Numpy

Numpy

- Numpy stands for Numerical Python.
- It is a python library used for working with an array.
- Array is core component of Numpy.

Numpy Vs List

- 1.Numpy Occupies Less Memory
- 2.Numpy is faster than list
- 3.Numpy is more felxible than list

Numpy Occupies Less Memory

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

[2]:
```
# Numpy
ary1 = np.arange(1,10,1)
ary1
```

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

[17]:
```
print("shape of ary1     :- ",ary1.shape)
print("dimension of ary1 :- ",ary1.ndim)
print("Type of ary1      :- ",type(ary1))
print("dtype of ary1     :- ",ary1.dtype)
print("size of ary1      :- ",ary1.size)
print("itemsize of ary1  :- ",ary1.itemsize)
```

```
shape of ary1     :-  (9,)
dimension of ary1 :-  1
Type of ary1      :-  <class 'numpy.ndarray'>
dtype of ary1     :-  int32
size of ary1      :-  9
itemsize of ary1  :-  4
```

[14]:
```
print("Memory of ary1    :- ",ary1.size*ary1.itemsize)
```

```
Memory of ary1    :-  36
```

[9]:
```
# List
nlist=list(range(1,10))
nlist
```

[9]:
```
[1, 2, 3, 4, 5, 6, 7, 8, 9]
```

[10]:
```
print("type of nlist      :- ",type(nlist))
```

```
type of nlist      :-  <class 'list'>
```

[18]:
```
import sys
sys.getsizeof(8)
```

[18]:
```
28
```

[21]:
```
print("Memory of nlist  :- ",len(nlist)*sys.getsizeof(8))
```

```
Memory of nlist  :-  252
```

Numpy Is Faster Than List

[23]:
```
num=100000
list1=list(range(num))
list2=list(range(num))
```

[24]:
```
#Time taken for list operation
import time
start_time=time.time()
reslist=[]
for x in range(100000):
    reslist.append(list1[x]+list2[x])
end_time=time.time()
print("total time:",(end_time-start_time)*10000)
```

```
total time: 552.8879165649414
```

[27]:
```
a1=np.arange(num)
a2=np.arange(num)
```

[30]:
```
#Time taken for Array operation
start=time.time()
res=a1+a2
end=time.time()
print(res.size)
print("time taken for array:",(end-start)*10000)
```

```
100000
time taken for array: 9.980201721191406
```

[ ]:

```
```

[ ]:

```
```

How Many Ways To Create Numpy Array

Five ways to create Numpy Array

- Traditional way
- Homogeneous way
- Diagonal way
- Numerical way
- Random way

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

1.Traditional way

```
[2]: #Array()
     np.array([10,20,30,40])
```

```
[2]: array([10, 20, 30, 40])
```

```
[3]: # List to array
     list1=[1,1,1,1,1]
     np.array(list1)
```

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

```
[4]: # Variables
     a=np.array([12,43,26,54])
     a
```

```
[4]: array([12, 43, 26, 54])
```

```
[5]: a.dtype
```

```
[5]: dtype('int32')
```

```
[6]: a.shape
```

```
[6]: (4,)
```

```
[7]: a.ndim
```

```
[7]: 1
```

```
[8]: a.size
```

```
[8]: 4
```

```
[9]: a.itemsize
```

```
[9]: 4
```

```
[10]: a=np.array([12,43,26,54])
      r=a.astype("float32")
      r
```

```
[10]: array([12., 43., 26., 54.], dtype=float32)
```

```
[11]: r.dtype
```

```
[11]: dtype('float32')
```

2.Homogeneous

- Ones
- Zeros

```
[12]: a1=np.ones((4,4))
```

```
[13]: a1
```

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

```
[14]: ones=np.ones((4, 5))
      ones
```

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

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

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

3.Diagonal way

- Eye
- Identify

[16]:
```
# Eye
np.eye(4,4)
```

[16]:
```
array([[1., 0., 0., 0.],
       [0., 1., 0., 0.],
       [0., 0., 1., 0.],
       [0., 0., 0., 1.]])
```

[17]:
```
np.eye(4,6,k=1)
```

[17]:
```
array([[0., 1., 0., 0., 0., 0.],
       [0., 0., 1., 0., 0., 0.],
       [0., 0., 0., 1., 0., 0.],
       [0., 0., 0., 0., 1., 0.]])
```

[18]:
```
# Identity
a=np.identity(4)
```

[19]:
```
a
```

[19]:
```
array([[1., 0., 0., 0.],
       [0., 1., 0., 0.],
       [0., 0., 1., 0.],
       [0., 0., 0., 1.]])
```

[20]:
```
r1=a.astype("int")
```

[21]:
```
r1
```

[21]:
```
array([[1, 0, 0, 0],
       [0, 1, 0, 0],
       [0, 0, 1, 0],
       [0, 0, 0, 1]])
```

