creating Numpy array

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

```
In [2]: np.array([4,2,5,7,9,8,2])#1d array
```

```
Out[2]: array([4, 2, 5, 7, 9, 8, 2])
```

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

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

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

```
<class 'numpy.ndarray'>
```

```
In [5]: np.array([[3,5,6,7,8],[4,6,8,9,3,]])#2d array
```

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

```
In [6]: a1=np.array([[3,5,6,7,8],[4,6,8,9,3,]])
        a1
```

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

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

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

```
In [8]: a2=np.array([[2,4,5,3,1],[3,5,6,9,8],[2,4,3,1,6]])#nd array
```

```
In [9]: a2
```

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

dtype in array

```
In [10]: np.array([23,43,56],dtype=float)
```

```
Out[10]: array([23., 43., 56.])
```

```
In [11]: np.array([23,43,56],dtype=bool)
```

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

```
In [12]: np.array([23,43,56],dtype=complex)
```

```
Out[12]: array([23.+0.j, 43.+0.j, 56.+0.j])
```

```
In [13]: np.array([23,43,56])

Out[13]: array([23, 43, 56])
```

arrange

```
In [14]: np.arange(1,10)

Out[14]: array([1, 2, 3, 4, 5, 6, 7, 8, 9])

In [15]: np.arange(20,50)

Out[15]: array([20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36,
                37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49])

In [16]: np.arange(2,50,5)

Out[16]: array([ 2,  7, 12, 17, 22, 27, 32, 37, 42, 47])

In [17]: np.arange(-20,30,5)

Out[17]: array([-20, -15, -10,  -5,   0,   5,  10,  15,  20,  25])
```

reshape

```
In [18]: np.arange(1,13 ).reshape(3,4)

Out[18]: array([[ 1,  2,  3,  4],
                [ 5,  6,  7,  8],
                [ 9, 10, 11, 12]])

In [19]: np.arange(1,50 ).reshape(7,7)

Out[19]: 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, 25, 26, 27, 28],
                [29, 30, 31, 32, 33, 34, 35],
                [36, 37, 38, 39, 40, 41, 42],
                [43, 44, 45, 46, 47, 48, 49]])

In [20]: np.arange(1,82).reshape(9,9)

Out[20]: 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, 25, 26, 27],
                [28, 29, 30, 31, 32, 33, 34, 35, 36],
                [37, 38, 39, 40, 41, 42, 43, 44, 45],
                [46, 47, 48, 49, 50, 51, 52, 53, 54],
                [55, 56, 57, 58, 59, 60, 61, 62, 63],
                [64, 65, 66, 67, 68, 69, 70, 71, 72],
                [73, 74, 75, 76, 77, 78, 79, 80, 81]])

In [21]: np.arange(0,100).reshape(10,10)
```

```
Out[21]: 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, 27, 28, 29],
                [30, 31, 32, 33, 34, 35, 36, 37, 38, 39],
                [40, 41, 42, 43, 44, 45, 46, 47, 48, 49],
                [50, 51, 52, 53, 54, 55, 56, 57, 58, 59],
                [60, 61, 62, 63, 64, 65, 66, 67, 68, 69],
                [70, 71, 72, 73, 74, 75, 76, 77, 78, 79],
                [80, 81, 82, 83, 84, 85, 86, 87, 88, 89],
                [90, 91, 92, 93, 94, 95, 96, 97, 98, 99]])
```

ones & zeros

```
In [22]: np.zeros((3,5))
```

```
Out[22]: array([[0., 0., 0., 0., 0.],
                [0., 0., 0., 0., 0.],
                [0., 0., 0., 0., 0.]])
```

```
In [23]: np.zeros((3,5),dtype=int)
```

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

```
In [24]: np.ones((5,4))
```

