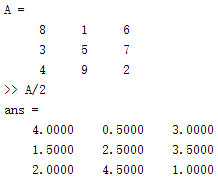
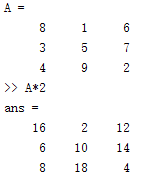
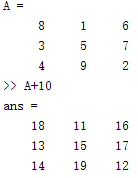
关于matrices的运算

# 矩阵与数值的运算

**加减乘除：**

一个矩阵与一个数组进行加减乘除操作时，都是矩阵中的每一个元素分别与这个数值进行加减乘除。



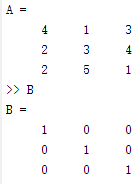
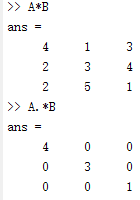
# 矩阵之间的运算

**两个矩阵相互加减，必须是同维度。**

**矩阵之间的乘法**：

## A\*B ： A的行数必须等于B的列数。

## 点乘： A.\* B维度必须相同，对应位置上的元素相乘。

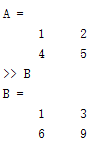
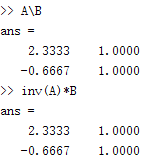
 

矩阵之间的除法：**除法分为左除和右除。简单的记忆方法：棍子|，向左倒是左除\，向右倒是右除/**。

**AB = C 则 A = CＢ－１ Ｂ＝　Ａ－１Ｃ**

**A/B** ： A右除B，等于A 乘以Ｂ的逆，即AB-1

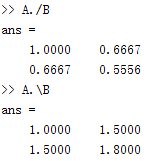
A\B : A 左除Ｂ，等于Ａ的逆乘以Ｂ，即A-1B。

**点除： 点右除和点左除**。

A./B : A 点右除B，即A中的每个元素除以B中的每个元素。

A.\B : A点左除Ｂ，即B中的每个元素除以Ａ中的每个元素。



# 矩阵的乘方

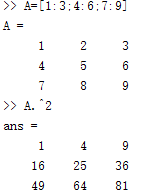
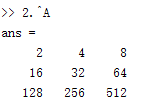
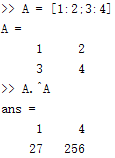
对于矩阵作为指数，有特殊的运算，在矩阵论中学习过。

如 **2^A 与2.^A 不一样**。

**矩阵的乘方用符号 .^ 实现，这是点乘方。**

点乘方有三种方式：

**以矩阵为底、底为数值、底和指数都是矩阵**。

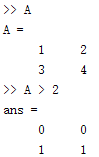
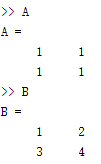
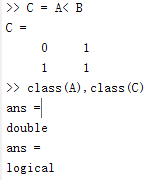
# 矩阵的关系运算

**关系运算符6种： < 、> 、== 、~= 、>= 、<=**

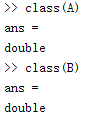
矩阵与数值关系比较： 矩阵中的每个元素分别与该数值比较大小。

矩阵与矩阵之间的比较： 维度必须相同，对应位置元素比较。

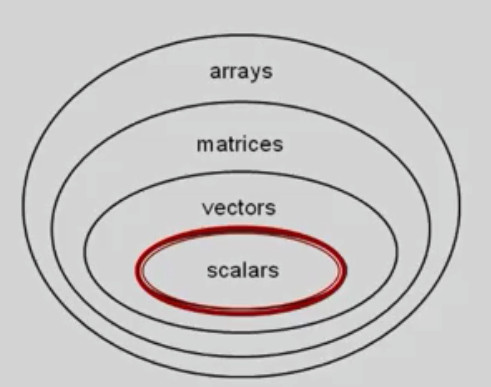
如果为真，则用1表示，为假用0表示。

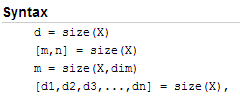
# class 可以判断数据类型。

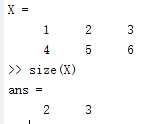
# Arrays与Matrices

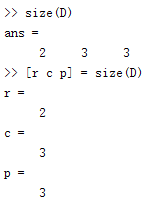
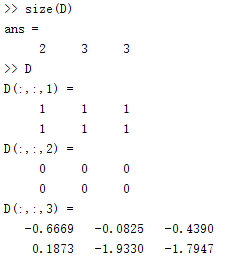


# size： 获取矩阵的维度。



m = size(X，dim ) 指定获取某一维度的大小。



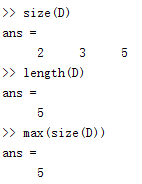


# length：获取矩阵的列数。

L = length(X) returns **the length of the largest array dimension in X.** For vectors, the length is simply the number of elements. For arrays with more dimensions, the length is max(size(X)). The length of an empty array is zero.