[22]:
```
r1.dtype
```

[22]:
```
dtype('int32')
```

[23]:
```
a.dtype
```

[23]:
```
dtype('float64')
```

[24]:
```
a.itemsize
```

```
[24]: 8
```

4.Numerical way

- Arrange
- linspace

```
[25]: # Arrange
      a=np.arange(1,10,1)
      a
```

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

```
[26]: a.shape
```

```
[26]: (9,)
```

```
[27]: a.reshape(3,3)
```

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

```
[28]: a.reshape(1,9)
```

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

```
[29]: # Linspace
      a1=np.linspace(1,10,40)
```

```
[30]: a1
```

```
[30]: array([ 1.        ,  1.23076923,  1.46153846,  1.69230769,  1.92307692,
              2.15384615,  2.38461538,  2.61538462,  2.84615385,  3.07692308,
              3.30769231,  3.53846154,  3.76923077,  4.        ,  4.23076923,
              4.46153846,  4.69230769,  4.92307692,  5.15384615,  5.38461538,
              5.61538462,  5.84615385,  6.07692308,  6.30769231,  6.53846154,
              6.76923077,  7.        ,  7.23076923,  7.46153846,  7.69230769,
              7.92307692,  8.15384615,  8.38461538,  8.61538462,  8.84615385,
              9.07692308,  9.30769231,  9.53846154,  9.76923077, 10.        ])
```

```
[31]: a1.shape
```

```
[31]: (40,)
```

```
[32]: a1.ndim
```

```
[32]: 1
```

```
[33]: a1.size
```

```
[33]: 40
```

```
[34]: a1.dtype
```

```
[34]: dtype('float64')
```

5.Random way

- Random.rand
- Random.randn
- Random.randint
- Random.choice

```
[35]: # 1.random.rand -- > its generates only positive float values
      rand=np.random.rand(1,5)
      rand
```

```
[35]: array([[0.65300178, 0.8524893 , 0.29683314, 0.80032111, 0.42222435]])
```

```
[36]: # 2.random.randn ---> it generates positive and negative float values
      randn=np.random.randn(1,10)
      randn
```

```
[36]: array([[ 0.2198343 , -1.10560589,  0.73114871,  0.64197329,  0.07768435,
               1.41193857,  1.04253933, -2.0359504 ,  0.25035268, -0.08393944]])
```

```
[37]: #3.random.randint ---> it generates all positive integer values only
      randint=np.random.randint(1,10,(2,4))
      randint
```

```
[37]: array([[4, 4, 1, 8],
             [6, 8, 8, 5]])
```

```
[38]: # 4.random choice
      nlist=[1,2,4,5,6]
      random=np.random.choice(nlist,(3,4),p=(0.0,0.1,0.2,0.1,0.6))
      random
```

```
[38]: array([[4, 6, 6, 6],
             [2, 6, 6, 6],
             [6, 4, 6, 2]])
```

```
[ ]:
```

Multi Dimension Indexing And Slicing

One Dimensional Array Indexing

[1]:
```
import numpy as np
ary1=np.random.randint(1,40,(10))
ary1
```

[1]:
```
array([26,  9, 32, 34,  2, 21,  5,  2,  6, 28])
```

[3]:
```
ary1.shape
```

[3]:
```
(10,)
```

[4]:
```
ary1.ndim
```

[4]:
```
1
```

[5]:
```
ary1.size
```

[5]:
```
10
```

[6]:
```
ary1[2]
```

[6]:
```
32
```

Two Dimensional Indexing

[8]:
```
ary2=np.random.randint(1,35,(4,6))
ary2
```

[8]:
```
array([[20,  3,  1, 13, 28, 21],
       [21, 19, 24, 27, 25, 19],
       [27, 15,  8, 17, 20, 15],
       [ 5, 14, 34, 19, 21, 30]])
```

[9]:
```
ary2[1][3]
```

```
[9]: 27
```

Three Dimensional Indeixing

```
[10]: ary3=np.random.randint(10,35,(2,4,5))
      ary3
```

```
[10]: array([[[23, 19, 10, 31, 24],
              [18, 34, 25, 16, 22],
              [32, 13, 18, 25, 10],
              [21, 25, 25, 12, 13]],

             [[17, 32, 32, 24, 24],
              [23, 10, 24, 11, 18],
              [12, 19, 29, 18, 30],
              [10, 19, 25, 24, 11]]])
```

```
[18]: ary3[1][2][3]
```

```
[18]: 22
```

Slicing Of One Dimensional Array

```
[12]: ary1
```

```
[12]: array([26,  9, 32, 34,  2, 21,  5,  2,  6, 28])
```

```
[13]: ary1[2:5]
```

```
[13]: array([32, 34,  2])
```

Slicing Of Two Dimensional Array

```
[14]: ary2
```

```
[14]: array([[20,  3,  1, 13, 28, 21],
             [21, 19, 24, 27, 25, 19],
             [27, 15,  8, 17, 20, 15],
             [ 5, 14, 34, 19, 21, 30]])
```

```
[15]: ary2[1:4,1:4]
```

```
[15]: array([[19, 24, 27],
             [15,  8, 17],
       [14, 34, 19]])
```

