In [3]:
```
import numpy as np
```

##1D Array

In [6]:
```
a=np.array([5,6,7,8,9,10])
a
```

Out[6]: array([5, 6, 7, 8, 9, 10])

In [7]:
```
a.shape
```

Out[7]: (6,)

In [8]:
```
len(a)
```

Out[8]: 6

In [9]:
```
a.ndim
```

Out[9]: 1

In [10]:
```
a.size
```

Out[10]: 6

In [11]:
```
a.dtype
```

Out[11]: dtype('int32')

In [15]:
```
a1=np.zeros(5)
a1
```

Out[15]: array([0., 0., 0., 0., 0.])

In [16]:
```
#create an array of one
a2=np.ones(6)
a2
```

Out[16]: array([1., 1., 1., 1., 1., 1.])

In [18]:
```
a3=np.arange(10,30,5)
a3
```

Out[18]: array([10, 15, 20, 25])

In [19]:
```
a4=np.linspace(0,10,7)
a4
```

Out[19]:
```
array([ 0.        ,  1.66666667,  3.33333333,  5.        ,  6.66666667,
        8.33333333, 10.        ])
```

##Arithmetic Operators

Addition

In [21]:
```
a=np.array([1,2,3,4,5])
b=np.array([6,7,8,9,10])
a+b
```

Out[21]:
```
array([ 7,  9, 11, 13, 15])
```

Subtraction

In [22]:
```
a-b
```

Out[22]:
```
array([-5, -5, -5, -5, -5])
```

Multiplication

In [23]:
```
a*b
```

Out[23]:
```
array([ 6, 14, 24, 36, 50])
```

Division

In [24]:
```
a/b
```

Out[24]:
```
array([0.16666667, 0.28571429, 0.375     , 0.44444444, 0.5       ])
```

In [25]:
```
np.exp(b)
```

Out[25]:
```
array([  403.42879349,  1096.63315843,  2980.95798704,  8103.08392758,
        22026.46579481])
```

In [26]:
```
np.sqrt(b)
```

Out[26]:
```
array([2.44948974, 2.64575131, 2.82842712, 3.        , 3.16227766])
```

Comparison

In [27]:
```
a == b
```

Out[27]:
```
array([False, False, False, False, False])
```

In [28]:
```
a > b
```

Out[28]:
```
array([False, False, False, False, False])
```

In [29]:
```
a < b
```

Out[29]:
```
array([ True,  True,  True,  True,  True])
```

Aggregate Functions

In [30]:
```
a.sum()
```

Out[30]:
```
15
```

In [31]:
```
a.min()
```

Out[31]:
```
1
```

In [32]:
```
a.max()
```

Out[32]:
```
5
```

In [33]:
```
a.cumsum()
```

Out[33]:
```
array([ 1,  3,  6, 10, 15])
```

In [34]:
```
a.mean()
```

Out[34]:
```
3.0
```

In [36]:
```
#correlation cofficient
np.corrcoef(a,b)
```

Out[36]:
```
array([[1., 1.],
       [1., 1.]])
```

In [37]:
```
np.std(a)
```

Out[37]:
```
1.4142135623730951
```

2D Array

In [55]:
```
a=np.array([[7,9,3],[4,5,6]])
a
```

Out[55]:
```
array([[1, 2, 3],
       [4, 5, 6]])
```

In [56]:
```
a.shape
```

Out[56]:
```
(2, 3)
```

In [57]:
```
len(a)
```

Out[57]:
```
2
```

In [58]:
```
a.ndim
```

Out[58]:
```
2
```

In [59]:
```
a.size
```

Out[59]:
```
6
```

In [68]:
```
a.dtype
```

Out[68]:
```
dtype('int32')
```

In [80]:
```
a1=np.zeros(6)
a1
```

Out[80]:
```
array([0., 0., 0., 0., 0., 0.])
```

In [50]:
```
#create an array of one
a2=np.ones(5)
a2
```

Out[50]:
```
array([1., 1., 1., 1., 1.])
```

In [64]:
```
a3=np.arange(5,10,30)
a3
```

Out[64]:
```
array([5])
```

In [69]:
```
a4=np.linspace(0,10,8)
a4
```

Out[69]:
```
array([ 0.        ,  1.42857143,  2.85714286,  4.28571429,  5.71428571,
        7.14285714,  8.57142857, 10.        ])
```

