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

In [2]:
```
l=[1,2,3,4]
```

In [3]:
```
ar=np.array(l)
```

In [4]:
```
ar
```

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

In [5]:
```
type(ar)
```

Out[5]:
```
numpy.ndarray
```

In [6]:
```
ar1=np.array([[1,2],[3,4]])
```

In [7]:
```
type(ar1)
```

Out[7]:
```
numpy.ndarray
```

In [8]:
```
np.asarray(l)
```

Out[8]:
```
array([1, 2, 3, 4])
```

In [9]:
```
m=np.matrix(l)
```

In [10]:
```
m
```

Out[10]:
```
matrix([[1, 2, 3, 4]])
```

In [11]:
```
np.asanyarray(m)
```

Out[11]:
```
matrix([[1, 2, 3, 4]])
```

In [12]:
```
a=np.array(l)
```

In [23]:
```
a
```

Out[23]:
```
array([1, 2, 3, 4])
```

In [13]:
```
c=a
```

In [14]:
```
c
```

Out[14]:
```
array([1, 2, 3, 4])
```

In [15]:
```
a
```

Out[15]:
```
array([1, 2, 3, 4])
```

In [16]:
```
a[0]=100
```

In [17]:
```
a
```

Out[17]:
```
array([100,   2,   3,   4])
```

In [18]:
```
c#it is called as swalo copy
```

Out[18]:
```
array([100,   2,   3,   4])
```

In [19]:
```
d=np.copy(a)
```

In [20]:
```
d
```

Out[20]:
```
array([100,   2,   3,   4])
```

In [21]:
```
a
```

Out[21]:
```
array([100,   2,   3,   4])
```

In [33]:
```
a[1]=400
```

In [22]:
```
a
```

Out[22]:
```
array([100,   2,   3,   4])
```

In [23]:
```
d#it is called as deep copy
```

Out[23]:
```
array([100,   2,   3,   4])
```

In [24]:
```
np.fromfunction(lambda i,j : i == j,(3,3))
```

Out[24]:
```
array([[ True, False, False],
       [False,  True, False],
       [False, False,  True]])
```

In [25]:
```
np.fromfunction(lambda i,j: i*j,(4,4))
```

Out[25]:
```
array([[0., 0., 0., 0.],
       [0., 1., 2., 3.],
       [0., 2., 4., 6.],
       [0., 3., 6., 9.]])
```

In [26]:
```
iterableobject=(i*i for i in range(5))
```

In [27]:
```
np.fromiter(iterableobject,int)
```

Out[27]:
```
array([ 0,  1,  4,  9, 16])
```

In [28]:
```
np.fromstring('234 235',sep=' ')
```

Out[28]:
```
array([234., 235.])
```

In [29]:
```
np.fromstring('4,5',sep=' ,')
```

Out[29]:
```
array([4., 5.])
```

In [30]:
```
# Numpy Datatypes
l=[2,3,4,5,6]
```

In [31]:
```
ar=np.array(l)
```

In [32]:
```
ar
```

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

In [33]:
```
ar.ndim # to know the dimension we use ndim
```

Out[33]:
```
1
```

In [34]:
```
ar2=np.array([[1,2,3,4],[5,6,7,8]])
```

In [35]:
```
ar2
```

Out[35]:
```
array([[1, 2, 3, 4],
       [5, 6, 7, 8]])
```

In [36]:
```
ar2.ndim
```

Out[36]:
```
2
```

In [54]:
```
ar.size # to know the number of elements
```

Out[54]:
```
5
```

In [37]:
```
ar2.size
```

Out[37]:
```
8
```

In [38]:
```
ar.shape
```

Out[38]:
```
(5,)
```

In [39]:
```
ar2.shape # to know the shape
```

Out[39]:
```
(2, 4)
```

In [40]:
```
# 2 rows and 4 columns
```

In [41]:
```
ar.dtype
```

Out[41]:
```
dtype('int64')
```

In [42]:
```
ar2.dtype
```

Out[42]:
```
dtype('int64')
```

In [43]:
```
ar22=np.array([(1.4,23,45),(54,76,88)])
```

In [63]:
```
ar22
```

Out[63]:
```
array([[ 1.4, 23. , 45. ],
       [54. , 76. , 88. ]])
```

In [44]:
```
type(ar22)
```

Out[44]:
```
numpy.ndarray
```

In [45]:
```
ar22.dtype
```

Out[45]:
```
dtype('float64')
```

In [46]:
```
range(5)
```

Out[46]:
```
range(0, 5)
```

In [47]:
```
list(range(0,5))
```

Out[47]:
```
[0, 1, 2, 3, 4]
```

In [68]:
```
# The range function does not take floating number
# But in Numpy there is a function called arange which takes floating number
```

