Creating Numpy array

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

In [2]:
```
np.array([2,4,56,422,32,1]) #1D array
```

Out[2]:
```
array([  2,   4,  56, 422,  32,   1])
```

In [3]:
```
a = np.array([2,4,56,422,32,1])
print(a)
```

```
[  2   4  56 422  32   1]
```

In [4]:
```
type(a)
```

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

In [5]:
```
#2D array
new = np.array([[45,34,22,2],[24,55,3,22]])
print(new)
```

```
[[45 34 22  2]
 [24 55  3 22]]
```

In [8]:
```
# 3 D ----

np.array( [[2,3,33,4,45],[23,45,56,66,2],[357,523,32,24,2],[32,32,44,33,234]])
```

Out[8]:
```
array([[  2,   3,  33,   4,  45],
       [ 23,  45,  56,  66,   2],
       [357, 523,  32,  24,   2],
       [ 32,  32,  44,  33, 234]])
```

dtype

In [9]:
```
np.array([11,23,44],dtype=float)
```

Out[9]:
```
array([11., 23., 44.])
```

In [10]:
```
np.array([11,23,44], dtype = bool)
```

Out[10]:
```
array([ True,  True,  True])
```

In [11]:
```
np.array([11,23,44], dtype =complex)
```

Out[11]:
```
array([11.+0.j, 23.+0.j, 44.+0.j])
```

Numpy Array vs Python sequences

In [12]:
```
np.arange(1,25)
```

Out[12]:
```
array([ 1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 12, 13, 14, 15, 16, 17,
       18, 19, 20, 21, 22, 23, 24])
```

In [13]:
```
np.arange(1,25,2)
```

Out[13]:
```
array([ 1,  3,  5,  7,  9, 11, 13, 15, 17, 19, 21, 23])
```

reshape

In [15]:
```
np.arange(1,11).reshape(5,2)
```

Out[15]:
```
array([[ 1,  2],
       [ 3,  4],
       [ 5,  6],
       [ 7,  8],
       [ 9, 10]])
```

In [16]:
```
np.arange(1,11).reshape(2,5)
```

Out[16]:
```
array([[ 1,  2,  3,  4,  5],
       [ 6,  7,  8,  9, 10]])
```

In [17]:
```
np.arange(1,13).reshape(3,4)
```

Out[17]:
```
array([[ 1,  2,  3,  4],
       [ 5,  6,  7,  8],
       [ 9, 10, 11, 12]])
```

ones & Zeros

In [18]:
```
# np.ones and np.zeros
np.ones((3,4)) # we have to mention iside tuple
```

Out[18]:
```
array([[1., 1., 1., 1.],
       [1., 1., 1., 1.],
       [1., 1., 1., 1.]])
```

In [19]:
```
np.zeros((3,4))
```

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

In [20]:
```
np.random.random((4,3))
```

Out[20]:
```
array([[0.87831171, 0.13956669, 0.23339016],
       [0.62579207, 0.20797325, 0.19519573],
       [0.60691434, 0.51892941, 0.91038864],
       [0.416482  , 0.79995236, 0.64161447]])
```

In [ ]:
```
# linspace
```

In [21]:
```
np.linspace(-10,10,10)
```

Out[21]:
```
array([-10.        ,  -7.77777778,  -5.55555556,  -3.33333333,
       -1.11111111,   1.11111111,   3.33333333,   5.55555556,
        7.77777778,  10.        ])
```

In [22]:
```
np.linspace(-2,12,6)
```

Out[22]:
```
array([-2. ,  0.8,  3.6,  6.4,  9.2, 12. ])
```

In [ ]:
```
# identity
```

In [23]:
```
np.identity(3)
```

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

In [24]:
```
np.identity(6)
```

Out[24]:
```
array([[1., 0., 0., 0., 0., 0.],
       [0., 1., 0., 0., 0., 0.],
       [0., 0., 1., 0., 0., 0.],
       [0., 0., 0., 1., 0., 0.],
       [0., 0., 0., 0., 1., 0.],
       [0., 0., 0., 0., 0., 1.]])
```