Arithmetic Operators

Addition

In [70]:
```
a=np.array([[4,5,6],[9,8,7]])
b=np.array([[2,3,4],[5,1,9]])
a+b
```

Out[70]:
```
array([[ 6,  8, 10],
       [14,  9, 16]])
```

Subtraction

In [71]:
```
a-b
```

Out[71]:
```
array([[ 2,  2,  2],
       [ 4,  7, -2]])
```

Multiplication

In [72]:
```
a*b
```

Out[72]:
```
array([[ 8, 15, 24],
       [45,  8, 63]])
```

Division

In [74]:
```
a/b
```

Out[74]:
```
array([[2.        , 1.66666667, 1.5       ],
       [1.8       , 8.        , 0.77777778]])
```

In [75]:
```
np.exp(b)
```

Out[75]:
```
array([[7.38905610e+00, 2.00855369e+01, 5.45981500e+01],
       [1.48413159e+02, 2.71828183e+00, 8.10308393e+03]])
```

In [76]:
```
np.sqrt(b)
```

Out[76]:
```
array([[1.41421356, 1.73205081, 2.        ],
       [2.23606798, 1.        , 3.        ]])
```

Comparsion

In [77]:
```
a == b
```

Out[77]:
```
array([[False, False, False],
       [False, False, False]])
```

In [78]:
```
a > b
```

Out[78]:
```
array([[ True,  True,  True],
       [ True,  True, False]])
```

In [79]:
```
a < b
```

Out[79]:
```
array([[False, False, False],
       [False, False,  True]])
```

Aggregate Functions

In [81]:
```
a.sum()
```

Out[81]:
```
39
```

In [82]:
```
a.min()
```

Out[82]:
```
4
```

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

Out[83]:
```
9
```

In [84]:
```
a.cumsum()
```

Out[84]:
```
array([ 4,  9, 15, 24, 32, 39])
```

In [111]:
```
a.mean()
```

Out[111]:
```
5.0
```

In [86]:
```
#correlation cofficient
np.corrcoef(a,b)
```

Out[86]:
```
array([[ 1. , -1. ,  1. ,  0.5],
       [-1. ,  1. , -1. , -0.5],
       [ 1. , -1. ,  1. ,  0.5],
       [ 0.5, -0.5,  0.5,  1. ]])
```

In [87]:
```
np.std(a)
```

Out[87]:
```
1.707825127659933
```

3D Array

In [88]:
```
a=np.array([[[1,2,3],[4,5,6],[7,8,9]]])
a
```

Out[88]:
```
array([[[1, 2, 3],
        [4, 5, 6],
        [7, 8, 9]]])
```

In [89]:
```
a.shape
```

Out[89]:
```
(1, 3, 3)
```

In [90]:
```
a.size
```

Out[90]:
```
9
```

In [91]:
```
len(a)
```

Out[91]:
```
1
```

In [92]:
```
a.ndim
```

Out[92]:
```
3
```

In [93]:
```
a.dtype
```

Out[93]:
```
dtype('int32')
```

In [94]:
```
a1=np.zeros(6)
a1
```

Out[94]:
```
array([0., 0., 0., 0., 0., 0.])
```

In [95]:
```
#create an array of one
a2=np.ones(5)
a2
```

Out[95]:
```
array([1., 1., 1., 1., 1.])
```

In [96]:
```
a3=np.arange(5,10,30)
a3
```

Out[96]:
```
array([5])
```

In [97]:
```
a4=np.linspace(0,10,8)
a4
```

Out[97]:
```
array([ 0.        ,  1.42857143,  2.85714286,  4.28571429,  5.71428571,
        7.14285714,  8.57142857, 10.        ])
```