In [48]:
```
np.arange(2.5,6.9)
```

Out[48]:
```
array([2.5, 3.5, 4.5, 5.5, 6.5])
```

In [49]:
```
np.arange(2.5,6.9,0.3)
```

Out[49]:
```
array([2.5, 2.8, 3.1, 3.4, 3.7, 4. , 4.3, 4.6, 4.9, 5.2, 5.5, 5.8, 6.1,
       6.4, 6.7])
```

In [50]:
```
list(np.arange(2.5,6.9,0.3))
```

Out[50]:
```
[2.5,
 2.8,
 3.0999999999999996,
 3.3999999999999995,
 3.6999999999999993,
 3.999999999999999,
 4.299999999999999,
 4.599999999999999,
 4.899999999999999,
 5.199999999999998,
 5.499999999999998,
 5.799999999999998,
 6.099999999999998,
 6.399999999999998,
 6.6999999999999975]
```

In [51]:
```
np.linspace(1,5,10)# linspace is used to find any random numbers in a particular
```

Out[51]:
```
array([1.        , 1.44444444, 1.88888889, 2.33333333, 2.77777778,
       3.22222222, 3.66666667, 4.11111111, 4.55555556, 5.        ])
```

In [52]:
```
np.zeros(5)
```

Out[52]:
```
array([0., 0., 0., 0., 0.])
```

In [53]:
```
np.zeros((3,4))
```

Out[53]:
```
array([[0., 0., 0., 0.],
       [0., 0., 0., 0.],
       [0., 0., 0., 0.]])
```

In [54]:
```
np.zeros((3,4,2))
```

Out[54]:
```
array([[[0.  0.]
```

```
Out[54]: array([[[0., 0.],
                 [0., 0.],
                 [0., 0.],
                 [0., 0.]],

                [[0., 0.],
                 [0., 0.],
                 [0., 0.],
                 [0., 0.]],

                [[0., 0.],
                 [0., 0.],
                 [0., 0.],
                 [0., 0.]]])
```

```
In [55]: np.ones(1)
```

```
Out[55]: array([1.])
```

```
In [56]: np.ones(4)
```

```
Out[56]: array([1., 1., 1., 1.])
```

```
In [57]: np.ones((2,3))
```

```
Out[57]: array([[1., 1., 1.],
                [1., 1., 1.]])
```

```
In [58]: on=np.ones((2,3,4))
```

```
In [59]: on
```

```
Out[59]: 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.]])
```

In [60]:
```
on + 5
```

Out[60]:
```
array([[[6., 6., 6., 6.],
        [6., 6., 6., 6.],
        [6., 6., 6., 6.]],

       [[6., 6., 6., 6.],
        [6., 6., 6., 6.],
        [6., 6., 6., 6.]]])
```

In [61]:
```
np.empty((3,6)) # to create empty array
```

Out[61]:
```
array([[0., 0., 0., 1., 1., 1.],
       [2., 2., 2., 0., 1., 2.],
       [0., 1., 2., 0., 1., 2.]])
```

In [62]:
```
np.eye(4)# for identity matrix:diagonals are 1
```

Out[62]:
```
array([[1., 0., 0., 0.],
       [0., 1., 0., 0.],
       [0., 0., 1., 0.],
       [0., 0., 0., 1.]])
```

In [63]:
```
np.linspace(2,4,20)
```

Out[63]:
```
array([2.        , 2.10526316, 2.21052632, 2.31578947, 2.42105263,
       2.52631579, 2.63157895, 2.73684211, 2.84210526, 2.94736842,
       3.05263158, 3.15789474, 3.26315789, 3.36842105, 3.47368421,
       3.57894737, 3.68421053, 3.78947368, 3.89473684, 4.        ])
```

In [64]:
```
np.logspace(2,5,10)
```

Out[64]:
```
array([   100.        ,    215.443469  ,    464.15888336,   1000.        ,
         2154.43469003,   4641.58883361,  10000.        ,  21544.34690032,
        46415.88833613, 100000.        ])
```

In [93]:
```
np.logspace(2,5,10,base=2)
```

Out[93]:
```
array([ 4.        ,  5.0396842 ,  6.34960421,  8.        , 10.0793684 ,
       12.69920842, 16.        , 20.1587368 , 25.39841683, 32.        ])
```