Slicing Of Three Dimensional Array

```
[16]: ary3
```

```
[16]: array([[[23, 19, 10, 31, 24],
              [18, 34, 25, 16, 22],
              [32, 13, 18, 25, 10],
              [21, 25, 25, 12, 13]],

             [[17, 32, 32, 24, 24],
              [23, 10, 24, 11, 18],
              [12, 19, 29, 18, 30],
              [10, 19, 25, 24, 11]]])
```

```
[19]: ary3[1:,1:4,1:4]
```

```
[19]: array([[[10, 24, 11],
              [19, 29, 18],
              [19, 25, 24]]])
```

```
[ ]:
```

Boardcasting Rules

Boardcasting Rules

- Case :-1 shape is same and dimension is same operation is perform on both

```
[1]: import numpy as np
     a1=np.random.randint(1,20,(2,2))
```

```
[2]: a1
```

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

```
[3]: a2=np.random.randint(21,40,(2,2))
```

```
[4]: a2
```

```
[4]: array([[22, 35],
            [33, 34]])
```

```
[5]: a1+a2
```

```
[5]: array([[27, 39],
            [36, 35]])
```

```
[6]: a1.shape
```

```
[6]: (2, 2)
```

```
[7]: a2.ndim
```

```
[7]: 2
```

Boardcasting Rules

- Case :-2 shape is different and dimensional is same

```
[8]: ary3=np.random.randint(21,40,(2,4,5))
```

```
[9]: ary3
```

```
[9]: array([[[34, 31, 30, 38, 23],
             [38, 26, 31, 31, 39],
             [26, 24, 30, 38, 36],
             [37, 23, 38, 29, 29]],

            [[25, 29, 26, 24, 34],
             [23, 27, 33, 26, 27],
             [21, 36, 26, 26, 32],
             [31, 31, 36, 35, 22]]])
```

```
[10]: ary3.shape
```

```
[10]: (2, 4, 5)
```

```
[11]: ary3.ndim
```

```
[11]: 3
```

```
[12]: ary4=np.random.randint(21,40,(2,4,1))
```

```
[13]: ary4
```

```
[13]: array([[[23],
              [36],
              [21],
              [25]],

             [[28],
              [28],
              [29],
              [22]]])
```

```
[14]: ary4.shape
```

```
[14]: (2, 4, 1)
```

```
[15]: ary4.ndim
```

```
[15]: 3
```

```
[16]: ary3+ary4
```

```
[16]: array([[[57, 54, 53, 61, 46],
              [74, 62, 67, 67, 75],
              [47, 45, 51, 59, 57],
              [62, 48, 63, 54, 54]],

             [[53, 57, 54, 52, 62],
              [51, 55, 61, 54, 55],
              [50, 65, 55, 55, 61],
              [53, 53, 58, 57, 44]]])
```

```
[17]: ary3
```

```
[17]: array([[[34, 31, 30, 38, 23],
         [38, 26, 31, 31, 39],
         [26, 24, 30, 38, 36],
         [37, 23, 38, 29, 29]],

        [[25, 29, 26, 24, 34],
         [23, 27, 33, 26, 27],
         [21, 36, 26, 26, 32],
         [31, 31, 36, 35, 22]]])
```

```
[18]: ary5=np.random.randint(21,40,(5,4))
      ary5
```

```
[18]: array([[32, 38, 22, 38],
        [39, 26, 29, 30],
        [38, 38, 38, 21],
        [22, 37, 30, 34],
        [35, 28, 39, 30]])
```

```
[19]: ary5.shape
```

```
[19]: (5, 4)
```

```
[20]: ary5.ndim
```

```
[20]: 2
```

Dimension different and shape aslo different

```
[21]: ary4+ary5
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
Cell In[21], line 1
----> 1 ary4+ary5
```

```
ValueError: operands could not be broadcast together with shapes (2,4,1) (5,4)
```

```
[ ]:
```

Boolean Indexing Numpy

Boolean Indexing

[1]:
```
import numpy as np
ary1=np.random.randint(10,300,(6,10))
```

[2]:
```
ary1
```

[2]:
```
array([[ 24, 276, 224, 106,  19, 224, 168,  43,  92,  88],
       [131,  78, 164,  67, 165,  24, 117,  97, 166,  87],
       [226, 105, 181,  29, 196,  74, 127, 280, 282,  12],
       [163, 201,  48,  61, 100,  14, 106, 100, 178, 204],
       [246,  23, 172, 184,  26,  16, 274,  16, 204, 203],
       [ 47,  93, 119, 162,  33, 202,  50, 199,  33, 282]])
```

[4]:
```
ary1 > 100
```

[4]:
```
array([[False,  True,  True,  True, False,  True,  True, False, False,
        False],
       [ True, False,  True, False,  True, False,  True, False,  True,
        False],
       [ True,  True,  True, False,  True, False,  True,  True,  True,
        False],
       [ True,  True, False, False, False, False,  True, False,  True,
         True],
       [ True, False,  True,  True, False, False,  True, False,  True,
         True],
       [False, False,  True,  True, False,  True, False,  True, False,
         True]])
```

