Practical 3

Aim : Perform experiments using NumPy.

In [1]:
```
print("hello world")
```

```
hello world
```

In [2]:
```
import numpy as np
```

In [3]:
```
arr = np.array([1,2,3,4,5])
```

In [4]:
```
arr
```

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

In [5]:
```
arr.ndim
```

Out[5]:
```
1
```

In [6]:
```
np.__version__
```

Out[6]:
```
'1.26.4'
```

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

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

In [8]:
```
arr = np.array(42)
```

In [9]:
```
arr.ndim
```

Out[9]:
```
0
```

In [10]:
```
arr
```

Out[10]:
```
array(42)
```

In [11]:
```
arr = np.array([[1,2,3],[4,5,6]])
print(arr)
```

```
[[1 2 3]
 [4 5 6]]
```

In [12]:
```
arr.ndim
```

Out[12]:
```
2
```

In [13]:
```
arr = np.array([[[1,2,3], [4,5,6]],[[1,2,3],[4,5,6]]])
```

```
arr
```

Out[13]:
```
array([[[1, 2, 3],
        [4, 5, 6]],

       [[1, 2, 3],
        [4, 5, 6]]])
```

In [14]:
```
arr.ndim
```

Out[14]:
```
3
```

In [15]:
```
my_array = np.arange(1000000)
```

In [16]:
```
my_list = list(range(1000000))
```

In [17]:
```
# Lets mulktiply each sequence by 2
%time for _ in range(10) : my_array2 = my_array * 2
```

```
CPU times: total: 0 ns
Wall time: 8 ms
```

In [18]:
```
%time for _ in range(10): my_list2 = [x*2 for x in my_list]
```

```
CPU times: total: 109 ms
Wall time: 334 ms
```

In [19]:
```
import numpy as np
```

In [20]:
```
data = np.random.randn(2,3)
data
```

Out[20]:
```
array([[ 0.73729933, -0.34247682, -2.04584446],
       [ 0.97486241,  2.12348854,  1.29192561]])
```

In [21]:
```
data*10
```

Out[21]:
```
array([[  7.3729933 ,  -3.42476824, -20.4584446 ],
       [  9.74862412,  21.23488545,  12.91925608]])
```

In [22]:
```
data +data
```

Out[22]:
```
array([[ 1.47459866, -0.68495365, -4.09168892],
       [ 1.94972482,  4.24697709,  2.58385122]])
```

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

Out[23]:
```
(2, 3)
```

In [24]:
```
data.dtype
```

Out[24]:
```
dtype('float64')
```

In [25]:
```
arr1 = [6,7.5,8,0,1]
```

In [26]:
```
arr1 = np.array(arr1)
```

In [27]:
```
arr1
```

Out[27]:
```
array([6. , 7.5, 8. , 0. , 1. ])
```

In [28]:
```
np.zeros(10)
```

Out[28]:
```
array([0., 0., 0., 0., 0., 0., 0., 0., 0., 0.])
```

In [29]:
```
np.zeros((3,6))
```

Out[29]:
```
array([[0., 0., 0., 0., 0., 0.],
       [0., 0., 0., 0., 0., 0.],
       [0., 0., 0., 0., 0., 0.]])
```

In [30]:
```
np.zeros((2,3,2))
```

Out[30]:
```
array([[[0., 0.],
        [0., 0.],
        [0., 0.]],

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

In [31]:
```
np.empty((2,3,2))
```

Out[31]:
```
array([[[0., 0.],
        [0., 0.],
        [0., 0.]],

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

In [32]:
```
arr1 = np.array([1,2,3], dtype=np.int64)
arr1.dtype
```

Out[32]:
```
dtype('int64')
```

In [33]:
```
arr = np.array([3.7,-1.2,-2.6,.5,12.9,10.1])
arr
```

Out[33]:
```
array([ 3.7, -1.2, -2.6,  0.5, 12.9, 10.1])
```

In [34]:
```
arr.astype(np.int32)
```

Out[34]:
```
array([ 3, -1, -2,  0, 12, 10])
```

In [35]:
```
numeric_string = np.array(['1.25', '-9.6', '42'], dtype= np.bytes_)
```

In [36]:
```
numeric_string.astype(float)
```

Out[36]:
```
array([ 1.25, -9.6 , 42.  ])
```

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

In [38]:
```
calibers = np.array([.22,.270, .357, .380, .44, .50], dtype=np.float64)
```

