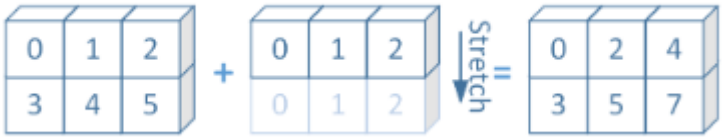
Numpy高级

numpy广播机制

不同形状和维度的数组在某些情况下, numpy可以执行加减乘除的运算,这种机制就叫做广播机制

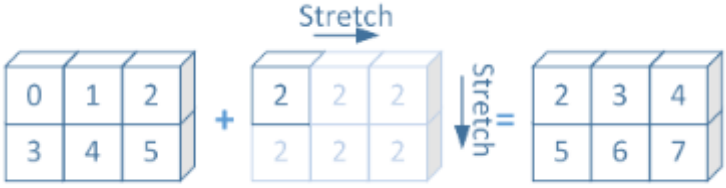
```
import numpy as np
a = np.arange(3)
b = 1
a + b
```


```
a = np.arange(6).reshape(2, 3)
b = np.array([0, 1, 2])
a + b
```

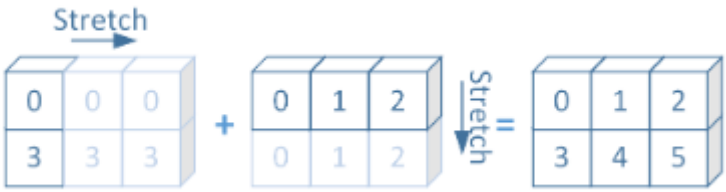



```
a = np.arange(6).reshape(2, 3)
b = np.array([0, 1]).reshape(2, 1)
a + b
```

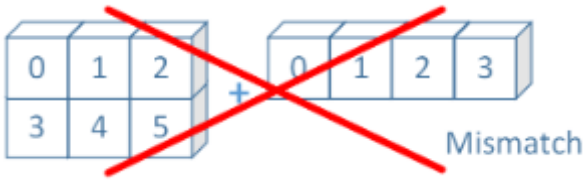

```
a = np.arange(6).reshape(2, 3)
b = 2
a + b
```



```
a = np.array([0, 3]).reshape(2, 1)
b = np.array([0, 1, 2])
a + b
```




```
a = np.arange(6).reshape(2, 3)
b = np.array([0, 1, 2, 3])
a + b
# 维度mismatch
```



我们可以通过下面这些方法对数组中元素进行搜索和计数。列举如下：

- `argmax(a ,axis,out)`：返回数组中指定轴的最大值的索引。
- `nanargmax(a ,axis)`：返回数组中指定轴的最大值的索引,忽略 NaN。
- `argmin(a ,axis,out)`：返回数组中指定轴的最小值的索引。
- `nanargmin(a ,axis)`：返回数组中指定轴的最小值的索引,忽略 NaN。
- `nonzero(a)`：返回数组中非 0 元素的索引。
- `where(条件,x,y)`：根据指定条件,从指定行、列返回元素。
- `count_nonzero(a)`：计算数组中非 0 元素的数量。

numpy矩阵的乘法

买菜算账问题

- 小明今天要做饭，消耗两斤肉，肉每斤20元，一共要花费多少？ $20 \times 2 = 40$
- 小明今天要做饭，消耗2斤肉，1斤蔬菜。肉每斤20元，蔬菜每斤5元，则一共需多少花费？

$$\begin{bmatrix} 20 & 5 \end{bmatrix} \times \begin{bmatrix} 2 \\ 1 \end{bmatrix} = 45$$

- 小明今天要做饭，消耗2斤肉，1斤蔬菜。在“钱大妈“ 肉每斤20元，蔬菜每斤5元，
  在沃尔玛肉每斤25元，蔬菜每斤10元，则一共需多少花费？

$$\begin{bmatrix} 20 & 5 \\ 25 & 10 \end{bmatrix} \times \begin{bmatrix} 2 \\ 1 \end{bmatrix} = \begin{bmatrix} 45 \\ 60 \end{bmatrix}$$

```
a = np.array([[20,5],[25,10]])
b = np.array([[2],[1]])
np.dot(a,b)
# a.dot(b)
# a @ b
```