Array Attributes

In [26]:
```
a1 = np.arange(10)
a1
```

Out[26]:
```
array([0, 1, 2, 3, 4, 5, 6, 7, 8, 9])
```

In [27]:
```
a2 =np.arange(12, dtype =float).reshape(3,4) # Matrix
a2
```

Out[27]:
```
array([[ 0.,  1.,  2.,  3.],
       [ 4.,  5.,  6.,  7.],
       [ 8.,  9., 10., 11.]])
```

In [28]:
```
a3 = np.arange(8).reshape(2,2,2) # 3D --> Tensor
a3
```

Out[28]:
```
array([[[0, 1],
        [2, 3]],

       [[4, 5],
        [6, 7]]])
```

ndim

In [29]:
```
a1.ndim
```

Out[29]:
```
1
```

In [30]:
```
a2.ndim
```

Out[30]:
```
2
```

In [31]:
```
a3.ndim
```

Out[31]:
```
3
```

shape

In [32]:
```
a1.shape
```

Out[32]:
```
(10,)
```

In [33]:
```
a2.shape
```

Out[33]:
```
(3, 4)
```

In [34]:
```
a3.shape
```

Out[34]:
```
(2, 2, 2)
```

size

In [37]:
```
a3
```

Out[37]:
```
array([[[0, 1],
        [2, 3]],

       [[4, 5],
        [6, 7]]])
```

In [38]:
```
a3.size
```

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

In [39]:
```
a2
```

Out[39]:
```
array([[ 0.,  1.,  2.,  3.],
       [ 4.,  5.,  6.,  7.],
       [ 8.,  9., 10., 11.]])
```

In [40]:
```
a2.size
```

Out[40]:
```
12
```

item size

a1

In [41]:
```
a1.itemsize
```

Out[41]: 4

In [42]:
```
a2.itemsize
```

Out[42]: 8

In [43]:
```
a3.itemsize
```

Out[43]: 4

dtype

In [44]:
```
print(a1.dtype)
print(a2.dtype)
print(a3.dtype)
```

```
int32
float64
int32
```

Changing Data type

In [45]:
```
x = np.array([33,22,2.5])
x
```

Out[45]: array([33. , 22. , 2.5])

In [46]:
```
x.astype(int)
```

Out[46]: array([33, 22, 2])

Array Operations

In [47]:
```
z1 = np.arange(12).reshape(3,4)
z2 = np.arange(12,24).reshape(3,4)
```

In [48]:
```
z1
```

Out[48]:
```
array([[ 0,  1,  2,  3],
       [ 4,  5,  6,  7],
       [ 8,  9, 10, 11]])
```

In [49]:
```
z2
```

Out[49]:
```
array([[12, 13, 14, 15],
       [16, 17, 18, 19],
       [20, 21, 22, 23]])
```

In [50]:
```
# scalar operations
```

In [51]:
```
# arithmetic
z1 + 2
```

Out[51]:
```
array([[ 2,  3,  4,  5],
       [ 6,  7,  8,  9],
       [10, 11, 12, 13]])
```

In [52]:
```
# Subtraction
z1 - 2
```

Out[52]:
```
array([[-2, -1,  0,  1],
       [ 2,  3,  4,  5],
       [ 6,  7,  8,  9]])
```

In [53]:
```
# Multiplication
z1 * 2
```

Out[53]:
```
array([[ 0,  2,  4,  6],
       [ 8, 10, 12, 14],
       [16, 18, 20, 22]])
```

In [54]:
```
# power
z1 ** 2
```

Out[54]:
```
array([[  0,   1,   4,   9],
       [ 16,  25,  36,  49],
       [ 64,  81, 100, 121]])
```

In [55]:
```
## Modulo
z1 % 2
```

Out[55]:
```
array([[0, 1, 0, 1],
       [0, 1, 0, 1],
       [0, 1, 0, 1]], dtype=int32)
```