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

```
In [25]: np.ones((5,4),dtype=int)
```

```
Out[25]: array([[1, 1, 1, 1],
                [1, 1, 1, 1],
                [1, 1, 1, 1],
                [1, 1, 1, 1],
                [1, 1, 1, 1]])
```

```
In [26]: np.random .rand(5)
```

```
Out[26]: array([0.24790233, 0.60817075, 0.65405924, 0.66120935, 0.09892906])
```

```
In [27]: np.random .rand(5,5)
```

```
Out[27]: array([[0.45355106, 0.67264235, 0.02192976, 0.8795053 , 0.5291788 ],
                [0.58472368, 0.51259961, 0.80588804, 0.91611238, 0.7490717 ],
                [0.92763001, 0.21926214, 0.35050733, 0.5618063 , 0.50953436],
                [0.12664946, 0.92556018, 0.89323393, 0.67524108, 0.25982033],
                [0.90114668, 0.50688248, 0.80736205, 0.09597215, 0.36194026]])
```

```
In [28]: np.random .randint(5)
```

```
Out[28]: 0
```

```
In [29]: np.random .randint(5,9)
```

```
Out[29]: 6
```

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

```
Out[30]: array([11, 19,  7, 12,  7, 19,  9,  9, 14, 15], dtype=int32)
```

```
In [31]: np.random .randint(5,20,(8,5))
```

```
Out[31]: array([[ 9, 12, 18, 10, 12],
                [ 8,  7, 17, 15,  7],
                [17, 19, 11, 11,  9],
                [ 6,  8, 12, 18, 12],
                [ 6,  9,  8, 11,  5],
                [17, 11,  9, 11, 17],
                [15, 11, 17, 12,  7],
                [15, 16, 16,  5, 19]], dtype=int32)
```

linspace(give range at equal distance)

```
In [32]: np.linspace(-10,10,5)
```

```
Out[32]: array([-10.,  -5.,   0.,   5.,  10.])
```

```
In [ ]: 
```

```
In [33]: np.linspace(1,50,3)
```

```
Out[33]: array([ 1. , 25.5, 50. ])
```

```
In [34]: np.linspace(20,100,5)
```

```
Out[34]: array([ 20.,  40.,  60.,  80., 100.])
```

Identity in matrix diagnal items will be 1 remaining will be zero

```
In [35]: np.identity(5)
```

```
Out[35]: array([[1., 0., 0., 0., 0.],
                [0., 1., 0., 0., 0.],
                [0., 0., 1., 0., 0.],
                [0., 0., 0., 1., 0.],
                [0., 0., 0., 0., 1.]])
```

```
In [36]: np.identity(10)
```

```
Out[36]: array([[1., 0., 0., 0., 0., 0., 0., 0., 0., 0.],
                [0., 1., 0., 0., 0., 0., 0., 0., 0., 0.],
                [0., 0., 1., 0., 0., 0., 0., 0., 0., 0.],
                [0., 0., 0., 1., 0., 0., 0., 0., 0., 0.],
                [0., 0., 0., 0., 1., 0., 0., 0., 0., 0.],
                [0., 0., 0., 0., 0., 1., 0., 0., 0., 0.],
                [0., 0., 0., 0., 0., 0., 1., 0., 0., 0.],
                [0., 0., 0., 0., 0., 0., 0., 1., 0., 0.],
                [0., 0., 0., 0., 0., 0., 0., 0., 1., 0.],
                [0., 0., 0., 0., 0., 0., 0., 0., 0., 1.]])
```

Array Attributes

```
In [37]: a4=np.arange(10)
```

```
In [38]: a4
```

```
Out[38]: array([0, 1, 2, 3, 4, 5, 6, 7, 8, 9])
```

```
In [39]: a5=np.arange(25, dtype=float).reshape(5,5)
         a5
```

```
Out[39]: 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.]])
```

```
In [40]: a6=np.arange(27).reshape(3,3,3)
```

```
In [41]: a6
```

```
Out[41]: 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]]])
```

ndim(give no of dimentions in the given array)

```
In [42]: a1.ndim
```

```
Out[42]: 2
```

```
In [43]: a.ndim
```

```
Out[43]: 1
```

```
In [44]: a2.ndim
```

```
Out[44]: 2
```

```
In [45]: a4.ndim
```

```
Out[45]: 1
```

```
In [46]: a5.ndim
```

```
Out[46]: 2
```

```
In [47]: a6.ndim
```

```
Out[47]: 3
```

shape(gives no of rows and columns)

```
In [48]: a.shape
```

```
Out[48]: (7,)
```

```
In [49]: a1.shape
```

```
Out[49]: (2, 5)
```

```
In [50]: a2.shape
```

```
Out[50]: (3, 5)
```

```
In [51]: a4.shape
```

```
Out[51]: (10,)
```

```
In [52]: a5.shape
```

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

```
In [53]: a6.shape
```

```
Out[53]: (3, 3, 3)
```

size(give no of items)

```
In [54]: a.size
```

```
Out[54]: 7
```

```
In [55]: a1.size
```

```
Out[55]: 10
```

```
In [56]: a2.size
```

```
Out[56]: 15
```

```
In [57]: a4.size
```

```
Out[57]: 10
```

```
In [58]: a5.size
```

```
Out[58]: 25
```

```
In [59]: a6.size
```

```
Out[59]: 27
```

item size(memory occupied by item)

```
In [60]: a.itemsize
```

```
Out[60]: 8
```

```
In [61]: a1.itemsize
```

```
Out[61]: 8
```

```
In [62]: a2.itemsize
```

```
Out[62]: 8
```

```
In [63]: a4.itemsize
```

```
Out[63]: 8
```

```
In [64]: a5.itemsize
```

```
Out[64]: 8
```

```
In [65]: a6.itemsize
```

```
Out[65]: 8
```

```
In [66]: a
```

```
Out[66]: array([4, 2, 5, 7, 9, 8, 2])
```

```
In [67]: a1
```

```
Out[67]: array([[3, 5, 6, 7, 8],
                [4, 6, 8, 9, 3]])
```

```
In [68]: a2
```

```
Out[68]: array([[2, 4, 5, 3, 1],
                [3, 5, 6, 9, 8],
                [2, 4, 3, 1, 6]])
```

```
In [69]: a2.itemsize
```

```
Out[69]: 8
```

dtype(give data type)

```
In [70]: print(a.dtype)
         print(a1.dtype)
         print(a4.dtype)
         print(a5.dtype)
```

```
int64
int64
int64
float64
```

change data type

```
In [71]: x=np.array([2.3,4.5,7.8])
         x
```

```
Out[71]: array([2.3, 4.5, 7.8])
```

```
In [72]: x.astype(int)
```

```
Out[72]: array([2, 4, 7])
```

```
In [73]: x.astype(bool)
```

```
Out[73]: array([ True,  True,  True])
```

```
In [74]: x.astype(complex)
```

```
Out[74]: array([2.3+0.j, 4.5+0.j, 7.8+0.j])
```

Array operations

```
In [75]: z=np.arange(12).reshape(3,4)
         z
```

```
Out[75]: array([[ 0,  1,  2,  3],
                [ 4,  5,  6,  7],
                [ 8,  9, 10, 11]])
```

```
In [76]:  z1=np.arange(0,50).reshape(10,5)
          z1
```

```
Out[76]: 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, 27, 28, 29],
                [30, 31, 32, 33, 34],
                [35, 36, 37, 38, 39],
                [40, 41, 42, 43, 44],
                [45, 46, 47, 48, 49]])
```

scalar operations(perform all mathematical operations on each element of array)

```
In [77]: z
```

```
Out[77]: array([[ 0,  1,  2,  3],
                [ 4,  5,  6,  7],
                [ 8,  9, 10, 11]])
```

z+2#arthematic operations

```
In [78]: z-2
```

```
Out[78]: array([[-2, -1,  0,  1],
                [ 2,  3,  4,  5],
                [ 6,  7,  8,  9]])
```

```
In [79]: z*2
```

```
Out[79]: array([[ 0,  2,  4,  6],
                [ 8, 10, 12, 14],
                [16, 18, 20, 22]])
```

```
In [80]: z/2
```

```
Out[80]: array([[0. , 0.5, 1. , 1.5],
                [2. , 2.5, 3. , 3.5],
                [4. , 4.5, 5. , 5.5]])
```

```
In [81]: z//2
```

```
Out[81]: array([[0, 0, 1, 1],
                [2, 2, 3, 3],
                [4, 4, 5, 5]])
```

```
In [82]: z%2
```

```
Out[82]: array([[0, 1, 0, 1],
                [0, 1, 0, 1],
                [0, 1, 0, 1]])
```

```
In [83]: z**2
```

```
Out[83]: array([[  0,   1,   4,   9],
                [ 16,  25,  36,  49],
                [ 64,  81, 100, 121]])
```

```
In [84]: z<10#relational operations
```

```
Out[84]: array([[ True,  True,  True,  True],
                [ True,  True,  True,  True],
                [ True,  True, False, False]])
```

```
In [85]: z>5
```

```
Out[85]: array([[False, False, False, False],
                [False, False,  True,  True],
                [ True,  True,  True,  True]])
```

```
In [86]: z==5
```

```
Out[86]: array([[False, False, False, False],
                [False,  True, False, False],
                [False, False, False, False]])
```

```
In [87]: z<=10
```

```
Out[87]: array([[ True,  True,  True,  True],
                [ True,  True,  True,  True],
                [ True,  True,  True, False]])
```

```
In [88]: z>=5
```

```
Out[88]: array([[False, False, False, False],
                [False,  True,  True,  True],
                [ True,  True,  True,  True]])
```

```
In [89]: z
```

```
Out[89]: array([[ 0,  1,  2,  3],
                [ 4,  5,  6,  7],
                [ 8,  9, 10, 11]])
```

```
In [90]: z1
```

```
Out[90]: 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, 27, 28, 29],
                [30, 31, 32, 33, 34],
                [35, 36, 37, 38, 39],
                [40, 41, 42, 43, 44],
                [45, 46, 47, 48, 49]])
```

```
In [91]: z3=np.arange(12) .reshape(3,4)
         z3
```

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

vector(we can add both numpy array)

```
In [92]: z
```

```
Out[92]: array([[ 0,  1,  2,  3],
                [ 4,  5,  6,  7],
                [ 8,  9, 10, 11]])
```

```
In [93]: z3
```

```
Out[93]: array([[ 0,  1,  2,  3],
                [ 4,  5,  6,  7],
                [ 8,  9, 10, 11]])
```

```
In [94]: z+z3#we can add when same no of rows and columns
```

```
Out[94]: array([[ 0,  2,  4,  6],
                [ 8, 10, 12, 14],
                [16, 18, 20, 22]])
```

```
In [95]: z-z3
```

```
Out[95]: array([[0, 0, 0, 0],
                [0, 0, 0, 0],
                [0, 0, 0, 0]])
```

```
In [96]: z*z3
```

```
Out[96]: array([[  0,   1,   4,   9],
                [ 16,  25,  36,  49],
                [ 64,  81, 100, 121]])
```

```
In [97]: z//z3
```

```
C:\Users\SWAPNA\AppData\Local\Temp\ipykernel_24580\2991310609.py:1: RuntimeWarnin
g: divide by zero encountered in floor_divide
  z//z3
```

```
Out[97]: array([[0, 1, 1, 1],
                [1, 1, 1, 1],
                [1, 1, 1, 1]])
```

Array functions

```
In [98]: k1=np.random.random((3,3))
         k1=np.round(k1*100)
         k1
```

```
Out[98]: array([[79., 97., 36.],
                [ 3., 26., 92.],
                [35., 86., 87.]])
```

```
In [99]: np.max(k1)
```

```
Out[99]: np.float64(97.0)
```

```
In [100…  np.min(k1)
```

```
Out[100…  np.float64(3.0)
```

```
In [101…  np.mean(k1)
```

```
Out[101…  np.float64(60.111111111111114)
```

```
In [102…  np.prod(k1)
```

```
Out[102…  np.float64(518405385476160.0)
```

```
In [103…  np.sum(k1)
```

```
Out[103…  np.float64(541.0)
```

in numpy 0=column 1=row

```
In [104…  k1
```

```
Out[104…  array([[79., 97., 36.],
                 [ 3., 26., 92.],
                 [35., 86., 87.]])
```

```
In [105…  np.max(k1,axis=1)#max of each row
```

```
Out[105…  array([97., 92., 87.])
```

```
In [106…  np.max(k1,axis=0)#max of each column
```

```
Out[106…  array([79., 97., 92.])
```

```
In [107…  np.mean(k1,axis=1)
```

```
Out[107…  array([70.66666667, 40.33333333, 69.33333333])
```

In [108…

```
np.median(k1,axis=1)
```

Out[108…

```
array([79., 26., 86.])
```

In [109…

```
np.prod(k1,axis=1)
```

Out[109…

```
array([275868.,   7176., 261870.])
```

In [110…

```
#statstical functions
```

In [111…

```
k1
```

Out[111…

```
array([[79., 97., 36.],
       [ 3., 26., 92.],
       [35., 86., 87.]])
```

In [112…

```
np.mean(k1)
```

Out[112…

```
np.float64(60.111111111111114)
```

In [113…

```
np.mean (k1,axis=0)
```

Out[113…

```
array([39.        , 69.66666667, 71.66666667])
```

In [114…

```
np.median(k1)
```

Out[114…

```
np.float64(79.0)
```

In [115…

```
np.median(k1,axis=0)
```

Out[115…

```
array([35., 86., 87.])
```

In [116…

```
np.std(k1)#stadard deviation
```

Out[116…

```
np.float64(32.939150783239505)
```

In [117…

```
np.var(k1)#variance
```

Out[117…

```
np.float64(1084.9876543209875)
```

trignometry functions

In [118…

```
np.sin(k1)
```

Out[118…

```
array([[-0.44411267,  0.37960774, -0.99177885],
       [ 0.14112001,  0.76255845, -0.77946607],
       [-0.42818267, -0.92345845, -0.82181784]])
```

In [119…

```
np.cos(k1)
```

Out[119…

```
array([[-0.89597095, -0.92514754, -0.12796369],
       [-0.9899925 ,  0.64691932, -0.62644445],
       [-0.90369221, -0.38369844,  0.56975033]])
```

In [120…

```
np.tan(k1)
```

Out[120…

```
array([[ 0.49567753, -0.4103213 ,  7.75047091],
       [-0.14254654,  1.17875355,  1.24427006],
       [ 0.47381472,  2.40672971, -1.44241747]])
```

dot product(possible only rows should be reverse,3,4=4,3

In [121…

```
z1
```

Out[121…

```
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, 27, 28, 29],
       [30, 31, 32, 33, 34],
       [35, 36, 37, 38, 39],
       [40, 41, 42, 43, 44],
       [45, 46, 47, 48, 49]])
```

In [122…

```
z3
```

Out[122…

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

In [123…

```
z
```

Out[123…

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

In [124…

```
s1=np.arange(1,13).reshape(4,3)
```

In [125…

```
s1
```

Out[125…

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

In [126…

```
np.dot(s1,z)
```

Out[126…

```
array([[ 32,  38,  44,  50],
       [ 68,  83,  98, 113],
       [104, 128, 152, 176],
       [140, 173, 206, 239]])
```

round,ceil and floor

In [127…

```
arr=np.array([1.2,3.7,4.8])
rounded_arr=np.round(arr)
print(rounded_arr)
```

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

In [128…

```
arr=np.array([1.234,3.754,4.856])#with two dcimal
rounded_arr=np.round(arr,decimals=2)
print(rounded_arr)
```

```
[1.23 3.75 4.86]
```

In [129…

```
arr=np.array([1.234,3.754,4.856])#with one dcimal
rounded_arr=np.round(arr,decimals=1)
print(rounded_arr)
```

```
[1.2 3.8 4.9]
```

In [130…

```
np.round(np.random.random((2,3))*100)
```

Out[130…

```
array([[61., 45., 17.],
       [65., 62., 10.]])
```

In [131…

```
arr=np.array([1.2,3.7,4.8])#floor
floored_arr=np.floor(arr)
print(floored_arr)
```

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

In [132…

```
arr=np.array([1.2,3.7,4.8])#ceil
ceiled_arr=np.ceil(arr)
print(ceiled_arr)
```

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

In [133…

```
p=np.arange(10)
p1=np.arange(1,13).reshape(3,4)
p2=np.arange(27).reshape(3,3,3)
```

INDEXING

In [134…

```
p1
```

Out[134…

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

In [135…

```
p2
```

Out[135…

```
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 [136…

```
p
```

Out[136…

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

In [137…

```
p1[1]
```

Out[137…

```
array([5, 6, 7, 8])
```

In [138…

```
p1
```

```
Out[138…  array([[ 1,  2,  3,  4],
                 [ 5,  6,  7,  8],
                 [ 9, 10, 11, 12]])
```

```
In [139…  p1[-1]
```

```
Out[139…  array([ 9, 10, 11, 12])
```

```
In [140…  p2
```

```
Out[140…  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 [141…  p2[0,0,2]
```