In [39]:
```
int_array.astype(calibers.dtype)
```

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

In [40]:
```
empty_unit32 = np.empty(8,dtype='u4')
empty_unit32
```

Out[40]:
```
array([         0, 1075314688,          0, 1075707904,          0,
       1075838976,          0, 1072693248], dtype=uint32)
```

In [41]:
```
arr = np.array([[1.,2.,3.], [4.,5.,6.], [7.,8.,9.]])
arr
```

Out[41]:
```
array([[1., 2., 3.],
       [4., 5., 6.],
       [7., 8., 9.]])
```

In [42]:
```
arr**2
```

Out[42]:
```
array([[ 1.,  4.,  9.],
       [16., 25., 36.],
       [49., 64., 81.]])
```

In [43]:
```
arr - arr
```

Out[43]:
```
array([[0., 0., 0.],
       [0., 0., 0.],
       [0., 0., 0.]])
```

In [44]:
```
1/arr
```

Out[44]:
```
array([[1.        , 0.5       , 0.33333333],
       [0.25      , 0.2       , 0.16666667],
       [0.14285714, 0.125     , 0.11111111]])
```

In [45]:
```
arr**.5
```

Out[45]:
```
array([[1.        , 1.41421356, 1.73205081],
       [2.        , 2.23606798, 2.44948974],
       [2.64575131, 2.82842712, 3.        ]])
```

In [46]:
```
arr2 = np.array([[0.,4.,1.],[7.,2.,12.], [3., 5., 6.]])
arr2
```

Out[46]:
```
array([[ 0.,  4.,  1.],
       [ 7.,  2., 12.],
       [ 3.,  5.,  6.]])
```

In [47]:
```
arr2>arr
```

Out[47]:
```
array([[False,  True, False],
       [ True, False,  True],
       [False, False, False]])
```

Basic indexing and sliceing

In [48]:
```
arr = np.arange(10)
arr
```

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

In [49]:
```
arr[5]
```

Out[49]:
```
5
```

In [50]:
```
arr[5:8]
```

Out[50]:
```
array([5, 6, 7])
```

In [51]:
```
arr[5:8] =12
arr
```

Out[51]:
```
array([ 0,  1,  2,  3,  4, 12, 12, 12,  8,  9])
```

In [52]:
```
arr_slice = arr[5:8]
```

In [53]:
```
arr_slice[1] = 12345
```

In [54]:
```
arr
```

Out[54]:
```
array([    0,     1,     2,     3,     4,    12, 12345,    12,     8,
           9])
```

In [55]:
```
arr_slice[:] = 64
arr
```

Out[55]:
```
array([ 0,  1,  2,  3,  4, 64, 64, 64,  8,  9])
```

In [56]:
```
arr2d = np.array([[1,2,3],[4,5,6], [7,8,9]])
print(arr2d[2])
print(arr2d[0][2])
print(arr2d[0,2])
```

```
[7 8 9]
3
3
```

In [57]:
```
names = np.array(['bob', 'joe', 'will', 'bob', 'will', 'joe', 'joe'])
data = np.random.randn(7,4)
names,data
```

Out[57]:
```
(array(['bob', 'joe', 'will', 'bob', 'will', 'joe', 'joe'], dtype='<U4'),
 array([[-0.21126019,  0.30404047, -0.35679496, -1.57094458],
        [-0.66951276,  0.36073056,  0.75994753, -0.14598681],
        [-0.08487371,  0.68720426, -1.13914444,  0.3454157 ],
        [ 0.5629359 ,  0.15735323,  0.13201485, -0.18493899],
        [-0.06517681,  0.78820599, -0.05679932, -0.03713521],
        [-0.9239707 , -0.6909104 , -0.90994236, -0.9502327 ],
        [-0.37067257, -0.76752753, -0.30612362,  0.64792188]]))
```

In [58]:
```
data[names == 'bob']
```

Out[58]:
```
array([[-0.21126019,  0.30404047, -0.35679496, -1.57094458],
       [ 0.5629359 ,  0.15735323,  0.13201485, -0.18493899]])
```

In [59]:
```
data[~(names == 'bob')]
```

Out[59]:
```
array([[-0.66951276,  0.36073056,  0.75994753, -0.14598681],
       [-0.08487371,  0.68720426, -1.13914444,  0.3454157 ],
       [-0.06517681,  0.78820599, -0.05679932, -0.03713521],
       [-0.9239707 , -0.6909104 , -0.90994236, -0.9502327 ],
       [-0.37067257, -0.76752753, -0.30612362,  0.64792188]])
```