In [56]:
```
# relational operators
```

In [57]:
```
z2
```

Out[57]:
```
array([[12, 13, 14, 15],
       [16, 17, 18, 19],
       [20, 21, 22, 23]])
```

In [58]:
```
z2 > 2
```

Out[58]:
```
array([[ True,  True,  True,  True],
       [ True,  True,  True,  True],
       [ True,  True,  True,  True]])
```

In [59]:
```
z2 > 20
```

Out[59]:
```
array([[False, False, False, False],
       [False, False, False, False],
       [False,  True,  True,  True]])
```

vector operator

In [60]:
```
z1
```

Out[60]:
```
array([[ 0,  1,  2,  3],
       [ 4,  5,  6,  7],
       [ 8,  9, 10, 11]])
```

In [61]:
```
z2
```

Out[61]:
```
array([[12, 13, 14, 15],
       [16, 17, 18, 19],
       [20, 21, 22, 23]])
```

In [62]:
```
z1 + z2
```

Out[62]:
```
array([[12, 14, 16, 18],
       [20, 22, 24, 26],
       [28, 30, 32, 34]])
```

In [63]:
```
z1 * z2
```

Out[63]:
```
array([[  0,  13,  28,  45],
       [ 64,  85, 108, 133],
       [160, 189, 220, 253]])
```

In [64]:
```
z1 - z2
```

Out[64]:
```
array([[-12, -12, -12, -12],
       [-12, -12, -12, -12],
       [-12, -12, -12, -12]])
```

In [65]:
```
z1 / z2
```

Out[65]:
```
array([[0.        , 0.07692308, 0.14285714, 0.2       ],
       [0.25      , 0.29411765, 0.33333333, 0.36842105],
       [0.4       , 0.42857143, 0.45454545, 0.47826087]])
```

Array Functions

In [66]:
```
k1 = np.random.random((3,3))
k1 = np.round(k1*100)
k1
```

Out[66]:
```
array([[37., 54., 86.],
       [ 1., 45., 89.],
       [ 9., 84., 26.]])
```

In [67]:
```
np.max(k1)
```

Out[67]:
```
89.0
```

In [68]:
```
np.min(k1)
```

Out[68]:
```
1.0
```

In [69]:
```
np.sum(k1)
```

Out[69]:
```
431.0
```

In [70]:
```
np.prod(k1)
```

Out[70]: 13526691927840.0

In Numpy

In [71]:
```
np.max(k1,axis=1)
```

Out[71]:
```
array([86., 89., 84.])
```

In [72]:
```
np.max(k1, axis = 0)
```

Out[72]:
```
array([37., 84., 89.])
```

In [73]:
```
np.prod(k1, axis = 0)
```

Out[73]:
```
array([   333., 204120., 199004.])
```

In [74]:
```
# Statistics related fuctions
```

In [75]:
```
k1
```

Out[75]:
```
array([[37., 54., 86.],
       [ 1., 45., 89.],
       [ 9., 84., 26.]])
```

In [76]:
```
np.mean(k1)
```

Out[76]:
```
47.888888888888886
```

In [79]:
```
k1.mean(axis=0)
```

Out[79]:
```
array([15.66666667, 61.        , 67.        ])
```

In [80]:
```
k1.mean(axis=0)
```

Out[80]:
```
array([15.66666667, 61.        , 67.        ])
```

In [81]:
```
np.median(k1)
```

Out[81]:
```
45.0
```

In [82]:
```
np.median(k1, axis = 1)
```

Out[82]:
```
array([54., 45., 26.])
```

In [83]:
```
np.std(k1)
```

Out[83]:
```
31.27101762351006
```

In [84]:
```
np.std(k1, axis =0)
```

Out[84]:
```
array([15.4344492 , 16.673332  , 29.01723626])
```

In [85]:
```
np.var(k1)
```

Out[85]: 977.8765432098766

Trignometry Functions

In [87]:
```
np.sin(k1) # sin
```

Out[87]:
```
array([[-0.64353813, -0.55878905, -0.92345845],
       [ 0.84147098,  0.85090352,  0.86006941],
       [ 0.41211849,  0.73319032,  0.76255845]])
```

