Numpy

> NumPy is the fundamental package for scientific computing in Python.

In [2]:

```
import numpy as np
```

In [5]:

```
np.__version__
```

Out[5]:

```
'1.13.1'
```

Array

numpy.array()

In [5]:

```
a = np.array([[1, 2],[3, 4]])
a
```

Out[5]:

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

Matrix

In [12]:

```
b = np.matrix([[1, 2],[3, 4]])
b
```

Out[12]:

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

Dtype

int8 = Byte (-128 to 127)
int16 = Integer (-32768 to 32767)
int32 = Integer (-2147483648 to +2147483647)
int64 = Integer (-9223372036854775808 to +9223372036854775807)
Boolean = (True or False) stored as a byte
float = Shorthand for float64 : Double precision float: sign bit, 11 bits exponent, 52 bits mantissa
complex = Shorthand for complex128 : Complex number

In [7]:

```
np.array([1, 2, 3], dtype='int8')
```

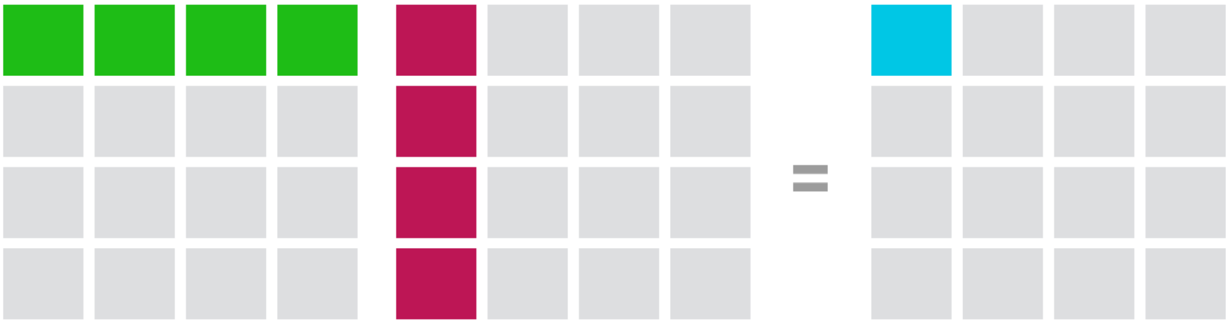
Out[7]:

```
array([1, 2, 3], dtype=int8)
```

عملگر های محاسباتی

Multiplication

@ or `numpy.dot()`



In [8]:

```
a @ a
```

Out[8]:

```
array([[ 7, 10],
       [15, 22]])
```

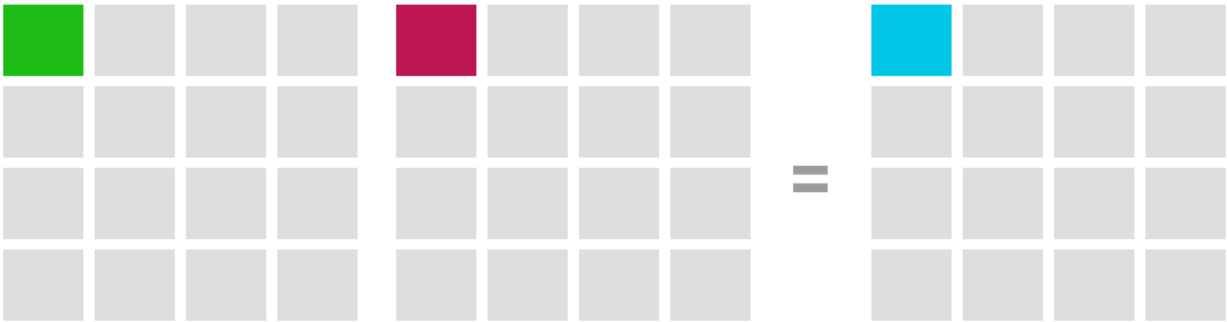
In [9]:

```
np.dot(a,a)
```

Out[9]:

```
array([[ 7, 10],
       [15, 22]])
```

`numpy.multiply()`



In [10]:

```
np.multiply(a,a)
```

Out[10]:

```
array([[ 1,  4],
       [ 9, 16]])
```

*

Matrix

In [14]:

```
b*b
```

Out[14]:

```
matrix([[ 7, 10],
        [15, 22]])
```

Array

In [17]:

```
a*a
```

Out[17]:

```
array([[ 1,  4],
       [ 9, 16]])
```

numpy.prod()