Fancy Index

In [60]:
```
arr = np.empty((8,4))
for i in range(8):
    arr[i] = i
arr
```

Out[60]:
```
array([[0., 0., 0., 0.],
       [1., 1., 1., 1.],
       [2., 2., 2., 2.],
       [3., 3., 3., 3.],
       [4., 4., 4., 4.],
       [5., 5., 5., 5.],
       [6., 6., 6., 6.],
       [7., 7., 7., 7.]])
```

In [61]:
```
arr[[4,3,0,6]]
```

Out[61]:
```
array([[4., 4., 4., 4.],
       [3., 3., 3., 3.],
       [0., 0., 0., 0.],
       [6., 6., 6., 6.]])
```

In [62]:
```
arr = np.arange(32).reshape((8,4))
arr
```

Out[62]:
```
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]])
```

In [63]:
```
arr = np.random.randn(6,3)
```

In [64]:
```
arr
```

Out[64]:
```
array([[ 1.89413492,  0.63078324, -1.15180049],
       [-0.08497186,  1.2867645 , -0.69370577],
       [-0.24475857,  1.53863136,  1.16052707],
       [-1.19266944, -0.86362876,  1.60605148],
       [-0.50059832, -0.76755245,  0.0071669 ],
       [ 0.35785664,  0.20636308, -1.81918847]])
```

In [65]:
```
np.dot(arr.T,arr)
```

Out[65]:
```
array([[ 5.45599452,  2.19696407, -4.97685394],
       [ 2.19696407,  5.79861394, -1.60149563],
       [-4.97685394, -1.60149563,  9.04359459]])
```

In [66]:
```
arr = np.arange(10)
```

In [67]:
```
np.sqrt(arr)
```

Out[67]:
```
array([0.        , 1.        , 1.41421356, 1.73205081, 2.        ,
       2.23606798, 2.44948974, 2.64575131, 2.82842712, 3.        ])
```

In [68]:
```
arr
```

Out[68]:
```
array([0, 1, 2, 3, 4, 5, 6, 7, 8, 9])
```

In [69]:
```
arr = np.arange(16).reshape((2,2,4))
```

In [70]:
```
arr
```

Out[70]:
```
array([[[ 0,  1,  2,  3],
        [ 4,  5,  6,  7]],

       [[ 8,  9, 10, 11],
        [12, 13, 14, 15]]])
```

In [71]:
```
arr.swapaxes(1, 2)
```

Out[71]:
```
array([[[ 0,  4],
        [ 1,  5],
        [ 2,  6],
        [ 3,  7]],

       [[ 8, 12],
        [ 9, 13],
        [10, 14],
        [11, 15]]])
```

In [72]:
```
arr = np.arange(10)
```

In [73]:
```
np.sqrt(arr)
```

Out[73]:
```
array([0.        , 1.        , 1.41421356, 1.73205081, 2.        ,
       2.23606798, 2.44948974, 2.64575131, 2.82842712, 3.        ])
```

In [74]:
```
np.exp(arr)
```

Out[74]:
```
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])
```

In [75]:
```
x = np.random.randn(8)
```

In [76]:
```
y = np.random.randn(8)
```

In [77]:
```
print(x, y)
```

```
[-0.33643913  1.53403476  0.70555564  0.90017905  0.51632283  0.46202404
 -0.09265694 -0.86555154] [-2.91465105 -0.06994805 -0.18774416 -0.03610816 -
1.21923896 -1.74239151
 -0.4196097  -0.83918731]
```

In [78]:
```
np.maximum(x, y)
```

Out[78]:
```
array([-0.33643913,  1.53403476,  0.70555564,  0.90017905,  0.51632283,
        0.46202404, -0.09265694, -0.83918731])
```

In [79]:
```
arr = np.random.randn(7)*5
```

In [80]:
```
arr
```

Out[80]:
```
array([  0.61242317,  -4.28747397,   1.96886949,  -6.28088575,
        -1.49987545,  -4.1204414 , -11.55296847])
```

In [81]:
```
reminder, whole_part = np.modf(arr)
```

In [82]:
```
print(reminder, whole_part)
```

```
[ 0.61242317 -0.28747397  0.96886949 -0.28088575 -0.49987545 -0.1204414
 -0.55296847] [  0.  -4.   1.  -6.  -1.  -4. -11.]
```