In [88]:
```
 np.cos(k1)
```

Out[88]:
```
array([[ 0.76541405, -0.82930983, -0.38369844],
       [ 0.54030231,  0.52532199,  0.51017704],
       [-0.91113026, -0.6800235 ,  0.64691932]])
```

In [89]:
```
np.tan(k1)
```

Out[89]:
```
array([[-0.84077126,  0.6738001 ,  2.40672971],
       [ 1.55740772,  1.61977519,  1.68582537],
       [-0.45231566, -1.07818381,  1.17875355]])
```

dot product

In [90]:
```
s2 = np.arange(12).reshape(3,4)
s3 = np.arange(12,24).reshape(4,3)
```

In [91]:
```
s2
```

Out[91]:
```
array([[ 0,  1,  2,  3],
       [ 4,  5,  6,  7],
       [ 8,  9, 10, 11]])
```

In [92]:
```
s3
```

Out[92]:
```
array([[12, 13, 14],
       [15, 16, 17],
       [18, 19, 20],
       [21, 22, 23]])
```

In [93]:
```
np.dot(s2,s3)
```

Out[93]:
```
array([[114, 120, 126],
       [378, 400, 422],
       [642, 680, 718]])
```

log and Exponents

In [94]:
```
np.exp(s2)
```

Out[94]:
```
array([[1.00000000e+00, 2.71828183e+00, 7.38905610e+00, 2.00855369e+01],
       [5.45981500e+01, 1.48413159e+02, 4.03428793e+02, 1.09663316e+03],
       [2.98095799e+03, 8.10308393e+03, 2.20264658e+04, 5.98741417e+04]])
```

round / floor / ceil

round

In [95]:
```
arr = np.array([1.2, 2.7, 3.5, 4.9])
rounded_arr = np.round(arr)
print(rounded_arr)
```

```
[1. 3. 4. 5.]
```

In [96]:
```
arr = np.array([1.234, 2.567, 3.891])
rounded_arr = np.round(arr, decimals=2)
print(rounded_arr)
```

```
[1.23 2.57 3.89]
```

In [97]:
```
np.round(np.random.random((2,3))*100)
```

Out[97]:
```
array([[56.,  73.,  1.],
       [51.,  32.,  0.]])
```

2.floor

In [98]:
```
arr = np.array([1.2, 2.7, 3.5, 4.9])
floored_arr = np.floor(arr)
print(floored_arr)
```

```
[1. 2. 3. 4.]
```

In [99]:
```
np.floor(np.random.random((2,3))*100)
```

Out[99]:
```
array([[72., 39., 75.],
       [ 3., 31., 76.]])
```

Ceil

In [100…
```
arr = np.array([1.2, 2.7, 3.5, 4.9])
ceiled_arr = np.ceil(arr)
print(ceiled_arr)
```

```
[2. 3. 4. 5.]
```

In [101…
```
np.ceil(np.random.random((2,3))*100)
```

Out[101…
```
array([[40., 84., 57.],
       [30.,  6., 38.]])
```

Indexing and slicing

In [102…
```
p1 = np.arange(10)
p2 = np.arange(12).reshape(3,4)
```

```
p3 = np.arange(8).reshape(2,2,2)
```

In [103…
```
p1
```

Out[103…
```
array([0, 1, 2, 3, 4, 5, 6, 7, 8, 9])
```

In [104…
```
p2
```

Out[104…
```
array([[ 0,  1,  2,  3],
       [ 4,  5,  6,  7],
       [ 8,  9, 10, 11]])
```

In [105…
```
p3
```

Out[105…
```
array([[[0, 1],
        [2, 3]],

       [[4, 5],
        [6, 7]]])
```