In [18]:

```
np.prod(a)
```

Out[18]:

```
24
```

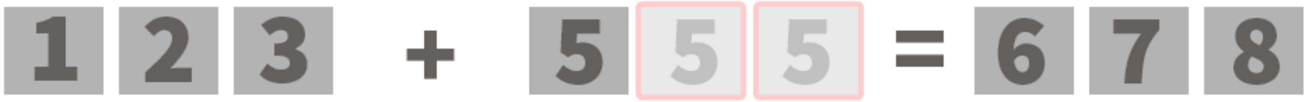
Broadcasting

In [21]:

```
my_array = np.array([1, 2, 3])
my_array + 5
```

Out[21]:

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

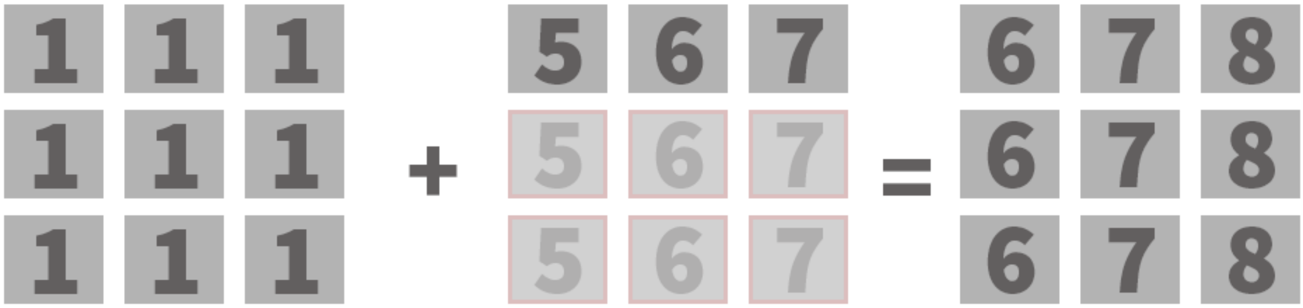


In [20]:

```
a = np.ones((3,3))
b = np.array([5, 6, 7])
a + b
```

Out[20]:

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

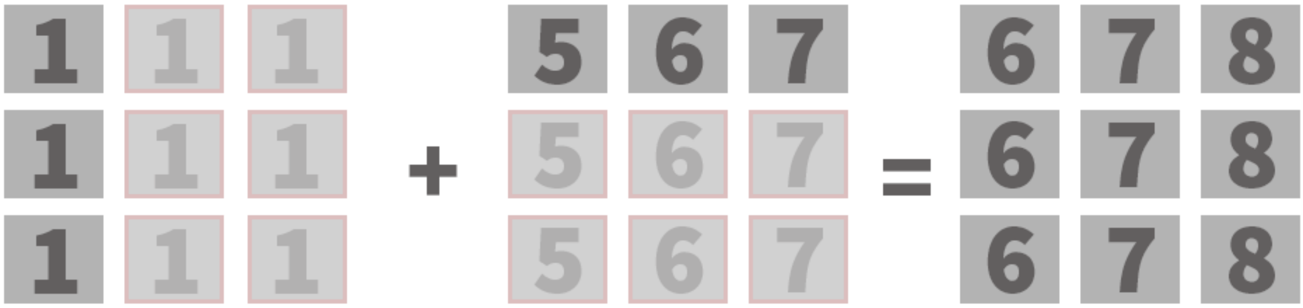


In [25]:

```
a = np.ones((3,1))
b = np.array([5, 6, 7])
a + b
```

Out[25]:

