```
In [7]: import numpy as np
In [8]: my_arr = np.arange(1000000)
In [9]: my_list = list(range(1000000))
In [10]: %time for _ in range(10): my_arr2 = my_arr * 2
In [11]: %time for _ in range(10): my_list2 = [x * 2 for x in my_list]
```

```
CPU times: user 17.4 ms, sys: 9.16 ms, total: 26.5 ms
Wall time: 37.3 ms
CPU times: user 641 ms, sys: 177 ms, total: 818 ms
Wall time: 819 ms
```

```
In [12]: import numpy as np
# Generate some random data
In [13]: data = np.random.randn(2, 3)
In [14]: data
```

```
array([[-1.43582158, -3.63289298, -0.53025215],
       [ 0.81947882,  1.78215578, -0.60097013]])
```

```
In [15]: data * 10
```

```
array([[ 14.50761561,  10.47227547, -12.05579057],
       [ -3.63298845,  -3.07859023,  10.66889354]])
```

```
In [16]: data + data
```

```
array([[ 2.90152312,  2.09445509, -2.41115811],
       [-0.72659769, -0.61571805,  2.13377871]])
```

```
In [17]: data.shape
```

```
(2, 3)
```

```
In [18]: data.dtype
```

```
In [19]: data1 = [6, 7.5, 8, 0, 1]
In [20]: arr1 = np.array(data1)
In [21]: arr1
```

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

```
In [22]: data2 = [[1, 2, 3, 4], [5, 6, 7, 8]]
In [23]: arr2 = np.array(data2)
In [24]: arr2
```

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


In [25]: arr2.ndim

     2


In [26]: arr2.shape

     (2, 4)


In [29]: np.zeros(10)

     array([0., 0., 0., 0., 0., 0., 0., 0., 0., 0.])


In [30]: np.zeros((3, 6))

     array([[0., 0., 0., 0., 0., 0.],
            [0., 0., 0., 0., 0., 0.],
            [0., 0., 0., 0., 0., 0.]])


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

     array([[[0., 0.],
             [0., 0.],
             [0., 0.]],

            [[0., 0.],
             [0., 0.],
             [0., 0.]],

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


In [32]: np.arange(12)

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


In [37]: arr = np.array([1, 2, 3, 4, 5])
In [38]: arr.dtype


     dtype('int64')


In [39]: float_arr = arr.astype(np.float64)
In [40]: float_arr.dtype
In [40]: float_arr.dtype


     dtype('float64')


In [41]: arr = np.array([3.7, -1.2, -2.6, 0.5, 12.9, 10.1])
In [43]: int_arr = arr.astype(np.int32)
int_arr


     array([ 3, -1, -2,  0, 12, 10], dtype=int32)


In [44]: numeric_strings = np.array(['1.25', '-9.6', '42'])
In [45]: numeric_strings.astype(float)


     array([ 1.25, -9.6 , 42.  ])


In [51]: arr = np.array([[1., 2., 3.], [4., 5., 6.]])
In [53]: arr * arr

     array([[ 1.,  4.,  9.],
            [16., 25., 36.]])


In [54]: arr - arr

     array([[0., 0., 0.],
            [0., 0., 0.]])


1/arr


     array([[1.        , 0.5       , 0.33333333],
            [0.25      , 0.2       , 0.16666667]])


In [56]: arr * 0.5

     array([[0.5, 1. , 1.5],
            [2. , 2.5, 3. ]])


In [57]: arr2 = np.array([[0., 4., 1.], [7., 2., 12.]])
In [59]: arr2 > arr

     array([[False,  True, False],
            [ True, False,  True]])


In [60]: arr = np.arange(10)
In [61]: arr
```

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

```
arr[5]
```

```
     5
```

```
In [63]: arr[5:8]
```

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

```
In [64]: arr[5:8] = 12
arr
```

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

```
In [66]: arr_slice = arr[5:8]
In [67]: arr_slice
```

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

```
In [68]: arr_slice[0] = 12345
In [69]: arr
```

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

```
In [70]: arr_slice[:] = 64
In [71]: arr
```

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

```
In [72]: arr2d = np.array([[1, 2, 3], [4, 5, 6], [7, 8, 9]])
In [73]: arr2d[2]
```