[5]:
```
ary1 <100
```

[5]:
```
array([[ True, False, False, False,  True, False, False,  True,  True,
         True],
       [False,  True, False,  True, False,  True, False,  True, False,
         True],
       [False, False, False,  True, False,  True, False, False, False,
         True],
       [False, False,  True,  True, False,  True, False, False, False,
        False],
       [False,  True, False, False,  True,  True, False,  True, False,
        False],
       [ True,  True, False, False,  True, False,  True, False,  True,
        False]])
```

```
[9]: ary1[ary1 <100 ]
```

```
[9]: array([24, 19, 43, 92, 88, 78, 67, 24, 97, 87, 29, 74, 12, 48, 61, 14, 23,
            26, 16, 16, 47, 93, 33, 50, 33])
```

```
[11]: (ary1>100) & (ary1 <200)
```

```
[11]: array([[ True, False, False, False,  True, False, False,  True,  True,
               True],
             [False,  True, False,  True, False,  True, False,  True, False,
               True],
             [False, False, False,  True, False,  True, False, False, False,
               True],
             [False, False,  True,  True, False,  True, False, False, False,
              False],
             [False,  True, False, False,  True,  True, False,  True, False,
              False],
             [ True,  True, False, False,  True, False,  True, False,  True,
              False]])
```

```
[13]: ary1[(ary1>100) & (ary1 <200) ]
```

```
[13]: array([106, 168, 131, 164, 165, 117, 166, 105, 181, 196, 127, 163, 106,
             178, 172, 184, 119, 162, 199])
```

Numpy Array Transformation

1. Reshape

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

```
[2]: a1=np.random.randint(10,50,(20,))
     a1
```

```
[2]: array([48, 23, 37, 11, 40, 47, 38, 45, 20, 23, 25, 17, 11, 23, 19, 24, 14,
            14, 18, 27])
```

```
[3]: a1.dtype
```

```
[3]: dtype('int32')
```

```
[4]: type(a1)
```

```
[4]: numpy.ndarray
```

```
[5]: a1.size
```

```
[5]: 20
```

```
[6]: a1=a1.reshape(5,4)
     a1
```

```
[6]: array([[48, 23, 37, 11],
            [40, 47, 38, 45],
            [20, 23, 25, 17],
            [11, 23, 19, 24],
            [14, 14, 18, 27]])
```

```
[7]: a1.reshape(1,20)
```

```
[7]: array([[48, 23, 37, 11, 40, 47, 38, 45, 20, 23, 25, 17, 11, 23, 19, 24,
             14, 14, 18, 27]])
```

```
[8]: a1
```

```
[8]: array([[48, 23, 37, 11],
            [40, 47, 38, 45],
            [20, 23, 25, 17],
            [11, 23, 19, 24],
            [14, 14, 18, 27]])
```

```
[9]: a1.reshape(2,2,5)
```

```
[9]: array([[[48, 23, 37, 11, 40],
             [47, 38, 45, 20, 23]],

            [[25, 17, 11, 23, 19],
             [24, 14, 14, 18, 27]]])
```

2. Resize

```
[10]: a1.size
```

```
[10]: 20
```

```
[11]: a1=a1.astype("int")
```

```
[12]: a1
```

```
[12]: array([[48, 23, 37, 11],
             [40, 47, 38, 45],
             [20, 23, 25, 17],
             [11, 23, 19, 24],
             [14, 14, 18, 27]])
```

```
[13]: a1.resize(5,6,refcheck = False)
```

```
[14]: a1
```

```
[14]: array([[48, 23, 37, 11, 40, 47],
             [38, 45, 20, 23, 25, 17],
             [11, 23, 19, 24, 14, 14],
             [18, 27,  0,  0,  0,  0],
             [ 0,  0,  0,  0,  0,  0]])
```

```
[15]: a1.shape
```

```
[15]: (5, 6)
```

```
[16]: a1.size
```

```
[16]: 30
```

```
[17]: a1=np.resize(a1,(6,8))
```

```
[18]: a1.size
```

```
[18]: 48
```

```
[19]: a1
```

```
[19]: array([[48, 23, 37, 11, 40, 47, 38, 45],
              [20, 23, 25, 17, 11, 23, 19, 24],
              [14, 14, 18, 27,  0,  0,  0,  0],
              [ 0,  0,  0,  0,  0,  0, 48, 23],
              [37, 11, 40, 47, 38, 45, 20, 23],
              [25, 17, 11, 23, 19, 24, 14, 14]])
```