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



Summation

In [46]:

```
np.sum(a)
```

Out[46]:

```
10
```

In [47]:

```
np.cumsum(a, axis=0)
```

Out[47]:

```
array([[1, 2],
       [4, 6]], dtype=int32)
```

Subtraction

In [48]:

```
np.subtract(a, a)
```

Out[48]:

```
array([[0, 0],
       [0, 0]])
```

division

In [49]:

```
np.divide([5, 6, 7],3)
```

Out[49]:

```
array([ 1.66666667,  2.        ,  2.33333333])
```

In [50]:

```
np.floor_divide([5,6,7],3)
```

Out[50]:

```
array([1, 2, 2], dtype=int32)
```

```
Numpy.math()
```

In [96]:

```
np.math.sqrt(5)
```

Out[96]:

```
2.23606797749979
```

In [53]:

```
np.math.nan
```

Out[53]:

```
nan
```

In [54]:

```
np.math.inf
```

Out[54]:

```
inf
```

Generate random numbers

توزیع یکنواخت پیوسته

In [222]:

```
np.random.uniform(1,5,(2,3))
```

Out[222]:

```
array([[ 3.82471777,  3.32763493,  1.56875982],
       [ 3.82676751,  4.28993081,  3.31292906]])
```

توزیع نرمال

In [123]:

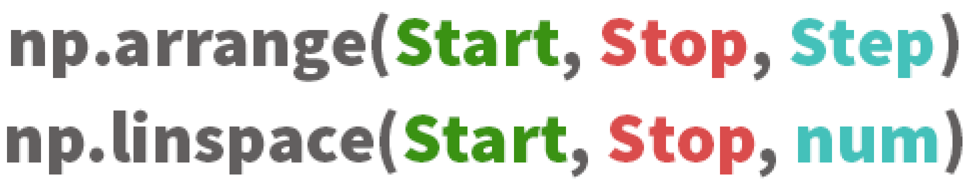
```
np.random.standard_normal((5,))
```

Out[123]:

```
array([-0.78931388,  0.16882026, -0.68168125,  1.47155377, -0.62524079])
```

https://docs.scipy.org/doc/numpy/reference/routines.random.html (https://docs.scipy.org/doc/numpy/reference/routines.random.html)

Sequences



np.arrange(Start, Stop, Step)

np.linspace(Start, Stop, num)

In [147]:

```
np.arange(1,10,3,dtype='float')
```

Out[147]:

```
array([ 1.,  4.,  7.])
```

In [164]:

```
np.linspace(1, 10, 4)
```

Out[164]:

```
array([  1.,   4.,   7.,  10.])
```

Mask

In [186]:

```
my_mask = a>2
a[my_mask]
```

Out[186]:

```
array([4, 5])
```

In [190]:

```
my_mask2 = np.logical_and(a>1,a<4)
a[my_mask2]
```

Out[190]:

```
array([2])
```

Array creation

In [93]:

```
my_zeros = np.zeros((3,2))
my_zeros
```

Out[93]:

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

In [94]:

```
my_ones = np.ones((3,2))
my_ones
```

Out[94]:

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

In [95]:

```
np.size(my_ones)
```

Out[95]:

```
6
```

In [96]:

```
np.shape(my_ones)
```

Out[96]:

```
(3, 2)
```

Sort

In [97]:

```
a = np.array([[1,4],[3,2]])
print(a)
np.sort(a, axis=0)
```

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

Out[97]:

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

In [98]:

```
np.argsort(a)
```

Out[98]:

```
array([[0, 1],
       [1, 0]], dtype=int64)
```

Data Analysis

In [99]:

```
a = np.array([1, 2, 3, 4, 1, 3]) # a set
a
```

Out[99]:

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