In [83]:
```
np.sqrt(arr)
```

```
C:\Users\admin\AppData\Local\Temp\ipykernel_6648\2296558006.py:1: RuntimeWar
ning: invalid value encountered in sqrt
  np.sqrt(arr)
```

Out[83]:
```
array([0.7825747,       nan, 1.4031641,       nan,       nan,       nan,
                nan])
```

In [84]:
```
np.sqrt(arr, arr)
```

```
C:\Users\admin\AppData\Local\Temp\ipykernel_6648\269955669.py:1: RuntimeWarn
ing: invalid value encountered in sqrt
  np.sqrt(arr, arr)
```

Out[84]:
```
array([0.7825747,       nan, 1.4031641,       nan,       nan,       nan,
                nan])
```

In [85]:
```
points = np.arange(-5, 5 , 0.01)
```

In [87]:
```
xs, ys = np.meshgrid(points, points)
```

In [88]:
```
ys
```

Out[88]:
```
array([[-5.  , -5.  , -5.  , ..., -5.  , -5.  , -5.  ],
       [-4.99, -4.99, -4.99, ..., -4.99, -4.99, -4.99],
       [-4.98, -4.98, -4.98, ..., -4.98, -4.98, -4.98],
       ...,
       [ 4.97,  4.97,  4.97, ...,  4.97,  4.97,  4.97],
       [ 4.98,  4.98,  4.98, ...,  4.98,  4.98,  4.98],
       [ 4.99,  4.99,  4.99, ...,  4.99,  4.99,  4.99]])
```

In [89]:
```
z = np.sqrt(xs**2 + ys**2)
```

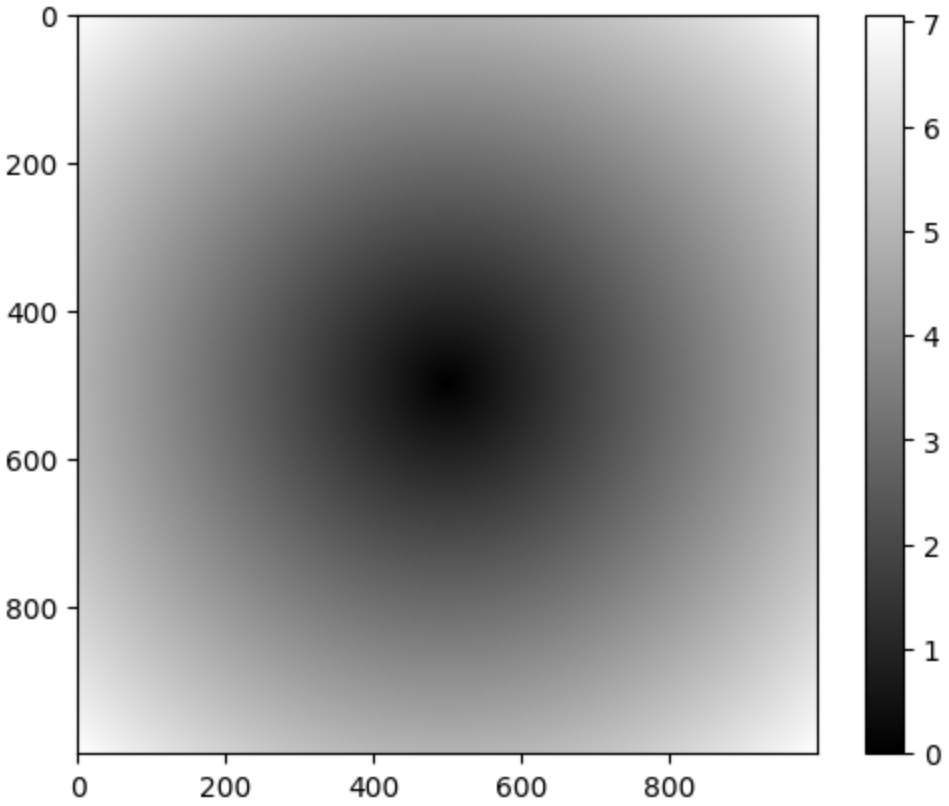
In [90]:
```
z
```

Out[90]:
```
array([[7.07106781, 7.06400028, 7.05693985, ..., 7.04988652, 7.05693985,
        7.06400028],
       [7.06400028, 7.05692568, 7.04985815, ..., 7.04279774, 7.04985815,
        7.05692568],
       [7.05693985, 7.04985815, 7.04278354, ..., 7.03571603, 7.04278354,
        7.04985815],
       ...,
       [7.04988652, 7.04279774, 7.03571603, ..., 7.0286414 , 7.03571603,
        7.04279774],
       [7.05693985, 7.04985815, 7.04278354, ..., 7.03571603, 7.04278354,
        7.04985815],
       [7.06400028, 7.05692568, 7.04985815, ..., 7.04279774, 7.04985815,
        7.05692568]])
```

