```
importing numpy-
import numpy
    or
import numpy as np
    or
from numpy import* -> for this no need to use numpy. or np.
```

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

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

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

Out[5]:
```
list
```

In [6]:
```
arr=np.array(list)
```

In [7]:
```
arr
```

Out[7]:
```
array([1, 2, 3])
```

In [8]:
```
type(arr)
```

Out[8]:
```
numpy.ndarray
```

```
attributes -
shape,dimension, type
```

In [10]:
```
arr.shape
```

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

In [11]:
```
arr.ndim
```

Out[11]:
```
1
```

In [12]:
```
arr2=np.array([3,4,5,6,7,8])
```

In [13]:
```
arr2
```

Out[13]:
```
array([3, 4, 5, 6, 7, 8])
```

In [15]:
```
arr2[2:4]
```

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

In [16]:
```
arr2[4]=90
```

In [18]:
```
arr2
```

Out[18]:
```
array([ 3,  4,  5,  6, 90,  8])
```

In [19]:
```
list1=[[1,2,3],[1,3,4,],[8,9,0]]
```

In [20]:
```
listarr=np.array(list1)
```

In [21]:
```
listarr
```

Out[21]:
```
array([[1, 2, 3],
       [1, 3, 4],
       [8, 9, 0]])
```

In [23]:
```
listarr.shape
```

Out[23]:
```
(3, 3)
```

In [24]:
```
listarr.ndim
```

Out[24]:
```
2
```

In [25]:
```
#Crete a 2d array of 4 rows and 5 cols
```

In [26]:
```
list2=[[1,2,3,4,5],[2,3,4,5,6],[3,4,5,6,7],[4,5,6,7,8]]
```

In [27]:
```
list2arr=np.array(list2)
```

In [28]:
```
list2arr.shape
```

Out[28]:
```
(4, 5)
```

In [29]:
```
list2arr.ndim
```

Out[29]:
```
2
```

In [30]:
```
list1=[[1,2,3,4,5],[6,7,8,9,10],[11,12,13,14,15],[16,17,18,19,20]]
listarr=np.array(list1)
```

In [31]:
```
listarr
```

Out[31]:
```
array([[ 1,  2,  3,  4,  5],
       [ 6,  7,  8,  9, 10],
       [11, 12, 13, 14, 15],
       [16, 17, 18, 19, 20]])
```

In [32]:
```
listarr[2]
```

Out[32]:
```
array([11, 12, 13, 14, 15])
```

In [34]:
```
listarr[:,3] # : means all rows and 3rd col, [ rows , col]
```

Out[34]:
```
array([ 4,  9, 14, 19])
```

In [124]:
```
listarr[2:4,2:5] # 2nd row : till 4-1=3 rows , 2 col : till 5-1=4 col [start:end]
```

Out[124]:
```
array([[13, 14, 15],
       [18, 19, 20]])
```

In [36]:
```
listarr.diagonal()
```

Out[36]:
```
array([ 1,  7, 13, 19])
```

In [37]:
```
listarr[0,0]
```

Out[37]:
```
1
```

In [38]:
```
listarr[1,2]
```

Out[38]:
```
8
```

In [39]:
```
listarr[2,3]
```

Out[39]:
```
14
```

```
listarr[[all rowno], [all colno]]
```

In [ ]:
```

```

In [40]:
```
listarr[[0,1,2],[0,2,3]]
```

Out[40]:
```
array([ 1,  8, 14])
```

In [43]:
```
dir(listarr)
```

Out[43]:
```
['T',
 '__abs__',
 '__add__',
 '__and__',
 '__array__',
 '__array_finalize__',
 '__array_interface__',
 '__array_prepare__',
 '__array_priority__',
 '__array_struct__',
 '__array_ufunc__',
 '__array_wrap__',
 '__bool__',
 '__class__',
 '__complex__',
 '__contains__',
 '__copy__',
 '__deepcopy__',
 '__delattr__',
```

In [44]:
```
zero=np.zeros(5) # zero function
```

In [45]:
```
zero
```

Out[45]:
```
array([0., 0., 0., 0., 0.])
```

In [47]:
```
zero=np.zeros((2,3))
```

In [48]:
```
zero
```

Out[48]:
```
array([[0., 0., 0.],
       [0., 0., 0.]])
```

In [49]:
```
one= np.ones(5) # one function
```

In [50]:
```
one
```

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

In [51]:
```
one= np.ones((5,4))
```

In [52]:
```
one
```

Out[52]:
```
array([[1., 1., 1., 1.],
       [1., 1., 1., 1.],
       [1., 1., 1., 1.],
       [1., 1., 1., 1.],
       [1., 1., 1., 1.]])
```

In [54]:
```
np.eye(4,5) # identity matrix
```

Out[54]:
```
array([[1., 0., 0., 0., 0.],
       [0., 1., 0., 0., 0.],
       [0., 0., 1., 0., 0.],
       [0., 0., 0., 1., 0.]])
```

```
create a 4,4 d array(0,15) and grab diagonals elements,3,8,15 and make
seperate array or list and grab last 2 cols and 2 rows
```