Indexing on 1D array

In [106…
```
p1
```

Out[106…
```
array([0, 1, 2, 3, 4, 5, 6, 7, 8, 9])
```

In [107…
```
p1[-1]
```

Out[107…
```
9
```

In [108…
```
p1[0]
```

Out[108…
```
0
```

Indexing on 2D array

In [109…
```
p2
```

Out[109…
```
array([[ 0,  1,  2,  3],
       [ 4,  5,  6,  7],
       [ 8,  9, 10, 11]])
```

In [110…
```
p2[1,2]
```

Out[110…
```
6
```

In [111…
```
p2[2,3]
```

Out[111…
```
11
```

In [112…
```
p2[1,0]
```

Out[112…
```
4
```

Indexing on 3D

In [113…
```
p3
```

Out[113…
```
array([[[0, 1],
        [2, 3]],

       [[4, 5],
        [6, 7]]])
```

In [114…
```
p3[1,0,1]
```

Out[114…
```
5
```

In [115…
```
p3[0,1,0]
```

Out[115…
```
2
```

In [116…
```
p3[0,0,0]
```

Out[116…
```
0
```

In [117…
```
p3[1,1,0]
```

Out[117…
```
6
```

Slicing

Slicing on 1D

In [119…
```
p1
```

Out[119…
```
array([0, 1, 2, 3, 4, 5, 6, 7, 8, 9])
```

In [120…
```
p1[2:5]
```

Out[120…
```
array([2, 3, 4])
```

In [121…
```
p1[2:5:2]
```

Out[121…
```
array([2, 4])
```

Slicing on 2D

In [122…
```
p2
```

Out[122…
```
array([[ 0,  1,  2,  3],
       [ 4,  5,  6,  7],
       [ 8,  9, 10, 11]])
```

In [123…
```
p2[0,:]
```

Out[123…
```
array([0, 1, 2, 3])
```

In [124…
```
p2[:,2]
```

Out[124…
```
array([ 2,  6, 10])
```

In [125…
```
p2
```

Out[125…
```
array([[ 0,  1,  2,  3],
       [ 4,  5,  6,  7],
       [ 8,  9, 10, 11]])
```

In [126…
```
p2[1:3]
```

Out[126…
```
array([[ 4,  5,  6,  7],
       [ 8,  9, 10, 11]])
```

In [127…
```
p2[1:3,1:3]
```

Out[127…
```
array([[ 5,  6],
       [ 9, 10]])
```

In [128…
```
p2
```

Out[128…
```
array([[ 0,  1,  2,  3],
       [ 4,  5,  6,  7],
       [ 8,  9, 10, 11]])
```

In [129…
```
p2[::2,::3]
```

Out[129…
```
array([[ 0,  3],
       [ 8, 11]])
```

In [130…
```
p2
```

Out[130…
```
array([[ 0,  1,  2,  3],
       [ 4,  5,  6,  7],
       [ 8,  9, 10, 11]])
```

In [131…
```
p2[::2]
```

Out[131…
```
array([[ 0,  1,  2,  3],
       [ 8,  9, 10, 11]])
```

In [132…
```
p2[::2,1::2]
```

Out[132…
```
array([[ 1,  3],
       [ 9, 11]])
```

In [133…
```
p2
```

Out[133…
```
array([[ 0,  1,  2,  3],
       [ 4,  5,  6,  7],
       [ 8,  9, 10, 11]])
```

In [134…
```
p2[1]
```

Out[134… array([4, 5, 6, 7])

In [135…
```
p2[1,::3]
```

Out[135… array([4, 7])

In [136…
```
p2
```

Out[136…
```
array([[ 0,  1,  2,  3],
       [ 4,  5,  6,  7],
       [ 8,  9, 10, 11]])
```

In [139…
```
p2[0:2]
```

Out[139…
```
array([[0, 1, 2, 3],
       [4, 5, 6, 7]])
```

In [140…
```
p2[0:2,1:]
```

Out[140…
```
array([[1, 2, 3],
       [5, 6, 7]])
```

In [141…
```
p2
```

Out[141…
```
array([[ 0,  1,  2,  3],
       [ 4,  5,  6,  7],
       [ 8,  9, 10, 11]])
```