3. Flatten

```
[20]: a1
```

```
[20]: array([[48, 23, 37, 11, 40, 47, 38, 45],
              [20, 23, 25, 17, 11, 23, 19, 24],
              [14, 14, 18, 27,  0,  0,  0,  0],
              [ 0,  0,  0,  0,  0,  0, 48, 23],
              [37, 11, 40, 47, 38, 45, 20, 23],
              [25, 17, 11, 23, 19, 24, 14, 14]])
```

```
[21]: a1.flatten()
```

```
[21]: array([48, 23, 37, 11, 40, 47, 38, 45, 20, 23, 25, 17, 11, 23, 19, 24, 14,
             14, 18, 27,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0, 48, 23, 37, 11,
             40, 47, 38, 45, 20, 23, 25, 17, 11, 23, 19, 24, 14, 14])
```

```
[22]: a1.flatten(order = "C") # Row wise order
```

```
[22]: array([48, 23, 37, 11, 40, 47, 38, 45, 20, 23, 25, 17, 11, 23, 19, 24, 14,
             14, 18, 27,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0, 48, 23, 37, 11,
             40, 47, 38, 45, 20, 23, 25, 17, 11, 23, 19, 24, 14, 14])
```

```
[23]: a1.flatten(order = "F") # Cloumn wise order
```

```
[23]: array([48, 20, 14,  0, 37, 25, 23, 23, 14,  0, 11, 17, 37, 25, 18,  0, 40,
             11, 11, 17, 27,  0, 47, 23, 40, 11,  0,  0, 38, 19, 47, 23,  0,  0,
             45, 24, 38, 19,  0, 48, 20, 14, 45, 24,  0, 23, 23, 14])
```

4. Ravel

```
[24]: a1
```

```
[24]: array([[48, 23, 37, 11, 40, 47, 38, 45],
             [20, 23, 25, 17, 11, 23, 19, 24],
             [14, 14, 18, 27,  0,  0,  0,  0],
             [ 0,  0,  0,  0,  0,  0, 48, 23],
             [37, 11, 40, 47, 38, 45, 20, 23],
             [25, 17, 11, 23, 19, 24, 14, 14]])
```

```
[25]: a1.ravel()
```

```
[25]: array([48, 23, 37, 11, 40, 47, 38, 45, 20, 23, 25, 17, 11, 23, 19, 24, 14,
             14, 18, 27,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0, 48, 23, 37, 11,
             40, 47, 38, 45, 20, 23, 25, 17, 11, 23, 19, 24, 14, 14])
```

```
[26]: np.ravel(a1,order="C")
```

```
[26]: array([48, 23, 37, 11, 40, 47, 38, 45, 20, 23, 25, 17, 11, 23, 19, 24, 14,
             14, 18, 27,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0, 48, 23, 37, 11,
             40, 47, 38, 45, 20, 23, 25, 17, 11, 23, 19, 24, 14, 14])
```

```
[27]: np.ravel(a1,order="F")
```

```
[27]: array([48, 20, 14,  0, 37, 25, 23, 23, 14,  0, 11, 17, 37, 25, 18,  0, 40,
             11, 11, 17, 27,  0, 47, 23, 40, 11,  0,  0, 38, 19, 47, 23,  0,  0,
             45, 24, 38, 19,  0, 48, 20, 14, 45, 24,  0, 23, 23, 14])
```