```
     array([7, 8, 9])
```

```
In [74]: arr2d[1][0]
```

```
     4
```

```
In [75]: arr2d[2, 2]
```

```
     9
```

```
In [76]: arr3d = np.array([[[1, 2, 3], [4, 5, 6]], [[7, 8, 9], [10, 11, 12]]])
In [77]: arr3d
```

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

            [[ 7,  8,  9],
             [10, 11, 12]]])
```

```
In [78]: arr3d[1]
```

```
     array([[ 7,  8,  9],
            [10, 11, 12]])
```

```
In [79]: old_values = arr3d[0].copy()
In [80]: arr3d[0] = 42
In [81]: arr3d
```

```
     array([[[42, 42, 42],
             [42, 42, 42]],

            [[ 7,  8,  9],
             [10, 11, 12]]])
```

```
In [84]: arr3d[1, 1,2]
```

```
     12
```

```
In [85]: x = arr3d[0]
In [87]: x[1]
```

```
     array([42, 42, 42])
```

```
arr[1:6]
```

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

```
In [91]: arr2d[:2]
```

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

```
In [92]: arr2d[:2, 2:]
```

```
     array([[3],
            [6]])
```

```
In [93]: arr2d[2, :2]
```

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

```
In [94]: arr2d[:2, 1]
```

```
     array([2, 5])
```

```
In [95]: arr2d[:, :2]
```

```
     array([[1, 2],
            [4, 5],
            [7, 8]])
```

```
In [96]: arr2d = np.array([[1, 2, 3], [4, 5, 6], [7, 8, 9]])
arr2d[1:, 1:] = 0
In [97]: arr2d
```

```
     array([[1, 2, 3],
            [4, 0, 0],
            [7, 0, 0]])
```

Boolean Indexing

```
In [98]: names = np.array(['Bob', 'Joe', 'Will', 'Bob', 'Will', 'Joe', 'Joe'])
In [99]: data = np.random.randn(7, 4)
In [100]: names
```

```
     array(['Bob', 'Joe', 'Will', 'Bob', 'Will', 'Joe', 'Joe'], dtype='<U4')
```

```
data
```

```
     array([[-0.29697953,  0.91915936, -0.69382982, -1.47995646],
            [ 1.39565922, -0.23735061, -0.89817553, -1.52265818],
            [-0.39322282, -0.01415557, -2.2120201 ,  1.60212617],
            [-0.24006546, -2.1321829 ,  0.71048549,  0.37201596],
            [ 0.37264683,  1.28094949, -2.53311724,  0.94034006],
            [-1.10056534, -0.48185341, -0.11336389,  0.36541902],
            [-1.26546128,  0.39897407, -0.6963294 , -0.10405951]])
```

```
In [102]: names == 'Bob'
```

```
     array([ True, False, False,  True, False, False, False])
```

```
In [103]: data[names == 'Bob']

     array([[-0.29697953,  0.91915936, -0.69382982, -1.47995646],
            [-0.24006546, -2.1321829 ,  0.71048549,  0.37201596]])


In [104]: data[names == 'Bob', 2:]

     array([[-0.69382982, -1.47995646],
            [ 0.71048549,  0.37201596]])


In [105]: data[names == 'Bob', 3]

     array([-1.47995646,  0.37201596])


In [106]: names != 'Bob'

     array([False,  True,  True, False,  True,  True,  True])


In [107]: data[~(names == 'Bob')]

     array([[ 1.39565922, -0.23735061, -0.89817553, -1.52265818],
            [-0.39322282, -0.01415557, -2.2120201 ,  1.60212617],
            [ 0.37264683,  1.28094949, -2.53311724,  0.94034006],
            [-1.10056534, -0.48185341, -0.11336389,  0.36541902],
            [-1.26546128,  0.39897407, -0.6963294 , -0.10405951]])


In [108]: cond = names == 'Bob'
In [109]: data[~cond]

     array([[ 1.39565922, -0.23735061, -0.89817553, -1.52265818],
            [-0.39322282, -0.01415557, -2.2120201 ,  1.60212617],
            [ 0.37264683,  1.28094949, -2.53311724,  0.94034006],
            [-1.10056534, -0.48185341, -0.11336389,  0.36541902],
            [-1.26546128,  0.39897407, -0.6963294 , -0.10405951]])


In [110]: mask = (names == 'Bob') | (names == 'Will')
In [111]: mask

     array([ True, False,  True,  True,  True, False, False])


In [112]: data[mask]

     array([[-0.29697953,  0.91915936, -0.69382982, -1.47995646],
            [-0.39322282, -0.01415557, -2.2120201 ,  1.60212617],
            [-0.24006546, -2.1321829 ,  0.71048549,  0.37201596],
            [ 0.37264683,  1.28094949, -2.53311724,  0.94034006]])


In [113]: data[data < 0] = 0
In [113]: data[data < 0] = 0
In [114]: data

     array([[0.        , 0.91915936, 0.        , 0.        ],
            [1.39565922, 0.        , 0.        , 0.        ],
            [0.        , 0.        , 0.        , 1.60212617],
            [0.        , 0.        , 0.71048549, 0.37201596],
            [0.37264683, 1.28094949, 0.        , 0.94034006],
            [0.        , 0.        , 0.        , 0.36541902],
            [0.        , 0.39897407, 0.        , 0.        ]])
```

```
In [115]: data[names != 'Joe'] = 7
In [116]: data

     array([[7.        , 7.        , 7.        , 7.        ],
            [1.39565922, 0.        , 0.        , 0.        ],
            [7.        , 7.        , 7.        , 7.        ],
            [7.        , 7.        , 7.        , 7.        ],
            [7.        , 7.        , 7.        , 7.        ],
            [0.        , 0.        , 0.        , 0.36541902],
            [0.        , 0.39897407, 0.        , 0.        ]])
```