In [142…
```
p2[0:2]
```

Out[142…
```
array([[0, 1, 2, 3],
       [4, 5, 6, 7]])
```

In [143…
```
p2[0:2,1:2]
```

Out[143…
```
array([[1],
       [5]])
```

slicing in 3D

In [144…
```
p3 = np.arange(27).reshape(3,3,3)
p3
```

Out[144…
```
array([[[ 0,  1,  2],
        [ 3,  4,  5],
        [ 6,  7,  8]],

       [[ 9, 10, 11],
        [12, 13, 14],
        [15, 16, 17]],

       [[18, 19, 20],
        [21, 22, 23],
        [24, 25, 26]]])
```

In [145…
```
p3[1]
```

Out[145…
```
array([[ 9, 10, 11],
       [12, 13, 14],
       [15, 16, 17]])
```

In [146…
```
p3[::2]
```

Out[146…
```
array([[[ 0,  1,  2],
        [ 3,  4,  5],
        [ 6,  7,  8]],

       [[18, 19, 20],
        [21, 22, 23],
        [24, 25, 26]]])
```

In [147…
```
p3
```

Out[147…
```
array([[[ 0,  1,  2],
        [ 3,  4,  5],
        [ 6,  7,  8]],

       [[ 9, 10, 11],
        [12, 13, 14],
        [15, 16, 17]],

       [[18, 19, 20],
        [21, 22, 23],
        [24, 25, 26]]])
```

In [148…
```
p3[0] # first numpy array
```

Out[148…
```
array([[0, 1, 2],
       [3, 4, 5],
       [6, 7, 8]])
```

In [149…
```
p3[0,1,:]
```

Out[149…
```
array([3, 4, 5])
```

In [150…
```
p3
```

Out[150…
```
array([[[ 0,  1,  2],
        [ 3,  4,  5],
        [ 6,  7,  8]],

       [[ 9, 10, 11],
        [12, 13, 14],
        [15, 16, 17]],

       [[18, 19, 20],
        [21, 22, 23],
        [24, 25, 26]]])
```

In [151…
```
p3[1]
```

Out[151…
```
array([[ 9, 10, 11],
       [12, 13, 14],
       [15, 16, 17]])
```

In [152…
```
p3[1,:,1]
```

Out[152… array([10, 13, 16])

In [153…
```
p3
```

Out[153…
```
array([[[ 0,  1,  2],
        [ 3,  4,  5],
        [ 6,  7,  8]],

       [[ 9, 10, 11],
        [12, 13, 14],
        [15, 16, 17]],

       [[18, 19, 20],
        [21, 22, 23],
        [24, 25, 26]]])
```

In [154…
```
p3[2]
```

Out[154…
```
array([[18, 19, 20],
       [21, 22, 23],
       [24, 25, 26]])
```

In [155…
```
p3[2,1:]
```

Out[155…
```
array([[21, 22, 23],
       [24, 25, 26]])
```

In [156…
```
p3[2,1:,1:]
```

Out[156…
```
array([[22, 23],
       [25, 26]])
```

In [157…
```
p3
```

Out[157…
```
array([[[ 0,  1,  2],
        [ 3,  4,  5],
        [ 6,  7,  8]],

       [[ 9, 10, 11],
        [12, 13, 14],
        [15, 16, 17]],

       [[18, 19, 20],
        [21, 22, 23],
        [24, 25, 26]]])
```

In [158…
```
p3[0::2,0]
```

Out[158…
```
array([[ 0,  1,  2],
       [18, 19, 20]])
```

In [159…
```
p3[0::2,0,::2]
```

Out[159…
```
array([[ 0,  2],
       [18, 20]])
```

Iterating

In [160…
```
p1
```

Out[160…
```
array([0, 1, 2, 3, 4, 5, 6, 7, 8, 9])
```

In [161…
```
for i in p1:
    print(i)
```

```
0
1
2
3
4
5
6
7
8
9
```

In [162…
```
p2
```

Out[162…
```
array([[ 0,  1,  2,  3],
       [ 4,  5,  6,  7],
       [ 8,  9, 10, 11]])
```