5. Transpose

```
[28]: ary1=np.random.randint(10,50,(4,8))
```

```
[29]: ary1
```

```
[29]: array([[49, 13, 45, 41, 30, 30, 45, 29],
             [17, 32, 30, 39, 40, 47, 14, 35],
             [28, 12, 45, 15, 15, 40, 35, 28],
             [12, 26, 43, 17, 23, 14, 34, 48]])
```

```
[30]: ary1.shape
```

```
[30]: (4, 8)
```

```
[31]: c1=ary1.transpose()
```

```
[32]: c1
```

```
[32]: array([[49, 17, 28, 12],
              [13, 32, 12, 26],
              [45, 30, 45, 43],
              [41, 39, 15, 17],
              [30, 40, 15, 23],
              [30, 47, 40, 14],
              [45, 14, 35, 34],
              [29, 35, 28, 48]])
```

```
[33]: c1.shape
```

```
[33]: (8, 4)
```

```
[34]: ary1.T
```

```
[34]: array([[49, 17, 28, 12],
              [13, 32, 12, 26],
              [45, 30, 45, 43],
              [41, 39, 15, 17],
              [30, 40, 15, 23],
              [30, 47, 40, 14],
              [45, 14, 35, 34],
              [29, 35, 28, 48]])
```

```
[35]: ary2=np.random.randint(10,50,(3,4,6))
```

```
[36]: ary2
```

```
[36]: array([[[34, 31, 20, 26, 49, 42],
               [16, 47, 14, 39, 16, 37],
               [37, 23, 42, 36, 25, 22],
               [29, 26, 45, 30, 24, 32]],

              [[17, 22, 39, 11, 31, 29],
               [29, 14, 14, 47, 14, 17],
               [23, 26, 39, 44, 24, 41],
               [40, 15, 30, 11, 44, 40]],

              [[28, 41, 25, 19, 30, 13],
               [37, 34, 23, 32, 36, 43],
               [30, 29, 37, 21, 41, 16],
               [11, 24, 44, 10, 19, 45]]])
```

```
[37]: ary2.T .shape#reverse the shape
```

```
[37]: (6, 4, 3)
```

```
[38]: ary2.shape
```

```
[38]: (3, 4, 6)
```

```
[39]: ary2.transpose(1,2,0)
```

```
[39]: array([[[34, 17, 28],
                [31, 22, 41],
                [20, 39, 25],
                [26, 11, 19],
                [49, 31, 30],
                [42, 29, 13]],

               [[16, 29, 37],
                [47, 14, 34],
                [14, 14, 23],
                [39, 47, 32],
                [16, 14, 36],
                [37, 17, 43]],

               [[37, 23, 30],
                [23, 26, 29],
                [42, 39, 37],
                [36, 44, 21],
                [25, 24, 41],
                [22, 41, 16]],

               [[29, 40, 11],
                [26, 15, 24],
                [45, 30, 44],
                [30, 11, 10],
                [24, 44, 19],
                [32, 40, 45]]])
```

```
[40]: ary2.transpose(1,2,0).shape
```

```
[40]: (4, 6, 3)
```

```
[41]: ary3=np.random.randint(10,50,(2,3,5,6))
      ary3
```

```
[41]: array([[[[24, 23, 21, 49, 12, 29],
                 [33, 19, 27, 30, 41, 20],
                 [15, 40, 11, 45, 25, 13],
                 [28, 12, 33, 43, 13, 30],
                 [48, 18, 14, 19, 35, 46]],

                [[41, 24, 43, 17, 37, 19],
                 [26, 29, 22, 17, 36, 13],
                 [34, 10, 11, 45, 20, 37],
                 [16, 31, 24, 12, 42, 17],
                 [36, 40, 14, 17, 20, 32]],

                [[24, 31, 25, 11, 31, 34],
                 [45, 35, 11, 20, 44, 41],
                 [37, 25, 32, 17, 23, 15],
                 [43, 14, 33, 31, 18, 37],
                 [10, 19, 33, 31, 26, 35]]],


               [[[16, 38, 32, 33, 28, 14],
                 [16, 25, 39, 24, 31, 24],
                 [44, 26, 34, 25, 48, 25],
                 [36, 21, 47, 17, 31, 40],
                 [10, 34, 36, 35, 27, 34]],

                [[45, 23, 47, 12, 49, 36],
                 [22, 46, 43, 39, 40, 27],
                 [36, 39, 32, 10, 28, 25],
                 [39, 34, 12, 41, 38, 26],
                 [36, 18, 31, 40, 18, 22]],

                [[27, 14, 27, 20, 19, 28],
                 [19, 15, 44, 33, 13, 42],
                 [36, 41, 22, 47, 33, 39],
                 [26, 24, 30, 23, 17, 49],
                 [33, 26, 13, 39, 41, 13]]]])
```

```
[42]: ary3.transpose(2,0,3,1)
```

```
[42]: array([[[[24, 41, 24],
         [23, 24, 31],
         [21, 43, 25],
         [49, 17, 11],
         [12, 37, 31],
         [29, 19, 34]],

        [[16, 45, 27],
         [38, 23, 14],
         [32, 47, 27],
         [33, 12, 20],
         [28, 49, 19],
         [14, 36, 28]]],


       [[[33, 26, 45],
         [19, 29, 35],
         [27, 22, 11],
         [30, 17, 20],
         [41, 36, 44],
         [20, 13, 41]],

        [[16, 22, 19],
         [25, 46, 15],
         [39, 43, 44],
         [24, 39, 33],
         [31, 40, 13],
         [24, 27, 42]]],


       [[[15, 34, 37],
         [40, 10, 25],
         [11, 11, 32],
         [45, 45, 17],
         [25, 20, 23],
         [13, 37, 15]],

        [[44, 36, 36],
         [26, 39, 41],
         [34, 32, 22],
         [25, 10, 47],
         [48, 28, 33],
         [25, 25, 39]]],


       [[[28, 16, 43],
         [12, 31, 14],
         [33, 24, 33],
         [43, 12, 31],
         [13, 42, 18],
         [30, 17, 37]],

        [[36, 39, 26],
         [21, 34, 24],
         [47, 12, 30],
         [17, 41, 23],
         [31, 38, 17],
         [40, 26, 49]]],
       [[[48, 36, 10],
         [18, 40, 19],
         [14, 14, 33],
         [19, 17, 31],
         [35, 20, 26],
         [46, 32, 35]],

        [[10, 36, 33],
         [34, 18, 26],
         [36, 31, 13],
         [35, 40, 39],
         [27, 18, 41],
         [34, 22, 13]]]])
```

```
[43]: ary3.transpose(2,0,3,1).shape
```

```
[43]: (5, 2, 6, 3)
```