In [91]:
```
import matplotlib.pyplot as plt
```

In [92]:
```
plt.imshow(z, cmap=plt.cm.gray); plt.colorbar()
```

Out[92]:
```
<matplotlib.colorbar.Colorbar at 0x1ec095edcd0>
```



In [93]:
```
xarr  = np.array([1.1, 1.2, 1.3, 1.4, 1.5])
```

In [94]:
```
yarr = np.array([2.1, 2.2, 2.3, 2.4, 2.5])
```

In [95]:
```
cond = np.array([True, False, True, True, False])
```

In [96]:
```
result = [(x if c else y)
          for x, y, c in zip(xarr, yarr, cond)]
```

In [97]:
```
result
```

Out[97]:
```
[1.1, 2.2, 1.3, 1.4, 2.5]
```

In [98]:
```
result = np.where(cond, xarr, yarr)
```

In [99]:
```
result
```

Out[99]:
```
array([1.1, 2.2, 1.3, 1.4, 2.5])
```

In [100…
```
arr = np.random.randn(4,4)
```

In [101…
```
arr>0
```

Out[101…
```
array([[ True, False,  True,  True],
       [ True,  True, False,  True],
       [ True, False,  True, False],
       [ True, False, False,  True]])
```

In [102…
```
np.where(arr>0, 2, -2)
```

Out[102…
```
array([[ 2, -2,  2,  2],
       [ 2,  2, -2,  2],
       [ 2, -2,  2, -2],
       [ 2, -2, -2,  2]])
```

In [103…
```
np.where(arr>0, 2, arr)
```

Out[103…
```
array([[ 2.        , -0.12588355,  2.        ,  2.        ],
       [ 2.        ,  2.        , -0.88037671,  2.        ],
       [ 2.        , -2.42040688,  2.        , -0.00592823],
       [ 2.        , -0.7619951 , -0.7160243 ,  2.        ]])
```

Mathematical and statistic methods

In [104…
```
arr = np.random.randn(5,4)
arr
```

Out[104…
```
array([[ 0.54837999,  0.34838494, -0.42683956,  1.00405538],
       [ 0.18785443, -1.38222706, -0.97054305,  0.67350827],
       [-0.51885585, -0.05072737,  0.46947371,  0.30844021],
       [-0.94526681, -0.8466369 ,  0.52898741, -0.20707469],
       [-1.60089876,  0.33449966,  0.29670468, -0.68450724]])
```

In [105…
```
arr.mean()
```

Out[105…
```
-0.14666442978433908
```

In [106…
```
np.mean(arr)
```

Out[106…
```
-0.14666442978433908
```

In [107…
```
arr.sum()
```

Out[107…
```
-2.933288595686782
```

In [108…
```
arr.mean(axis=1)
```

Out[108…
```
array([ 0.36849519, -0.37285185,  0.05208267, -0.36749775, -0.41355041])
```

In [109…
```
arr
```

Out[109…
```
array([[ 0.54837999,  0.34838494, -0.42683956,  1.00405538],
       [ 0.18785443, -1.38222706, -0.97054305,  0.67350827],
       [-0.51885585, -0.05072737,  0.46947371,  0.30844021],
       [-0.94526681, -0.8466369 ,  0.52898741, -0.20707469],
       [-1.60089876,  0.33449966,  0.29670468, -0.68450724]])
```