In [65]:
```
arr=np.random.randn(3,4)
```

In [66]:
```
import pandas as pd
```

In [67]:
```
pd.DataFrame(arr)
```

Out[67]:

	0	1	2	3
0	0.620077	0.712638	0.738966	-0.455145
1	0.296466	-0.429610	0.861479	0.334964
2	-0.165766	1.620771	-0.139649	0.315307

In [68]:
```
arr11=np.random.rand(3,4)
```

In [69]:
```
import pandas as pd
```

In [70]:
```
pd.DataFrame(arr11)
```

Out[70]:

	0	1	2	3
0	0.512576	0.991670	0.793543	0.996761
1	0.396549	0.699137	0.338419	0.518485
2	0.297071	0.419881	0.601609	0.685677

In [104…
```
#Difference between randn and rand function is randn  function gives the value wh
```

In [71]:
```
np.random.randint(1,100,(3,4))
```

Out[71]:
```
array([[26, 68, 74, 41],
       [97, 49, 90, 60],
       [76, 75, 17, 34]])
```

In [72]:
```
arr22=np.random.randint(1,100,(300,400))
```

In [73]:
```
pd.DataFrame(arr22).to_csv('text.csv')
```

In [74]:
```
arr33=np.random.rand(3,4)
```

In [111…
```
arr33
```

Out[111…
```
array([[0.40665704, 0.38896667, 0.94937969, 0.76972212],
       [0.8716088 , 0.54751645, 0.86667766, 0.80881079],
       [0.86179758, 0.85611335, 0.50335735, 0.17639481]])
```

In [75]:
```
arr33.reshape(2,6)
```

Out[75]:
```
array([[0.44534548, 0.07087468, 0.67622891, 0.33958697, 0.96247624,
        0.34937988],
       [0.53761668, 0.52631283, 0.92905601, 0.94757284, 0.27271039,
        0.05131836]])
```

In [76]:
```
arr33.reshape(6,2)
```

Out[76]:
```
array([[0.44534548, 0.07087468],
       [0.67622891, 0.33958697],
       [0.96247624, 0.34937988],
       [0.53761668, 0.52631283],
       [0.92905601, 0.94757284],
       [0.27271039, 0.05131836]])
```

In [79]:
```
arr44=arr33.reshape(6,-1111)# - anyvalue = 2
```

In [82]:
```
arr44[2][1]
```

Out[82]:
```
0.3493798750867355
```

In [83]:
```
arr44[2:5]
```

Out[83]:
```
array([[0.96247624, 0.34937988],
       [0.53761668, 0.52631283],
       [0.92905601, 0.94757284]])
```

In [84]:
```
arr44[2:5,1]
```

Out[84]:
```
array([0.34937988, 0.52631283, 0.94757284])
```

In [86]:
```
arr55=arr44[2:5]
```

In [88]:
```
arr55[0]
```

Out[88]:
```
array([0.96247624, 0.34937988])
```

In [91]:
```
arr66=np.random.randint(1,100,(5,5))
```

In [93]:
```
arr66>50
```

Out[93]:
```
array([[False, False,  True, False,  True],
       [ True, False,  True, False,  True],
       [False, False, False, False, False],
       [False, False, False,  True, False],
       [False,  True, False,  True, False]])
```

In [95]:
```
arr66[arr66>50]
```

Out[95]:
```
array([75, 82, 87, 67, 71, 91, 66, 83])
```

In [96]:
```
arr66
```

Out[96]:
```
array([[47, 45, 75,  1, 82],
       [87, 39, 67, 47, 71],
       [42, 19, 38, 23, 43],
       [26, 15, 40, 91,  7],
       [33, 66, 46, 83, 21]])
```

In [98]:
```
arr66[2:4,[2,3]]
```

Out[98]:
```
array([[38, 23],
       [40, 91]])
```

In [99]:
```
arr66[0][0]=5000
```

In [100…
```
arr66
```

Out[100…
```
array([[5000,   45,   75,    1,   82],
       [  87,   39,   67,   47,   71],
       [  42,   19,   38,   23,   43],
       [  26,   15,   40,   91,    7],
       [  33,   66,   46,   83,   21]])
```

In [104…
```
arr1=np.random.randint(1,3,(3,3))
arr2=np.random.randint(1,3,(3,3))
```