6. Swap Axes

```
[44]: b1=np.random.randint(10,60,(4,5))
```

```
[45]: b1
```

```
[45]: array([[16, 24, 29, 38, 42],
              [48, 18, 15, 13, 14],
              [35, 42, 57, 58, 55],
              [49, 18, 29, 49, 22]])
```

```
[46]: b1.shape
```

```
[46]: (4, 5)
```

```
[47]: b1.swapaxes(1,0)
```

```
[47]: array([[16, 48, 35, 49],
              [24, 18, 42, 18],
              [29, 15, 57, 29],
              [38, 13, 58, 49],
              [42, 14, 55, 22]])
```

```
[48]: b1.swapaxes(1,0).shape
```

```
[48]: (5, 4)
```

[49]:
```
ary3.shape
```

[49]:
```
(2, 3, 5, 6)
```

[50]:
```
ary3.swapaxes(1,0).shape
```

[50]:
```
(3, 2, 5, 6)
```

7. Flip

[51]:
```
b1
```

[51]:
```
array([[16, 24, 29, 38, 42],
       [48, 18, 15, 13, 14],
       [35, 42, 57, 58, 55],
       [49, 18, 29, 49, 22]])
```

[52]:
```
b1.shape
```

[52]:
```
(4, 5)
```

[53]:
```
np.flip(b1)
```

[53]:
```
array([[22, 49, 29, 18, 49],
       [55, 58, 57, 42, 35],
       [14, 13, 15, 18, 48],
       [42, 38, 29, 24, 16]])
```

[54]:
```
np.fliplr(b1)
```

[54]:
```
array([[42, 38, 29, 24, 16],
       [14, 13, 15, 18, 48],
       [55, 58, 57, 42, 35],
       [22, 49, 29, 18, 49]])
```

[55]:
```
np.flipud(b1)
```

[55]:
```
array([[49, 18, 29, 49, 22],
       [35, 42, 57, 58, 55],
       [48, 18, 15, 13, 14],
       [16, 24, 29, 38, 42]])
```

[56]:
```
np.flip(b1,axis=0)
```

[56]:
```
array([[49, 18, 29, 49, 22],
       [35, 42, 57, 58, 55],
       [48, 18, 15, 13, 14],
       [16, 24, 29, 38, 42]])
```

```
[57]: np.flip(b1,axis=1)
```

```
[57]: array([[42, 38, 29, 24, 16],
             [14, 13, 15, 18, 48],
             [55, 58, 57, 42, 35],
             [22, 49, 29, 18, 49]])
```

Numpy-Functions

np.Where()

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

```
[2]: a=100
     b=200
     np.where(a>=b ,a,b)
```

```
[2]: array(200)
```

```
[3]: x=150
     y=250
     z=425
     np.where((x>=y)&(x>=z),x,(np.where((y<z),z,y)))
```

```
[3]: array(425)
```

```
[4]: ary1=np.random.randint(10,50,(8,))
```

```
[5]: ary1
```

```
[5]: array([43, 21, 26, 36, 44, 42, 19, 19])
```

```
[6]: ary2=np.random.randint(10,35,(8,))
```

```
[7]: ary2
```

```
[7]: array([31, 30, 20, 20, 34, 24, 26, 13])
```

```
[8]: np.where(ary1>ary2,ary1,ary2)
```

```
[8]: array([43, 30, 26, 36, 44, 42, 26, 19])
```

ArgMax & ArgMin

```
[9]: ary1
```

```
[9]: array([43, 21, 26, 36, 44, 42, 19, 19])
```

```
[10]: np.argmax(ary1)
```

```
[10]: 4
```

```
[11]: np.argmin(ary1)
```

```
[11]: 6
```

```
[12]: a=np.random.randint(10,50,(4,6))
      a
```

```
[12]: array([[24, 37, 28, 49, 17, 44],
             [17, 46, 21, 44, 22, 16],
             [19, 11, 48, 35, 13, 44],
             [41, 11, 18, 15, 43, 35]])
```

```
[13]: np.argmax(a)
```

```
[13]: 3
```

```
[14]: np.argmin(a)
```

```
[14]: 13
```

```
[15]: np.argmax(a,axis=0) # columns wise
```

```
[15]: array([3, 1, 2, 0, 3, 0], dtype=int64)
```

```
[16]: np.argmax(a,axis=1) # Row wise
```

```
[16]: array([3, 1, 2, 4], dtype=int64)
```

```
[17]: np.argmin(a)
```

```
[17]: 13
```

```
[18]: np.argmin(a,axis=0)
```

```
[18]: array([1, 2, 3, 3, 2, 1], dtype=int64)
```

```
[19]: ary1
```

```
[19]: array([43, 21, 26, 36, 44, 42, 19, 19])
```

```
[20]: np.max(ary1)
```

```
[20]: 44
```

```
[21]: np.argmax(ary1)
```

```
[21]: 4
```

```
[22]: np.mean(ary1)
```

```
[22]: 31.25
```

Diag

```
[23]: np.eye(4)
```

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

```
[24]: np.diag(np.arange(3,6))
```

```
[24]: array([[3, 0, 0],
             [0, 4, 0],
             [0, 0, 5]])
```

```
[25]: np.diag([3,4,5,6])
```

```
[25]: array([[3, 0, 0, 0],
             [0, 4, 0, 0],
             [0, 0, 5, 0],
             [0, 0, 0, 6]])
```

```
[26]: np.diag(np.arange(1,6))
```