Fancy Indexing

```
In [117]: arr = np.empty((8, 4))
In [118]: for i in range(8):
        arr[i] = i
In [119]: arr

     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 [120]: arr[[4, 3, 0, 6]]

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

```
In [120]: arr[[4, 2, 1, 5]]

     array([[4., 4., 4., 4.],
            [2., 2., 2., 2.],
            [1., 1., 1., 1.],
            [5., 5., 5., 5.]])
```

```
In [121]: arr[[-3, -5, -7]]

     array([[5., 5., 5., 5.],
            [3., 3., 3., 3.],
            [1., 1., 1., 1.]])
```

```
In [122]: arr = np.arange(32).reshape((8, 4))
In [123]: arr

     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 [124]: arr[[1, 5, 7, 2], [0, 3, 1, 2]]

     array([ 4, 23, 29, 10])
```

```
arr[3:, [0,3,1,2]]

     array([[12, 15, 13, 14],
            [16, 19, 17, 18],
            [20, 23, 21, 22],
            [24, 27, 25, 26],
            [28, 31, 29, 30]])
```

```
In [125]: arr[[1, 5, 7, 2]][:, [0, 3, 1, 2]]

     array([[ 4,  7,  5,  6],
            [20, 23, 21, 22],
            [28, 31, 29, 30],
            [ 8, 11,  9, 10]])
```

Transposing Arrays and Swapping Axes

```
In [126]: arr = np.arange(15).reshape((3, 5))
In [127]: arr

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

```
arr.T
```

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

```
In [132]: arr = np.arange(16).reshape((2, 2, 4))
In [133]: arr
```

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

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

```
In [134]: arr.transpose((1, 0, 2))
```

```
array([[[ 0,  1,  2,  3],
        [ 8,  9, 10, 11]],

       [[ 4,  5,  6,  7],
        [12, 13, 14, 15]]])
```

```
In [134]: arr.transpose((0, 2, 1))
```

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

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

```
data=np.arange(24).reshape((2,3,4))
data
```

```
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]]])
```

```
data.transpose((1, 0, 2))
```

```
array([[[ 0,  1,  2,  3],
        [12, 13, 14, 15]],

       [[ 4,  5,  6,  7],
             [16, 17, 18, 19]],

            [[ 8,  9, 10, 11],
             [20, 21, 22, 23]]])
```

```
arr.transpose((2, 1, 0))
```

```
     array([[[ 0,  8],
             [ 4, 12]],

            [[ 1,  9],
             [ 5, 13]],

            [[ 2, 10],
             [ 6, 14]],

            [[ 3, 11],
             [ 7, 15]]])
```

```
arr.swapaxes(1, 2)
```

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

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

```
data.swapaxes(1, 2)
```

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

            [[12, 16, 20],
             [13, 17, 21],
             [14, 18, 22],
             [15, 19, 23]]])
```

Universal Functions: Fast Element-Wise Array Functions

```
In [137]: arr = np.arange(10)
In [138]: arr
```

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

```
In [139]: np.sqrt(arr)
```

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

```
In [140]: np.exp(arr)
```

```
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 [141]: x = np.random.randn(8)
In [142]: y = np.random.randn(8)
x
```