In [105…
```
arr1
```

Out[105…
```
array([[2, 1, 1],
       [2, 1, 2],
       [1, 1, 1]])
```

In [106…
```
arr2
```

Out[106…
```
array([[1, 1, 1]
```

```
       [1, 2, 2],
       [2, 2, 2]])
```

In [108…
```
arr1+arr2
```

Out[108…
```
array([[2, 2, 2],
       [2, 4, 4],
       [4, 4, 4]])
```

In [109…
```
arr1-arr2
```

Out[109…
```
array([[0, 0, 0],
       [0, 0, 0],
       [0, 0, 0]])
```

In [110…
```
arr1/arr2
```

Out[110…
```
array([[1., 1., 1.],
       [1., 1., 1.],
       [1., 1., 1.]])
```

In [112…
```
arr1*arr2# this is normal index multiplication
```

Out[112…
```
array([[1, 1, 1],
       [1, 4, 4],
       [4, 4, 4]])
```

In [113…
```
arr1@arr2# @ is used to matrix multiplication
```

Out[113…
```
array([[ 4,  5,  5],
       [ 7,  9,  9],
       [ 8, 10, 10]])
```

In [114…
```
arr1/0
```

```
/tmp/ipykernel_77/1510032488.py:1: RuntimeWarning: divide by zero encountered in d
ivide
  arr1/0
```

Out[114…
```
array([[inf, inf, inf],
       [inf, inf, inf],
       [inf, inf, inf]])
```

In [115…
```
arr1
```

Out[115…
```
array([[1, 1, 1],
       [1, 2, 2],
       [2, 2, 2]])
```

In [116…
```
arr1+100
```

Out[116…
```
array([[101, 101, 101],
       [101, 102, 102],
       [102, 102, 102]])
```

In [117…
```
arr1**2
```

Out[117…
```
array([[1, 1, 1],
       [1, 4, 4],
       [4, 4, 4]])
```

In [118…
```
#Numpy_Broadcasting
```

In [127…
```
arr=np.zeros((4,4))
```

In [128…
```
arr
```

Out[128…
```
array([[0., 0., 0., 0.],
       [0., 0., 0., 0.],
       [0., 0., 0., 0.],
       [0., 0., 0., 0.]])
```

In [124…
```
row=np.array([1,2,3,4])
```

In [125…
```
row
```

Out[125…
```
array([1, 2, 3, 4])
```

In [126…
```
arr+row
```

Out[126…
```
array([[1., 2., 3., 4.],
       [1., 2., 3., 4.],
       [1., 2., 3., 4.],
       [1., 2., 3., 4.]])
```

In [129…
```
row.T
```

Out[129…
```
array([1, 2, 3, 4])
```

In [130…
```
col=np.array([[1,2,3,4]])
```

In [131…
```
col
```

Out[131…
```
array([[1, 2, 3, 4]])
```

In [133…
```
col.T+arr
```

Out[133…
```
array([[1., 1., 1., 1.],
       [2., 2., 2., 2.],
       [3., 3., 3., 3.],
       [4., 4., 4., 4.]])
```

In [134…
```
arr1
```

Out[134…
```
array([[1, 1, 1],
       [1, 2, 2],
       [2, 2, 2]])
```

In [136…
```
arr
```

Out[136…
```
array([[0., 0., 0., 0.],
       [0., 0., 0., 0.],
       [0., 0., 0., 0.],
       [0., 0., 0., 0.]])
```

In [139…
```
arr1=np.random.randint(1,4,(3,4))
```

In [140…
```
arr1
```

Out[140…
```
array([[3, 1, 1, 3],
       [1, 2, 3, 3],
       [3, 3, 3, 2]])
```

In [141…
```
np.sqrt(arr1)
```

Out[141…
```
array([[1.73205081, 1.        , 1.        , 1.73205081],
       [1.        , 1.41421356, 1.73205081, 1.73205081],
       [1.73205081, 1.73205081, 1.73205081, 1.41421356]])
```

In [142…
```
np.exp(arr1)
```

Out[142…
```
array([[20.08553692,  2.71828183,  2.71828183, 20.08553692],
       [ 2.71828183,  7.3890561 , 20.08553692, 20.08553692],
       [20.08553692, 20.08553692, 20.08553692,  7.3890561 ]])
```

In [143…
```
np.log10(arr1)
```

Out[143…
```
array([[0.47712125, 0.        , 0.        , 0.47712125],
       [0.        , 0.30103   , 0.47712125, 0.47712125],
```