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

Full

```
[27]: np.full((5),9)
```

```
[27]: array([9, 9, 9, 9, 9])
```

```
[28]: np.full((4,4),8)
```

```
[28]: array([[8, 8, 8, 8],
             [8, 8, 8, 8],
             [8, 8, 8, 8],
             [8, 8, 8, 8]])
```

Repeat

```
[29]: np.repeat([10,20,30,40],3)
```

```
[29]: array([10, 10, 10, 20, 20, 20, 30, 30, 30, 40, 40, 40])
```

Seed

```
[30]: np.random.randint(10,20,(4,5))
```

```
[30]: array([[16, 18, 16, 15, 12],
             [17, 14, 14, 16, 11],
             [17, 12, 18, 15, 17],
             [12, 15, 18, 18, 13]])
```

```
[31]: np.random.seed(0)
      np.random.randint(10,20,(3,3))
```

```
[31]: array([[15, 10, 13],
             [13, 17, 19],
             [13, 15, 12]])
```

```
[32]: b1=np.array([[10,20],[40,50]])
      b1
```

```
[32]: array([[10, 20],
             [40, 50]])
```

```
[33]: m1=np.matrix([[12,23],[10,45]])
      m1
```

```
[33]: matrix([[12, 23],
              [10, 45]])
```

```
[34]: type(m1)
```

```
[34]: numpy.matrix
```

```
[35]: type(b1)
```

```
[35]: numpy.ndarray
```

```
[36]: m2=np.matrix("3,4 ; 5,6")
      m2
```

```
[36]: matrix([[3, 4],
              [5, 6]])
```

```
[37]: m3=np.matrix("2,4;3,5")
      m3
```

```
[37]: matrix([[2, 4],
              [3, 5]])
```

```
[38]: m2*m3
```

```
[38]: matrix([[18, 32],
              [28, 50]])
```

Determinant

```
[39]: # Create a 2x2 matrix
      a1=np.random.randint(4,10,(2,2))

      # Calculate the determinant
      # det(matrix)=(a * d) -(b * c)
      determinant = np.linalg.det(a1)

      # Display the result
      print("Matrix 2x2:")
      print(a1)
      print("Determinant:", determinant)
```

```
Matrix 2x2:
[[8 4]
 [4 8]]
Determinant: 47.999999999999986
```

```
[40]: b=np.random.randint(2,8,(3,3))
      b
```

```
[40]: array([[4, 3, 2],
             [3, 7, 3],
             [7, 2, 3]])
```

```
[41]: np.linalg.det(b)
```

```
[41]: 9.999999999999993
```

Round

```
[42]: n=50.494342189
```

```
[43]: np.round(n)
```

```
[43]: 50.0
```

```
[44]: np.round(n,2)
```

```
[44]: 50.49
```

ABS

```
[45]: n1=np.array([[1,-2,4],[-6,4,-2]])
      n1
```

```
[45]: array([[ 1, -2,  4],
             [-6,  4, -2]])
```

```
[46]: np.abs(n1)
```

```
[46]: array([[1, 2, 4],
             [6, 4, 2]])
```

Expand_dims

```
[47]: ary1
```

```
[47]: array([43, 21, 26, 36, 44, 42, 19, 19])
```

```
[48]: ary1.ndim
```

```
[48]: 1
```

```
[49]: x=np.expand_dims(ary1,axis=0)
      x
```

```
[49]: array([[43, 21, 26, 36, 44, 42, 19, 19]])
```

```
[50]: x.ndim
```

```
[50]: 2
```

```
[51]: x.shape
```

```
[51]: (1, 8)
```

```
[52]: n=np.expand_dims(ary1,axis=1)
      n
```

```
[52]: array([[43],
             [21],
             [26],
             [36],
             [44],
             [42],
             [19],
             [19]])
```

```
[53]: n.ndim
```

```
[53]: 2
```

```
[54]: n.shape
```

```
[54]: (8, 1)
```

Squeeze

```
[55]: n
```

```
[55]: array([[43],
             [21],
             [26],
             [36],
             [44],
             [42],
             [19],
        [19]])
```

```
[56]: np.squeeze(n)
```

```
[56]: array([43, 21, 26, 36, 44, 42, 19, 19])
```

Count Non-Zero

```
[57]: z=np.random.randint(0,5,(10))
```

```
[58]: z
```

```
[58]: array([4, 3, 0, 3, 0, 2, 3, 0, 1, 3])
```

```
[59]: np.count_nonzero(z)
```

```
[59]: 7
```

Sort

```
[60]: ary1
```

```
[60]: array([43, 21, 26, 36, 44, 42, 19, 19])
```

```
[61]: np.sort(ary1)
```

```
[61]: array([19, 19, 21, 26, 36, 42, 43, 44])
```