In [110…
```
arr.sum(axis = 0)
```

Out[110…
```
array([-2.32878699, -1.59670673, -0.10221681,  1.09442194])
```

In [111…
```
arr = np.array([0, 1, 2, 3, 4, 5, 6, 7])
```

In [112…
```
arr.cumsum()
```

Out[112…
```
array([ 0,  1,  3,  6, 10, 15, 21, 28])
```

In [113…
```
arr = np.array([[0,1,2], [3,4,5],[6,7,8]])
```

In [114…
```
arr
```

Out[114…
```
array([[0, 1, 2],
       [3, 4, 5],
       [6, 7, 8]])
```

In [115…
```
arr.cumsum(axis = 0)
```

Out[115…
```
array([[ 0,  1,  2],
       [ 3,  5,  7],
       [ 9, 12, 15]])
```

In [116…
```
arr.cumprod(axis =1 )
```

Out[116…
```
array([[  0,   0,   0],
       [  3,  12,  60],
       [  6,  42, 336]])
```

In [117…
```
arr = np.random.randn(100)
```

In [118…
```
(arr > 0).sum()
```

Out[118…
```
56
```

In [119…
```
bools = np.array([False, False, True, False])
```

In [120…
```
bools.any()
```

Out[120…
```
True
```

In [121…
```
bools.all()
```

Out[121…
```
False
```

In [122…
```
arr = np.random.randn(6)
```

In [123…
```
arr
```

Out[123…
```
array([-0.13504742, -0.4328896 ,  0.08317906,  0.13127486, -0.79592489,
       -0.20827801])
```

In [124…
```
arr.sort()
```

In [125…
```
arr
```

Out[125…
```
array([-0.79592489, -0.4328896 , -0.20827801, -0.13504742,  0.08317906,
        0.13127486])
```

In [126…
```
arr = np.random.randn(5,3)
```

In [127…
```
arr
```

Out[127…
```
array([[ 0.71977922, -1.04682355, -1.6952777 ],
       [ 0.39990402,  1.2124756 ,  0.13171366],
       [ 0.09740052, -0.58357185, -1.00226477],
       [-0.09997888,  0.06676777,  1.3745965 ],
       [ 1.10551214,  2.94140787,  2.1043411 ]])
```

In [128…
```
arr.sort(1)
```

In [129…
```
arr
```

Out[129…
```
array([[-1.6952777 , -1.04682355,  0.71977922],
       [ 0.13171366,  0.39990402,  1.2124756 ],
       [-1.00226477, -0.58357185,  0.09740052],
       [-0.09997888,  0.06676777,  1.3745965 ],
       [ 1.10551214,  2.1043411 ,  2.94140787]])
```