In [ ]:
```

```

Arithmetic Operators

Addition

In [98]:
```
a=np.array([[[1,2,3],[4,5,6],[7,8,9]]])
b=np.array([[[4,5,6],[7,8,9],[1,2,3]]])
a+b
```

Out[98]:
```
array([[[ 5,  7,  9],
        [11, 13, 15],
        [ 8, 10, 12]]])
```

Subtraction

In [99]:
```
a-b
```

Out[99]:
```
array([[[-3, -3, -3],
        [-3, -3, -3],
        [ 6,  6,  6]]])
```

Multiplication

In [100]:
```
a*b
```

Out[100]:
```
array([[[ 4, 10, 18],
        [28, 40, 54],
        [ 7, 16, 27]]])
```

Division

In [101]:
```
a/b
```

Out[101]:
```
array([[[0.25      , 0.4       , 0.5       ],
        [0.57142857, 0.625     , 0.66666667],
        [7.        , 4.        , 3.        ]]])
```

In [102]:
```
np.exp(b)
```

Out[102]:
```
array([[[5.45981500e+01, 1.48413159e+02, 4.03428793e+02],
        [1.09663316e+03, 2.98095799e+03, 8.10308393e+03],
        [2.71828183e+00, 7.38905610e+00, 2.00855369e+01]]])
```

In [103]:
```
np.sqrt(b)
```

Out[103]:
```
array([[[2.        , 2.23606798, 2.44948974],
        [2.64575131, 2.82842712, 3.        ],
        [1.        , 1.41421356, 1.73205081]]])
```

Comparsion

In [104]:
```
a == b
```

Out[104]:
```
array([[[False, False, False],
        [False, False, False],
        [False, False, False]]])
```

In [105]:
```
a < b
```

Out[105]:
```
array([[[ True,  True,  True],
        [ True,  True,  True],
        [False, False, False]]])
```

In [106]:
```
a > b
```

Out[106]:
```
array([[[False, False, False],
        [False, False, False],
        [ True,  True,  True]]])
```

Aggregate Functions

In [107]:
```
a.sum()
```

Out[107]: 45

In [108]:
```
a.min()
```

Out[108]: 1

In [110]:
```
a.max()
```

Out[110]: 9

In [112]:
```
a.cumsum()
```

Out[112]:
```
array([ 1,  3,  6, 10, 15, 21, 28, 36, 45])
```

In [113]:
```
a.mean()
```

Out[113]:
```
5.0
```

In [115]:
```
#correlation cofficient
np.corrcoef(a,b)
```

```
---------------------------------------------------------------------
--
ValueError                                Traceback (most recent call las
t)
Cell In[115], line 2
      1 #correlation cofficient
----> 2 np.corrcoef(a,b)

File <__array_function__ internals>:200, in corrcoef(*args, **kwargs)

File ~\AppData\Local\Programs\Python\Python310\lib\site-packages\numpy\li
b\function_base.py:2846, in corrcoef(x, y, rowvar, bias, ddof, dtype)
   2842 if bias is not np._NoValue or ddof is not np._NoValue:
   2843     # 2015-03-15, 1.10
   2844     warnings.warn('bias and ddof have no effect and are deprecate
d',
   2845                   DeprecationWarning, stacklevel=3)
-> 2846 c = cov(x, y, rowvar, dtype=dtype)
   2847 try:
   2848     d = diag(c)

File <__array_function__ internals>:200, in cov(*args, **kwargs)

File ~\AppData\Local\Programs\Python\Python310\lib\site-packages\numpy\li
b\function_base.py:2618, in cov(m, y, rowvar, bias, ddof, fweights, aweig
hts, dtype)
   2616 m = np.asarray(m)
   2617 if m.ndim > 2:
-> 2618     raise ValueError("m has more than 2 dimensions")
   2620 if y is not None:
   2621     y = np.asarray(y)

ValueError: m has more than 2 dimensions
```

In [116]:
```
np.std(a)
```

Out[116]:
```
2.581988897471611
```

In [ ]:
```

```