In [163…
```
for i in p2:
    print(i)
```

```
[0 1 2 3]
[4 5 6 7]
[ 8  9 10 11]
```

In [164…
```
p3
```

Out[164…
```
array([[[ 0,  1,  2],
        [ 3,  4,  5],
        [ 6,  7,  8]],

       [[ 9, 10, 11],
        [12, 13, 14],
        [15, 16, 17]],

       [[18, 19, 20],
        [21, 22, 23],
        [24, 25, 26]]])
```

In [165…
```
for i in p3:
    print(i)
```

```
[[0 1 2]
 [3 4 5]
 [6 7 8]]
[[ 9 10 11]
 [12 13 14]
 [15 16 17]]
[[18 19 20]
 [21 22 23]
 [24 25 26]]
```

In [166…
```
for i in np.nditer(p3):
    print(i)
```

```
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
```

Reshaping

In [167…
```
p2
```

Out[167…
```
array([[ 0,  1,  2,  3],
       [ 4,  5,  6,  7],
       [ 8,  9, 10, 11]])
```

In [168…
```
np.transpose(p2)
```

Out[168…
```
array([[ 0,  4,  8],
       [ 1,  5,  9],
       [ 2,  6, 10],
       [ 3,  7, 11]])
```

In [169…
```
p2.T
```

Out[169…
```
array([[ 0,  4,  8],
       [ 1,  5,  9],
       [ 2,  6, 10],
       [ 3,  7, 11]])
```

In [170…
```
p3
```

Out[170…
```
array([[[ 0,  1,  2],
        [ 3,  4,  5],
        [ 6,  7,  8]],

       [[ 9, 10, 11],
        [12, 13, 14],
        [15, 16, 17]],

       [[18, 19, 20],
        [21, 22, 23],
        [24, 25, 26]]])
```

In [171…
```
p3.T
```

Out[171…
```
array([[[ 0,  9, 18],
        [ 3, 12, 21],
        [ 6, 15, 24]],

       [[ 1, 10, 19],
        [ 4, 13, 22],
        [ 7, 16, 25]],

       [[ 2, 11, 20],
        [ 5, 14, 23],
        [ 8, 17, 26]]])
```

Ravel

In [172…
```
p2
```

Out[172…
```
array([[ 0,  1,  2,  3],
       [ 4,  5,  6,  7],
       [ 8,  9, 10, 11]])
```

In [173…
```
p2.ravel()
```

Out[173…
```
array([ 0,  1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11])
```

In [174…
```
p3
```

Out[174…
```
array([[[ 0,  1,  2],
        [ 3,  4,  5],
        [ 6,  7,  8]],

       [[ 9, 10, 11],
        [12, 13, 14],
        [15, 16, 17]],

       [[18, 19, 20],
        [21, 22, 23],
        [24, 25, 26]]])
```

In [175…
```
p3.ravel()
```

Out[175…
```
array([ 0,  1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 12, 13, 14, 15, 16,
       17, 18, 19, 20, 21, 22, 23, 24, 25, 26])
```

stacking

In [ ]:
```
w1 = np.arange(12).reshape(3,4)
w2 = np.arange(12,24).reshape(3,4)
```

In [ ]:
```
w1
```

In [ ]:
```
w2
```

In [ ]:
```
np.hstack((w1,w2))
```

In [ ]:
```
w1
```

In [ ]:
```
w2
```

In [ ]:
```
# using vstack for vertical stacking
```

In [ ]:
```
np.vstack((w1,w2))
```

In [ ]:
```
# Splitting
```

In [ ]:
```
# Horizontal splitting
w1
```

In [ ]:
```
np.hsplit(w1,2) # splitting by 2
```

In [ ]:
```
np.hsplit(w1,4) # splitting by 4
```

In [ ]:
```
# Vertical splitting
w2
```

In [ ]:
```
np.vsplit(w2,3) # splittig into 3 rows
```

In [ ]:

In [ ]:

In [ ]:

In [ ]: