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

arr1 = np.array([1,2,3])

print(arr1)
```
```
[1 2 3]
```
In [2]:
```
print(type(arr1))
```
```
<class 'numpy.ndarray'>
```
In [3]:
```
print(arr1.shape)
```
```
(3,)
```
In [4]:
```
print(arr1[2])
```
```
3
```
In [5]:
```
arr1[2]=5
arr1
```
Out[5]:
```
array([1, 2, 5])
```
In [7]:
```
arr2=np.array([[1,2,3],[3,4,5]])
arr2
```
Out[7]:
```
array([[1, 2, 3],
       [3, 4, 5]])
```
In [8]:
```
arr2[1][2]
```
Out[8]:
```
5
```
In [9]:
```
print(arr2.shape)
```
```
(2, 3)
```
In [10]:
```
arr2[1,2]
```
Out[10]:
```
5
```
In [13]:
```
arr2[1][-2]
```
Out[13]:
```
4
```
In [15]:
```
arrS = np.array(['China','India'])
arrS
```
Out[15]:
```
array(['China', 'India'], dtype='<U5')
```
In [17]:
```
arrR = np.arange(0,20,2)
arrR
```
Out[17]:
```
array([ 0,  2,  4,  6,  8, 10, 12, 14, 16, 18])
```
In [18]:
```
arrL = np.linspace(0,10,20)
arrL
```
Out[18]:
```
array([ 0.        ,  0.52631579,  1.05263158,  1.57894737,  2.10526316,
        2.63157895,  3.15789474,  3.68421053,  4.21052632,  4.73684211,
        5.26315789,  5.78947368,  6.31578947,  6.84210526,  7.36842105,
        7.89473684,  8.42105263,  8.94736842,  9.47368421, 10.        ])
```
In [19]:
```
arr=np.random.rand(10)
arr
```
Out[19]:
```
array([0.29531706, 0.61900339, 0.39096234, 0.1576005 , 0.19316195,
       0.93379078, 0.42747038, 0.47964894, 0.74195925, 0.29753461])
```
In [20]:
```
arr = np.random.rand(3,4)
arr
```
Out[20]:
```
array([[0.87574535, 0.45755144, 0.90474273, 0.67299808],
       [0.20745002, 0.43367441, 0.45232681, 0.85162852],
       [0.21962735, 0.31533464, 0.81022012, 0.20322926]])
```
In [22]:
```
print(np.zeros(10))
print('/n')
print(np.zeros((2,3)))
```
```
[0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
/n
[[0. 0. 0.]
 [0. 0. 0.]]
```
In [24]:
```
print(np.ones((10,10,3)))
```
```
[[[1. 1. 1.]
  [1. 1. 1.]
  [1. 1. 1.]
  [1. 1. 1.]
  [1. 1. 1.]
  [1. 1. 1.]
  [1. 1. 1.]
  [1. 1. 1.]
  [1. 1. 1.]
  [1. 1. 1.]]

 [[1. 1. 1.]
  [1. 1. 1.]
  [1. 1. 1.]
  [1. 1. 1.]
  [1. 1. 1.]
  [1. 1. 1.]
  [1. 1. 1.]
  [1. 1. 1.]
  [1. 1. 1.]
  [1. 1. 1.]]

 [[1. 1. 1.]
  [1. 1. 1.]
  [1. 1. 1.]
  [1. 1. 1.]
  [1. 1. 1.]
  [1. 1. 1.]
  [1. 1. 1.]
  [1. 1. 1.]
  [1. 1. 1.]
  [1. 1. 1.]]

 [[1. 1. 1.]
  [1. 1. 1.]
  [1. 1. 1.]
  [1. 1. 1.]
  [1. 1. 1.]
  [1. 1. 1.]
  [1. 1. 1.]
  [1. 1. 1.]
  [1. 1. 1.]
  [1. 1. 1.]]

 [[1. 1. 1.]
  [1. 1. 1.]
  [1. 1. 1.]
  [1. 1. 1.]
  [1. 1. 1.]
  [1. 1. 1.]
  [1. 1. 1.]
  [1. 1. 1.]
  [1. 1. 1.]
  [1. 1. 1.]]

 [[1. 1. 1.]
  [1. 1. 1.]
  [1. 1. 1.]
  [1. 1. 1.]
  [1. 1. 1.]
  [1. 1. 1.]
  [1. 1. 1.]
  [1. 1. 1.]
  [1. 1. 1.]
  [1. 1. 1.]]

 [[1. 1. 1.]
  [1. 1. 1.]
  [1. 1. 1.]
  [1. 1. 1.]
  [1. 1. 1.]
  [1. 1. 1.]
  [1. 1. 1.]
  [1. 1. 1.]
  [1. 1. 1.]
  [1. 1. 1.]]

 [[1. 1. 1.]
  [1. 1. 1.]
  [1. 1. 1.]
  [1. 1. 1.]
  [1. 1. 1.]
  [1. 1. 1.]
  [1. 1. 1.]
  [1. 1. 1.]
  [1. 1. 1.]
  [1. 1. 1.]]

 [[1. 1. 1.]
  [1. 1. 1.]
  [1. 1. 1.]
  [1. 1. 1.]
  [1. 1. 1.]
  [1. 1. 1.]
  [1. 1. 1.]
  [1. 1. 1.]
  [1. 1. 1.]
  [1. 1. 1.]]

 [[1. 1. 1.]
  [1. 1. 1.]
  [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 [25]:
```
arr =[0,1,2]
print(np.repeat(arr,3))
```
```
[0 0 0 1 1 1 2 2 2]
```
In [26]:
```
print(np.tile(arr,3))
```
```
[0 1 2 0 1 2 0 1 2]
```
In [27]:
```
identity_matrix = np.eye(3)
identity_matrix
```
Out[27]:
```
array([[1., 0., 0.],
       [0., 1., 0.],
       [0., 0., 1.]])
```
In [28]:
```
np.diag([1,2,3,4,5])
```
Out[28]:
```
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]])
```
In [30]:
```
arr =np.random.rand(5,5)
arr
```
Out[30]:
```
array([[0.4297257 , 0.79545942, 0.69995775, 0.87309865, 0.99333168],
       [0.29143136, 0.17683406, 0.09118351, 0.34197015, 0.87798396],
       [0.73802835, 0.96382742, 0.38443617, 0.45886988, 0.22978325],
       [0.35396299, 0.16167098, 0.04142867, 0.252322  , 0.3367416 ],
       [0.3452353 , 0.39091279, 0.44837455, 0.62470269, 0.0597782 ]])
```
In [31]:
```
np.diag(arr)
```
Out[31]:
```
array([0.4297257 , 0.17683406, 0.38443617, 0.252322  , 0.0597782 ])
```
In [32]:
```
arr.ndim
```
Out[32]:
```
2
```
In [33]:
```
arr.size
```
Out[33]:
```
25
```
In [34]:
```
arr.shape
```
Out[34]:
```
(5, 5)
```
In [35]:
```
np.random.randint(-10,10,4)
```
Out[35]:
```
array([ -4, -10,   5,   2])
```
In [36]:
```
arr+10
```
Out[36]:
```
array([[10.4297257 , 10.79545942, 10.69995775, 10.87309865, 10.99333168],
       [10.29143136, 10.17683406, 10.09118351, 10.34197015, 10.87798396],
       [10.73802835, 10.96382742, 10.38443617, 10.45886988, 10.22978325],
       [10.35396299, 10.16167098, 10.04142867, 10.252322  , 10.3367416 ],
       [10.3452353 , 10.39091279, 10.44837455, 10.62470269, 10.0597782 ]])
```
In [37]:
```
arr-10
```
Out[37]:
```
array([[-9.5702743 , -9.20454058, -9.30004225, -9.12690135, -9.00666832],
       [-9.70856864, -9.82316594, -9.90881649, -9.65802985, -9.12201604],
       [-9.26197165, -9.03617258, -9.61556383, -9.54113012, -9.77021675],
       [-9.64603701, -9.83832902, -9.95857133, -9.747678  , -9.6632584 ],
       [-9.6547647 , -9.60908721, -9.55162545, -9.37529731, -9.9402218 ]])
```
In [38]:
```
arr*2
```
Out[38]:
```
array([[0.85945139, 1.59091884, 1.39991549, 1.7461973 , 1.98666335],
       [0.58286271, 0.35366812, 0.18236703, 0.6839403 , 1.75596792],
       [1.4760567 , 1.92765484, 0.76887233, 0.91773975, 0.45956651],
       [0.70792598, 0.32334196, 0.08285735, 0.504644  , 0.6734832 ],
       [0.6904706 , 0.78182558, 0.8967491 , 1.24940539, 0.1195564 ]])
```
In [39]:
```
arr/2
```
Out[39]:
```
array([[0.21486285, 0.39772971, 0.34997887, 0.43654932, 0.49666584],
       [0.14571568, 0.08841703, 0.04559176, 0.17098508, 0.43899198],
       [0.36901418, 0.48191371, 0.19221808, 0.22943494, 0.11489163],
       [0.17698149, 0.08083549, 0.02071434, 0.126161  , 0.1683708 ],
       [0.17261765, 0.1954564 , 0.22418728, 0.31235135, 0.0298891 ]])
```
In [40]:
```
np.exp(arr)
```
Out[40]:
```
array([[1.5368359 , 2.21545859, 2.01366762, 2.39431852, 2.70021575],
       [1.33834176, 1.19343304, 1.09547002, 1.40771828, 2.40604413],
       [2.09180713, 2.62171169, 1.46878594, 1.5822848 , 1.25832724],
       [1.42470246, 1.17547342, 1.04229882, 1.28701038, 1.40037716],
       [1.4123222 , 1.47832959, 1.56576504, 1.8676906 , 1.06160106]])
```
In [41]:
```
arr
```
Out[41]:
```
array([[0.4297257 , 0.79545942, 0.69995775, 0.87309865, 0.99333168],
       [0.29143136, 0.17683406, 0.09118351, 0.34197015, 0.87798396],
       [0.73802835, 0.96382742, 0.38443617, 0.45886988, 0.22978325],
       [0.35396299, 0.16167098, 0.04142867, 0.252322  , 0.3367416 ],
       [0.3452353 , 0.39091279, 0.44837455, 0.62470269, 0.0597782 ]])
```
In [42]:
```
np.log(arr)
```
Out[42]:
```
array([[-0.84460819, -0.22883545, -0.35673531, -0.13570673, -0.00669066],
       [-1.23295078, -1.73254351, -2.39488117, -1.07303182, -0.13012695],
       [-0.30377304, -0.03684302, -0.95597752, -0.7789886 , -1.47061879],
       [-1.03856292, -1.82219201, -3.18378204, -1.37704924, -1.08843941],
       [-1.06352907, -0.93927078, -0.80212634, -0.47047943, -2.8171142 ]])
```
In [43]:
```
np.log2(arr)
```
Out[43]:
```
array([[-1.21851205, -0.33013976, -0.51466026, -0.19578343, -0.00965258],
       [-1.77877198, -2.49953192, -3.45508318, -1.54805769, -0.18773351],
       [-0.43825186, -0.05315325, -1.37918403, -1.123843  , -2.12165443],
       [-1.49832957, -2.62886737, -4.59322656, -1.98666211, -1.57028614],
       [-1.53434811, -1.3550813 , -1.1572237 , -0.67875835, -4.06423668]])
```
In [44]:
```
np.cos(arr)
```
Out[44]:
```
array([[0.90908006, 0.69995673, 0.76486941, 0.64245507, 0.54590145],
       [0.95783359, 0.98440556, 0.99584566, 0.94209582, 0.63870369],
       [0.73979658, 0.5703804 , 0.92701004, 0.89655364, 0.97371579],
       [0.93800644, 0.98695969, 0.99914196, 0.96833534, 0.94383629],
       [0.94099585, 0.92456164, 0.90115293, 0.81113704, 0.99821382]])
```
In [45]:
```
np.sin(arr)
```
Out[45]:
```
array([[0.41662145, 0.71418525, 0.64418537, 0.76632336, 0.83784939],
       [0.28732352, 0.17591389, 0.09105721, 0.33534381, 0.76945279],
       [0.6728306 , 0.82138067, 0.37503652, 0.44293517, 0.22776648],
       [0.34661783, 0.16096762, 0.04141682, 0.2496531 , 0.33041346],
       [0.3384181 , 0.38103251, 0.43350133, 0.58485614, 0.05974261]])
```
In [46]:
```
np.tan(arr)
```
Out[46]:
```
array([[0.45828907, 1.02032772, 0.84221615, 1.19280459, 1.53479971],
       [0.29997227, 0.17870063, 0.09143707, 0.3559551 , 1.20471011],
       [0.90948055, 1.44005766, 0.40456576, 0.49404202, 0.23391475],
       [0.36952607, 0.16309442, 0.04145239, 0.25781678, 0.35007496],
       [0.35963825, 0.41212234, 0.4810519 , 0.72103247, 0.05984951]])
```
In [47]:
```
arr
```
Out[47]:
```
array([[0.4297257 , 0.79545942, 0.69995775, 0.87309865, 0.99333168],
       [0.29143136, 0.17683406, 0.09118351, 0.34197015, 0.87798396],
       [0.73802835, 0.96382742, 0.38443617, 0.45886988, 0.22978325],
       [0.35396299, 0.16167098, 0.04142867, 0.252322  , 0.3367416 ],
       [0.3452353 , 0.39091279, 0.44837455, 0.62470269, 0.0597782 ]])
```
In [48]:
```
np.sum(arr)
```
Out[48]:
```
11.36105107095126
```
In [49]:
```
np.sum(arr, axis=0)
```
Out[49]:
```
array([2.15838369, 2.48870467, 1.66538065, 2.55096337, 2.49761868])
```
In [50]:
```
np.sum(arr, axis=1)
```
Out[50]:
```
array([3.79157319, 1.77940304, 2.77494507, 1.14612624, 1.86900354])
```
In [51]:
```
np.min(arr)
```
Out[51]:
```
0.041428673615771516
```
In [52]:
```
np.min(arr, axis=0)
```
Out[52]:
```
array([0.29143136, 0.16167098, 0.04142867, 0.252322  , 0.0597782 ])
```
In [53]:
```
np.max(arr)
```
Out[53]:
```
0.9933316762523227
```
In [54]:
```
np.max(arr, axis=1)
```
Out[54]:
```
array([0.99333168, 0.87798396, 0.96382742, 0.35396299, 0.62470269])
```
In [55]:
```
np.mean(arr,axis=0)
```
Out[55]:
```
array([0.43167674, 0.49774093, 0.33307613, 0.51019267, 0.49952374])
```
In [56]:
```
np.median(arr, axis=0)
```
Out[56]:
```
array([0.35396299, 0.39091279, 0.38443617, 0.45886988, 0.3367416 ])
```
In [57]:
```
np.std(arr)
```
Out[57]:
```
0.281966018580189
```
In [58]:
```
np.var(arr)
```
Out[58]:
```
0.07950483563396349
```
In [59]:
```
arr[1:,2:3]
```
Out[59]:
```
array([[0.09118351],
       [0.38443617],
       [0.04142867],
       [0.44837455]])
```
In [60]:
```
arr[1:,2:4]
```
Out[60]:
```
array([[0.09118351, 0.34197015],
       [0.38443617, 0.45886988],
       [0.04142867, 0.252322  ],
       [0.44837455, 0.62470269]])
```
In [61]:
```
arr[1:,2:4-1]
```
Out[61]:
```
array([[0.09118351],
       [0.38443617],
       [0.04142867],
       [0.44837455]])
```
In [62]:
```
arr[1:,2:-1]
```
Out[62]:
```
array([[0.09118351, 0.34197015],
       [0.38443617, 0.45886988],
       [0.04142867, 0.252322  ],
       [0.44837455, 0.62470269]])
```
In [64]:
```
np.sort(arr, axis=0)
```
Out[64]:
```
array([[0.29143136, 0.16167098, 0.04142867, 0.252322  , 0.0597782 ],
       [0.3452353 , 0.17683406, 0.09118351, 0.34197015, 0.22978325],
       [0.35396299, 0.39091279, 0.38443617, 0.45886988, 0.3367416 ],
       [0.4297257 , 0.79545942, 0.44837455, 0.62470269, 0.87798396],
       [0.73802835, 0.96382742, 0.69995775, 0.87309865, 0.99333168]])
```
In [65]:
```
arr
```
Out[65]:
```
array([[0.4297257 , 0.79545942, 0.69995775, 0.87309865, 0.99333168],
       [0.29143136, 0.17683406, 0.09118351, 0.34197015, 0.87798396],
       [0.73802835, 0.96382742, 0.38443617, 0.45886988, 0.22978325],
       [0.35396299, 0.16167098, 0.04142867, 0.252322  , 0.3367416 ],
       [0.3452353 , 0.39091279, 0.44837455, 0.62470269, 0.0597782 ]])
```
In [66]:
```
arr.T
```
Out[66]:
```
array([[0.4297257 , 0.29143136, 0.73802835, 0.35396299, 0.3452353 ],
       [0.79545942, 0.17683406, 0.96382742, 0.16167098, 0.39091279],
       [0.69995775, 0.09118351, 0.38443617, 0.04142867, 0.44837455],
       [0.87309865, 0.34197015, 0.45886988, 0.252322  , 0.62470269],
       [0.99333168, 0.87798396, 0.22978325, 0.3367416 , 0.0597782 ]])
```
In [67]:
```
arr[:3,:].T
```
Out[67]:
```
array([[0.4297257 , 0.29143136, 0.73802835],
       [0.79545942, 0.17683406, 0.96382742],
       [0.69995775, 0.09118351, 0.38443617],
       [0.87309865, 0.34197015, 0.45886988],
       [0.99333168, 0.87798396, 0.22978325]])
```
In [68]:
```
arr[:3,:].transpose()
```
Out[68]:
```
array([[0.4297257 , 0.29143136, 0.73802835],
       [0.79545942, 0.17683406, 0.96382742],
       [0.69995775, 0.09118351, 0.38443617],
       [0.87309865, 0.34197015, 0.45886988],
       [0.99333168, 0.87798396, 0.22978325]])
```
In [70]:
```
arr[:3,:].flatten()
```
Out[70]:
```
array([0.4297257 , 0.79545942, 0.69995775, 0.87309865, 0.99333168,
       0.29143136, 0.17683406, 0.09118351, 0.34197015, 0.87798396,
       0.73802835, 0.96382742, 0.38443617, 0.45886988, 0.22978325])
```
In [71]:
```
arr = np.array([4,5,6,7,8])
arr
```
Out[71]:
```
array([4, 5, 6, 7, 8])
```
In [73]:
```
arr1=np.append(arr,8)
arr1
```
Out[73]:
```
array([4, 5, 6, 7, 8, 8])
```
In [74]:
```
arr2= np.insert(arr, 0,[1,2,3])
arr2
```
Out[74]:
```
array([1, 2, 3, 4, 5, 6, 7, 8])
```
In [75]:
```
arr2 =np.insert(arr,1,[1,2,3])
arr2
```
Out[75]:
```
array([4, 1, 2, 3, 5, 6, 7, 8])
```
In [76]:
```
arr2= np.insert(arr,-1,[1,2,3])
arr2
```
Out[76]:
```
array([4, 5, 6, 7, 1, 2, 3, 8])
```
In [77]:
```
arr3= np.delete(arr2, 0)
arr3
```
Out[77]:
```
array([5, 6, 7, 1, 2, 3, 8])
```
In [78]:
```
arr3= np.delete(arr2, [1,3])
arr3
```
Out[78]:
```
array([4, 6, 1, 2, 3, 8])
```
In [80]:
```
arrC =arr3.copy()
arrC
```
Out[80]:
```
array([4, 6, 1, 2, 3, 8])
```
In [83]:
```
arr1 = np.array([[1,2,3,4],[1,2,3,4]])
arr2= np.array([[5,6,7,8],[5,6,7,8]])
cat= np.concatenate((arr1,arr2), axis=0)
print(cat)
```
```
[[1 2 3 4]
 [1 2 3 4]
 [5 6 7 8]
 [5 6 7 8]]
```
In [82]:
```
arr1 = np.array([[1,2,3,4],[1,2,3,4]])
arr2= np.array([[5,6,7,8],[5,6,7,8]])
cat= np.concatenate((arr1,arr2), axis=1)
print(cat)
```
```
[[1 2 3 4 5 6 7 8]
 [1 2 3 4 5 6 7 8]]
```
In [84]:
```
catV =np.vstack((arr1, arr2))
catV
```
Out[84]:
```
array([[1, 2, 3, 4],
       [1, 2, 3, 4],
       [5, 6, 7, 8],
       [5, 6, 7, 8]])
```
In [85]:
```
catM = np.hstack((arr1, arr2))
catM
```
Out[85]:
```
array([[1, 2, 3, 4, 5, 6, 7, 8],
       [1, 2, 3, 4, 5, 6, 7, 8]])
```
In [86]:
```
arr = np.array([1,2,3,4,5,6,3,6,1,2,4])
np.unique(arr)
```
Out[86]:
```
array([1, 2, 3, 4, 5, 6])
```
In [88]:
```
uniques,counts= np.unique(arr, return_counts=True)
print(uniques)
print(counts)
```
```
[1 2 3 4 5 6]
[2 2 2 2 1 2]
```
In [92]:
```
arr1=np.array([1,2,3,4,5])
arr2=np.array([3,4,5,6,7])
print(np.intersect1d(arr1,arr2))
print(np.union1d(arr1,arr2))
print(np.setdiff1d(arr1,arr2))
print(np.setxor1d(arr1,arr2))
```
```
[3 4 5]
[1 2 3 4 5 6 7]
[1 2]
[1 2 6 7]
```
In [ ]:
```

```