- 代数由数字和位置表述
- 数字的位置有特殊的意义
- 在左边的这个矩阵的每一行，都代表了一种价目表；在右边的矩阵的每一列，都代表了一种做饭方式。那么所有可能的组合所最终产生的花费，则在结果矩阵中表示出来了。
- 矩阵运算就是定义了一组数据排放的格式， 不同位置的数据代表不同的含义。

numpy求解鸡兔同笼问题

鸡兔同笼是中国古代的数学名题之一。

大约在1500年前，《孙子算经》中就记载了这个有趣的问题。书中是这样叙述的：
今有鸡兔同笼，上有三十五头，下有九十四足，问鸡兔各几何？

翻译： 有若干只鸡兔同在一个笼子里，从上面数，有35个头，从下面数，有94只脚。问笼中各有多少只鸡和兔？

$$\text{Solve } \mathbf{ax} = \mathbf{b} \Rightarrow \begin{bmatrix} 1 & 1 \\ 2 & 4 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 35 \\ 94 \end{bmatrix} \text{ for } \mathbf{x}$$

```
a = np.array([[1,1],[2,4]])
b = np.array([35,94])

np.linalg.slove(a,b)

array([23., 12.])
```

numpy倒数和逆矩阵

倒数（reciprocal / multiplicative inverse）是一个数学学科术语，拼音是dào shù。是指数学上设一个数x与其相乘的积为1的数，记为1/x，过程为“乘法逆”，除了0以外的数都存在倒数， 分子和分母相倒并且两个乘积是1的数互为倒数，0没有倒数。

$$3 \times X = 27$$

3的倒数是1/3 等式两边同时乘以 1/3

$$\frac{1}{3} \times 3 \times X = \frac{1}{3} \times 27$$

化简后：

$$X = 9$$

逆矩阵跟倒数是一样的。

设 **A** 是数域上的一个n阶矩阵，若在相同数域上存在另一个n阶矩阵 **B** ，使得： **AB=BA=E** ，则我们称 **B** 是 **A** 的逆矩阵，而A则被称为可逆矩阵。注：E为单位矩阵。

$$AX = B$$

已知A的逆矩阵乘以A矩阵等于E，单位矩阵

$$A^{-1}A = E$$

则把原始等式同时左乘 A^{-1} 有：

$$A^{-1}AX = A^{-1}B.$$

$$X = A^{-1}B$$

```
#求解鸡兔同笼，等同于A的逆矩阵乘以b矩阵
np.dot(np.linalg.inv(a),b)
```

作业题

基本Numpy操作

导入numpy包

创建10个0的一维数组

创建10个1的一维数组

创建10个5的一维数组

创建10-50的一维数组

创建10-50偶数的一维数组

创建3x3二维数组 里面内容0-8

创建3x3二维单位矩阵

创建0~1之间的随机数

创建25个正态分布的随机数

创建10x10的矩阵从0.01 到1

创建0到1,中间20个相同的线段

创建一个 5x5 的二维数组，其中边界值为1，其余值为0

找出两个一维数组里面相同的元素

创建一个长度为 5 的一维数组，并将其中最大值替换成 0

数组操作及计算

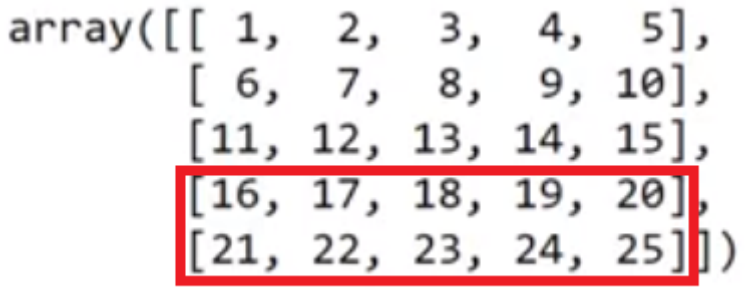
切割数组

```
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]])
```

```
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]])
```

```
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]])
```

```
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]])
```



求mat所有元素的和

求mat所有元素的标准差

按行求mat所有元素的和