along an axis.
np.argmax(c)
```

Out[18]:
```
11
```

In [19]:
```
np.argmin(c)
```

Out[19]:
```
0
```

In [20]:
```
c[np.argpartition(c,8)]
```

Out[20]:
```
array([ 1. ,  0. ,  3. ,  3.3,  4. ,  5. ,  7. ,  6. , 10. , 37.5, 44.5,
       48. ])
```

In [21]:
```
x = np.array([7, 8, 9, 10,11,12])
x[np.argpartition(x, 1)]
```

In [31]:
```
z= np.array([1,2,3,4,5,6,4,5,5])
y= np.array([7,8,9,1,2,3])
z.ndim
np.concatenate((z,y))
```

Out[31]:
```
array([1, 2, 3, 4, 5, 6, 4, 5, 5, 7, 8, 9, 1, 2, 3])
```

2D arrays

In [32]:
```
a = np.array([[1,2,3,4],[5,6,7,8]])
a
```

Out[32]:
```
array([[1, 2, 3, 4],
       [5, 6, 7, 8]])
```

In [36]:
```
b = np.array([[8,7,6,5],[4,3,2,1]])
b
```

Out[36]:
```
array([[8, 7, 6, 5],
       [4, 3, 2, 1]])
```

In [37]:
```
np.concatenate((a,b))
```

Out[37]:
```
array([[1, 2, 3, 4],
       [5, 6, 7, 8],
       [8, 7, 6, 5],
       [4, 3, 2, 1]])
```

In [38]:
```
np.concatenate((a,b),axis=1)
```

Out[38]:
```
array([[1, 2, 3, 4, 8, 7, 6, 5],
       [5, 6, 7, 8, 4, 3, 2, 1]])
```

3D arrays

In [39]:
```
a = np.array([[[0, 1, 2, 3],
                          [4, 5, 6, 7]],

                         [[0, 1, 2, 3],
                          [4, 5, 6, 7]],

                         [[0 ,1 ,2, 3],
                          [4, 5, 6, 7]]])
a
```

Out[39]:
```
array([[[0, 1, 2, 3],
        [4, 5, 6, 7]],

       [[0, 1, 2, 3],
        [4, 5, 6, 7]],

       [[0, 1, 2, 3],
        [4, 5, 6, 7]]])
```

In [40]:
```
a.ndim
```

Out[40]:
```
3
```

In [43]:
```
# number of elements
a.size
```

Out[43]:
```
24
```

In [44]:
```
# Shape of an array in dimensions
a.shape
```

Out[44]:
```
(3, 2, 4)
```

In [50]:
```
a = np.arange(6)
a
```

Out[50]:
```
array([0, 1, 2, 3, 4, 5])
```

In [52]:
```
# Reshaping of arrays in different dimensions
a = a.reshape(3,2)
a
```

Out[52]:
```
array([[0, 1],
       [2, 3],
       [4, 5]])
```

In [56]:
```
d = np.arange(9)
d
```

Out[56]:
```
array([0, 1, 2, 3, 4, 5, 6, 7, 8])
```

In [63]:
```
# Some other ways of reshape of arrays
np.reshape(d,newshape=(3,3),order ="C")
```

Out[63]:
```
array([[0, 1, 2],
       [3, 4, 5],
       [6, 7, 8]])
```

In [65]:
```
#  convert 1D into 2D
f =np.arange(10)
f
```

Out[65]:
```
array([0, 1, 2, 3, 4, 5, 6, 7, 8, 9])
```

In [66]:
```
f.shape
```

Out[66]:
```
(10,)
```

In [73]:
```
# row wise 2d
d = f[np.newaxis, :]
d
```

Out[73]:
```
array([[0, 1, 2, 3, 4, 5, 6, 7, 8, 9]])
```

In [75]:
```
# column wise 2d
d = f[:,np.newaxis]
d
```

Out[75]:
```
array([[0],
       [1],
       [2],
       [3],
       [4],
       [5],
       [6],
       [7],
       [8],
       [9]])
```

In [78]:
```
a = np.arange(10)
a
```

Out[78]:
```
array([0, 1, 2, 3, 4, 5, 6, 7, 8, 9])
```

In [80]:
```
# Slicing
a[2:6]
```

Out[80]:
```
array([2, 3, 4, 5])
```

In [81]:
```
a*6
```

Out[81]:
```
array([ 0,  6, 12, 18, 24, 30, 36, 42, 48, 54])
```

In [82]:
```
a+6
```

Out[82]:
```
array([ 6,  7,  8,  9, 10, 11, 12, 13, 14, 15])
```

In [83]:
```
a.sum()
```

Out[83]:
```
45
```

In [85]:
```
a.mean()
```

Out[85]:
```
4.5
```