```
Out[141…  np.int64(2)
```

```
In [142…  p2[1,2,0]
```

```
Out[142…  np.int64(15)
```

```
In [143…  p2[2,1,0]
```

```
Out[143…  np.int64(21)
```

```
In [144…  p2[0,2,0]
```

```
Out[144…  np.int64(6)
```

```
In [145…  p1
```

```
Out[145…  array([[ 1,  2,  3,  4],
                 [ 5,  6,  7,  8],
                 [ 9, 10, 11, 12]])
```

```
In [146…  p1[0,0]
```

```
Out[146…  np.int64(1)
```

```
In [147…  p1[1,0]
```

```
Out[147…  np.int64(5)
```

```
In [148…  p1[-3,-4]
```

```
Out[148…  np.int64(1)
```

```
In [149…  p1[-2,2]
```

Out[149…

```
np.int64(7)
```

SLICING

In [150…

```
p1
```

Out[150…

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

In [151…

```
p1[2,1:]
```

Out[151…

```
array([10, 11, 12])
```

In [152…

```
p1[1,2:]
```

Out[152…

```
array([7, 8])
```

In [153…

```
p2
```

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…

```
p2[2,1:]
```

Out[154…

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

In [155…

```
p3=np.arange(0,100).reshape(10,10)
p3
```

Out[155…

```
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, 27, 28, 29],
       [30, 31, 32, 33, 34, 35, 36, 37, 38, 39],
       [40, 41, 42, 43, 44, 45, 46, 47, 48, 49],
       [50, 51, 52, 53, 54, 55, 56, 57, 58, 59],
       [60, 61, 62, 63, 64, 65, 66, 67, 68, 69],
       [70, 71, 72, 73, 74, 75, 76, 77, 78, 79],
       [80, 81, 82, 83, 84, 85, 86, 87, 88, 89],
       [90, 91, 92, 93, 94, 95, 96, 97, 98, 99]])
```

In [156…

```
p3[-3,:2]
```

Out[156…

```
array([70, 71])
```

In [157…

```
p3[-3,:-3]
```

Out[157…

```
array([70, 71, 72, 73, 74, 75, 76])
```

```
In [158…  p3[2,:5]

Out[158…  array([20, 21, 22, 23, 24])

In [159…  p3[:5]

Out[159…  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, 27, 28, 29],
                 [30, 31, 32, 33, 34, 35, 36, 37, 38, 39],
                 [40, 41, 42, 43, 44, 45, 46, 47, 48, 49]])

In [160…  p3[-3:]

Out[160…  array([[70, 71, 72, 73, 74, 75, 76, 77, 78, 79],
                 [80, 81, 82, 83, 84, 85, 86, 87, 88, 89],
                 [90, 91, 92, 93, 94, 95, 96, 97, 98, 99]])

In [161…  p3[:-3]

Out[161…  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, 27, 28, 29],
                 [30, 31, 32, 33, 34, 35, 36, 37, 38, 39],
                 [40, 41, 42, 43, 44, 45, 46, 47, 48, 49],
                 [50, 51, 52, 53, 54, 55, 56, 57, 58, 59],
                 [60, 61, 62, 63, 64, 65, 66, 67, 68, 69]])

In [162…  p3[-1,:3]

Out[162…  array([90, 91, 92])

In [163…  p3[-1,:-1]

Out[163…  array([90, 91, 92, 93, 94, 95, 96, 97, 98])

In [164…  p3[-1,-1:]

Out[164…  array([99])
```

Transpose(converting rows into columns and columns into rows)

```
In [166…  z

Out[166…  array([[ 0,  1,  2,  3],
                 [ 4,  5,  6,  7],
                 [ 8,  9, 10, 11]])

In [167…  np.transpose(z)

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

In [168…  z1
```

```
Out[168…  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, 27, 28, 29],
                 [30, 31, 32, 33, 34],
                 [35, 36, 37, 38, 39],
                 [40, 41, 42, 43, 44],
                 [45, 46, 47, 48, 49]])
```

```
In [169…  a
```

```
Out[169…  array([4, 2, 5, 7, 9, 8, 2])
```

```
In [170…  np.ravel(z)
```

```
Out[170…  array([ 0,  1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11])
```

```
In [171…  np.ravel(z1)
```

```
Out[171…  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, 27, 28, 29, 30, 31, 32, 33,
                 34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49])
```

```
In [ ]:
```