In [55]:
```
a=[[0,1,2,3],[4,5,6,7],[8,9,10,11],[12,13,14,15]]
```

In [57]:
```
a2=np.array(a)
```

In [58]:
```
a2
```

Out[58]:
```
array([[ 0,  1,  2,  3],
       [ 4,  5,  6,  7],
       [ 8,  9, 10, 11],
       [12, 13, 14, 15]])
```

In [60]:
```
a2.diagonal()
```

Out[60]:
```
array([ 0,  5, 10, 15])
```

In [63]:
```
a2[[0,2,3],[3,0,3]]
```

Out[63]:
```
array([ 3,  8, 15])
```

In [74]:
```
a2[2:4,2:4]
```

Out[74]:
```
array([[10, 11],
       [14, 15]])
```

In [79]:
```
# generate array of no in range given by user and 2 is step
arange=np.arange(20,30)
```

In [80]:
```
arange
```

Out[80]:
```
array([20, 21, 22, 23, 24, 25, 26, 27, 28, 29])
```

In [78]:
```
arange.reshape(2,5) # reshaping the array in row,col
```

Out[78]:
```
array([[20, 21, 22, 23, 24],
       [25, 26, 27, 28, 29]])
```

```
create an arry with 16 elements init and reshape it in all possible wasys
```

In [81]:
```
a=np.arange(16)
```

In [83]:
```
a.reshape(4,4)
```

Out[83]:
```
array([[ 0,  1],
       [ 2,  3],
       [ 4,  5],
       [ 6,  7],
       [ 8,  9],
       [10, 11],
       [12, 13],
       [14, 15]])
```

In [84]:
```
a.reshape(2,8)
```

Out[84]:
```
array([[ 0,  1,  2,  3,  4,  5,  6,  7],
       [ 8,  9, 10, 11, 12, 13, 14, 15]])
```

In [85]:
```
a.reshape(8,2)
```

Out[85]:
```
array([[ 0,  1],
       [ 2,  3],
       [ 4,  5],
       [ 6,  7],
       [ 8,  9],
       [10, 11],
       [12, 13],
       [14, 15]])
```

In [86]:
```
np.random.rand(2) # creating 2 random nos
```

Out[86]:
```
array([0.39389756, 0.65528811])
```

In [95]:
```
np.random.rand(4,5) #creating 4,5 random nos
```

Out[95]:
```
array([[0.90631904, 0.18821663, 0.01066084, 0.24338181, 0.11929234],
       [0.1058155 , 0.60843075, 0.73043025, 0.92512629, 0.51702459],
       [0.41784827, 0.85026696, 0.08332693, 0.13078615, 0.64958416],
       [0.92534638, 0.69379332, 0.96526897, 0.18576601, 0.12175198]])
```

In [94]:
```
# returing 20 random no between 1-100 in integer type
np.random.randint(1,100,20).reshape(4,5)
```

Out[94]:
```
array([[46, 39, 31, 48, 33],
       [37, 44, 50, 99, 13],
       [79, 19,  6, 83, 17],
       [94, 54,  9, 82,  1]])
```

In [98]:
```
# return 3 linearly spaced nos in 1-10
np.linspace(0,10,3)
```

Out[98]:
```
array([ 0. ,  5.5, 11. ])
```

```
'ravel' and 'flatten' function convert multi dimensional array to single dim
 if something changed in 'ravel' then it occur so in original array
 but something changed in 'flatten' then no change occur so in original array
```

In [103]:
```
# concatination
```

In [110]:
```
a=np.arange(9).reshape(3,3)
```

In [104]:
```
a
```

Out[104]:
```
array([[0, 1, 2],
       [3, 4, 5],
       [6, 7, 8]])
```

In [105]:
```
b=np.random.randint(1,10,9).reshape(3,3)
```

In [107]:
```
b
```

Out[107]:
```
array([[1, 3, 7],
       [9, 6, 1],
       [2, 1, 1]])
```

In [108]:
```
c=np.concatenate((a,b)) # concatinate in row form coz default axis=0
```

In [111]:
```
c
```

Out[111]:
```
array([[0, 1, 2],
       [3, 4, 5],
       [6, 7, 8],
       [1, 3, 7],
       [9, 6, 1],
       [2, 1, 1]])
```

In [112]:
```
c=np.concatenate((a,b),axis=1) # concatinate in col form
```

In [113]:
```
c
```

Out[113]:
```
array([[0, 1, 2, 1, 3, 7],
       [3, 4, 5, 9, 6, 1],
       [6, 7, 8, 2, 1, 1]])
```

In [116]:
```
s="hi "
p="hello"
q=s+p
```

In [118]:
```
q
```

Out[118]:
```
'hi hello'
```

In [121]:
```
e=c.transpose()
```

In [122]:
```
e
```

Out[122]:
```
array([[0, 3, 6],
       [1, 4, 7],
       [2, 5, 8],
       [1, 9, 2],
       [3, 6, 1],
       [7, 1, 1]])
```

In [ ]:
```

```