```
array([-0.12030836, -0.94359539, -0.08632051, -0.61943396,  0.13086726,
        0.86218457, -1.58912993, -0.30011585])
```

```
y
```

```
array([-2.00586907,  0.95343254, -0.31492545,  0.16124615, -0.14670123,
       -0.99284744, -0.42418282,  0.74156191])
```

```
In [145]: np.maximum(x, y)
```

```
array([-0.12030836,  0.95343254, -0.08632051,  0.16124615,  0.13086726,
        0.86218457, -0.42418282,  0.74156191])
```

Sorting

```
In [195]: arr = np.random.randn(6)
In [196]: arr
```

```
array([1.15353516, 0.1698373 , 0.71595498, 1.45208226, 0.72602978,
       1.54844227])
```

```
In [197]: arr.sort()
In [198]: arr
```

```
array([0.1698373 , 0.71595498, 0.72602978, 1.15353516, 1.45208226,
       1.54844227])
```

```
In [199]: arr = np.random.randn(5, 3)
In [200]: arr
```

```
array([[ 0.47929092, -0.63831671,  0.3715362 ],
           [ 0.05946153,  0.57518439,  0.57415745],
           [-0.08339772,  1.32975319, -1.33700294],
           [-1.67367899,  1.99790936, -1.63852941],
           [-0.37817687, -0.05338839, -1.0369437 ]])
```

```
In [201]: arr.sort(1)
In [202]: arr
```

```
    array([[-1.67367899, -1.63852941, -0.63831671],
           [-1.33700294, -0.37817687, -0.05338839],
           [-1.0369437 , -0.08339772,  0.57518439],
           [ 0.05946153,  0.3715362 ,  1.32975319],
           [ 0.47929092,  0.57415745,  1.99790936]])
```

Unique and Other Set Logic

```
In [206]: names = np.array(['Bob', 'Joe', 'Will', 'Bob', 'Will', 'Joe', 'Joe'])
In [207]: np.unique(names)
```

```
    array(['Bob', 'Joe', 'Will'], dtype='<U4')
```

```
In [208]: ints = np.array([3, 3, 3, 2, 2, 1, 1, 4, 4])
In [209]: np.unique(ints)
```

```
    array([1, 2, 3, 4])
```

```
In [210]: sorted(set(names))
```

```
    ['Bob', 'Joe', 'Will']
```

```
In [211]: values = np.array([6, 0, 0, 3, 2, 5, 6])
In [212]: np.in1d(values, [2, 3, 7])
```

```
    array([False, False, False,  True,  True, False, False])
```

File Input and Output with Arrays

```
In [213]: arr = np.arange(10)
In [214]: np.save('some_array', arr)
```

```
In [215]: np.load('some_array.npy')
```

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

```
In [216]: np.savez('array_archive.npz', a=arr, b=arr)
In [217]: arch = np.load('array_archive.npz')
In [218]: arch['b']

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

```
x = np.arange(10)
y = np.arange(11, 20)
np.savez('temp_arra.npz', a=x, b=y)
```

```
arch1=np.load('temp_arra.npz')
arch1['a']

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

Linear Algebra

```
In [223]: x = np.array([[1., 2., 3.], [4., 5., 6.]])
In [224]: y = np.array([[6., 23.], [-1, 7], [8, 9]])
```

```
x.dot(y)

     array([[ 28.,  64.],
            [ 67., 181.]])
```

```
np.dot(x, y)

     array([[ 28.,  64.],
            [ 67., 181.]])
```

```
np.dot(x, np.ones(3))

     array([ 6., 15.])
```

```
x @ np.ones(3)

     array([ 6., 15.])
```

```
In [232]: X = np.random.randn(5, 5)
In [233]: mat = X.T.dot(X)
In [234]: inv(mat)
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-35-3102f011b32c> in <module>()
      1 X = np.random.randn(5, 5)
      2 mat = X.T.dot(X)
----> 3 inv(mat)
```

```
import random

random.seed(3)

print(random.randint(1, 1000))

random.seed(3)
print(random.randint(1, 1000))

print(random.randint(1, 1000))
```

```
244
244
607
```

Example: Random Walks

```
import random
import matplotlib.pyplot as plt
position = 0
walk = [position]
steps = 1000
for i in range(steps):
  step = 1 if random.randint(0, 1) else -1
  position += step
  walk.append(position)
plt.plot(walk[:100])
```