In [130…
```
large_arr = np.random.randn(1000)
```

In [131…
```
large_arr.sort()
```

In [132…
```
large_arr[int(0.05*len(large_arr))]
```

Out[132…
```
-1.6612066523439113
```

In [133…
```
names = np.array(['a', 'b','c', 'a', 'c', 'b', 'b'])
```

In [134…
```
np.unique(names)
```

Out[134…
```
array(['a', 'b', 'c'], dtype='<U1')
```

In [135…
```
ints = np.array([3,3,3,2,2,1,1,4,4])
```

In [136…
```
np.unique(ints)
```

Out[136…
```
array([1, 2, 3, 4])
```

In [137…
```
values = np.array([6,0,0,3,2,5,6])
```

In [138…
```
np.in1d(values, [2,3,6])
```

Out[138…
```
array([ True, False, False,  True,  True, False,  True])
```

In [139…
```
arr = np.arange(10)
```

In [140…
```
np.save('some_array',arr)
```

In [141…
```
np.load('some_array.npy')
```

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

In [142…
```
np.savez('array_archive.npz', a = arr, b = arr)
```

In [143…
```
arch = np.load('array_archive.npz')
```

In [144…
```
arch['a']
```

Out[144…
```
array([0, 1, 2, 3, 4, 5, 6, 7, 8, 9])
```

Linear Algebra

In [145…
```
x = np.array([[1,2,3], [4,5,6]])
```

In [146…
```
y = np.array([[6,23], [-1,7], [8,9]])
```

In [147…
```
x
```

Out[147…
```
array([[1, 2, 3],
       [4, 5, 6]])
```

In [148…
```
y
```

Out[148…
```
array([[ 6, 23],
       [-1,  7],
       [ 8,  9]])
```

In [149…
```
x.dot(y)
```

Out[149…
```
array([[ 28,  64],
       [ 67, 181]])
```

In [150…
```
np.dot(x, np.ones(3))
```

Out[150…
```
array([ 6., 15.])
```

In [151…
```
x @ np.ones(3)
```

Out[151…
```
array([ 6., 15.])
```

In [152…
```
x@y
```

Out[152…
```
array([[ 28,  64],
       [ 67, 181]])
```

In [153…
```
from numpy.linalg import inv, qr
```

In [154…
```
x = np.random.randn(5,5)
x
```

Out[154…
```
array([[ 1.38499282, -1.17780607,  0.59438826, -0.70418088,  0.58714006],
       [ 0.69531595, -0.80125079, -1.05907275, -0.01227899, -0.9845574 ],
       [ 0.24754996,  0.30637575, -0.59692091,  1.37087671, -0.96346644],
       [-1.6724927 ,  0.19632887,  1.17759857,  0.72310343,  0.91443157],
       [-1.35831748,  1.26149197, -1.79282903, -0.04445812, -0.99860826]])
```

In [155…
```
mat = x.T.dot(x)
mat
```

Out[155…
```
array([[ 7.1052086 , -4.15439732,  0.40477154, -1.79345973, -0.28285285],
       [-4.15439732,  3.7530031 , -2.06481614,  1.34511297, -1.27804947],
       [ 0.40477154, -2.06481614,  6.43222137, -0.29262611,  4.83398752],
       [-1.79345973,  1.34511297, -0.29262611,  2.90017953, -1.01653229],
       [-0.28285285, -1.27804947,  4.83398752, -1.01653229,  4.07575787]])
```

In [156…
```
x = inv(mat)
x
```

Out[156…
```
array([[  1.4208774 ,   0.45689457,  -4.04405744,   2.21860952,
          5.59160904],
       [  0.45689457,   1.35122499,   1.53193733,  -0.73068529,
         -1.54375333],
       [ -4.04405744,   1.53193733,  20.77554847, -10.60922155,
        -27.08682495],
       [  2.21860952,  -0.73068529, -10.60922155,   5.88360733,
         13.97517012],
       [  5.59160904,  -1.54375333, -27.08682495,  13.97517012,
         35.76075904]])
```

In [157…
```
mat @ x
```

Out[157…
```
array([[ 1.00000000e+00, -5.55111512e-17, -1.77635684e-15,
        -3.10862447e-15, -5.32907052e-15],
       [ 3.55271368e-15,  1.00000000e+00,  0.00000000e+00,
         1.06581410e-14,  7.10542736e-15],
       [ 0.00000000e+00,  8.88178420e-16,  1.00000000e+00,
         1.42108547e-14,  2.84217094e-14],
       [-1.77635684e-15,  2.22044605e-16,  7.10542736e-15,
         1.00000000e+00, -7.10542736e-15],
       [ 0.00000000e+00,  1.77635684e-15, -1.42108547e-14,
        -7.10542736e-15,  1.00000000e+00]])
```

In [158…
```
q,r = qr(mat)
```

In [159…
```
q
```

Out[159…
```
array([[-0.84202817, -0.24536865, -0.46404916, -0.03863061,  0.11810522],
       [ 0.49233172, -0.36810674, -0.73142851,  0.29335309, -0.03260695],
       [-0.0479689 ,  0.69242294, -0.41275868, -0.14336489, -0.57212433],
       [ 0.21254036, -0.05262665, -0.20723199, -0.90663096,  0.29518169],
       [ 0.03352049,  0.56751438, -0.19067222,  0.26442096,  0.75533401]])
```

In [160…
```
r
```

Out[160…
```
array([[-8.43820774,  5.68793894, -1.56610734,  2.76875289, -0.70236804],
       [ 0.        , -2.58797315,  7.87332944, -0.98723081,  6.25357356],
       [ 0.        ,  0.        , -2.19358934, -0.43800203, -1.49568665],
       [ 0.        ,  0.        ,  0.        , -2.39235719,  0.94231835],
       [ 0.        ,  0.        ,  0.        ,  0.        ,  0.02112187]])
```

In [161…
```
sample = np.random.normal(size = (4,4))
```

In [162…
```
sample @ (inv(sample))
```

Out[162…
```
array([[ 1.00000000e+00, -5.55111512e-17,  1.38777878e-17,
         0.00000000e+00],
       [ 0.00000000e+00,  1.00000000e+00, -3.46944695e-17,
         3.46944695e-17],
       [ 4.16333634e-17,  5.65519853e-16,  1.00000000e+00,
        -1.18394877e-16],
       [ 0.00000000e+00,  0.00000000e+00,  1.38777878e-17,
         1.00000000e+00]])
```