Length（X）获取矩阵X的各个维度长度的最大值。

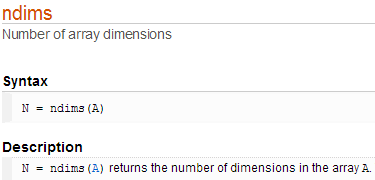
相当于 **max(size(X))**



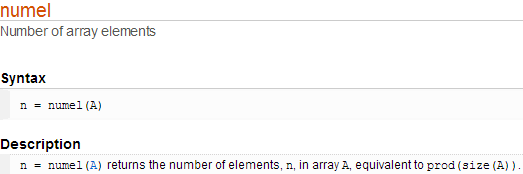
# ndims 与 numel

ndims 用于获取矩阵的维数，大于等于2；认为向量也是两个维度。

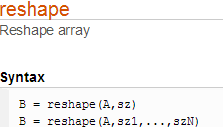
The number of dimensions is always greater than or equal to 2. The function ignores trailing singleton dimensions, for which size(A,dim) = 1.



numel 用于获取矩阵的元素的个数

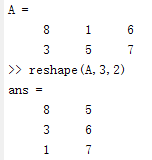


# reshape 函数



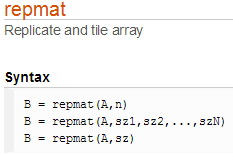
将一个矩阵变形为其他维数的矩阵，元素个数必须相同，否则报错。





# repmat 函数

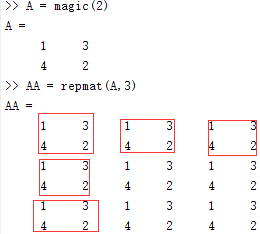
利用复制扩展矩阵。

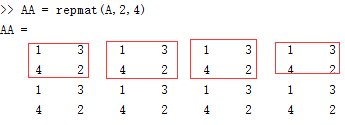


**repmat(A,n)** 行和列同时复制扩展n份

**repmat（A,m，n）** 行复制扩展m份，列复制扩展n份。

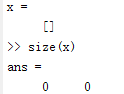
**sz 表示[sz1 sz2 sz3 ....]**

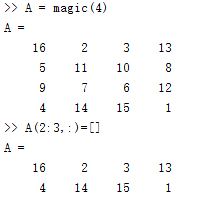




# 空矩阵

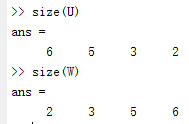
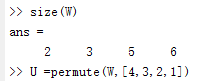
也可以用于**删除矩阵中的某一行或列或某些行或列**。





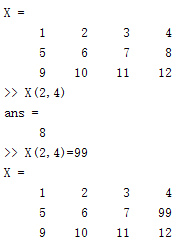
# permute ipermute 转置

多维数组的转置与取消转置

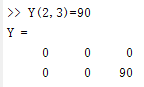
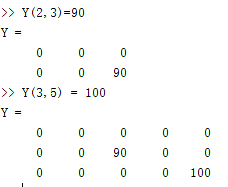


# 获取矩阵中某一个或某一部分元素

## 获取或修改某一个元素：

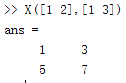


如果某个变量不存在，直接赋予其中某一个元素值，则默认其他值为零。扩展矩阵的大小，扩展的其他元素为零。。

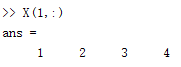
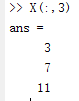
 

## 获取部分值

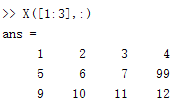
1. 获取四个元素：



B． 获取某一行或某一列：

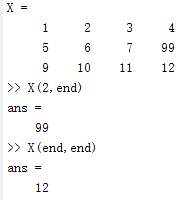
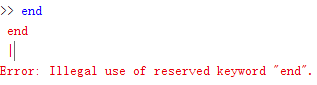
 

C． 获取某些行或列：

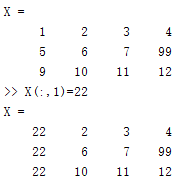


D. 利用end表示最后一个

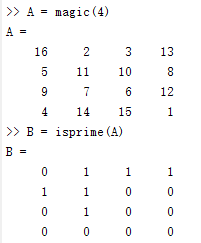
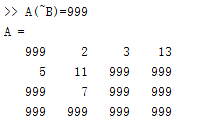
如 X(end,end) end只可以当索引使用，不可以直接获取该值。



# A（~B） ~B表示对B中每个元素判断逻辑，如果true则选中，否则不选中。

# 多维数组

针对一维数组和二维数组，使用的比较多，但是对于多维数组中，其中三维数组在图像处理中有着非常广泛的应用。

MATLAB对多维数组的操作与对一维数组和二维数组的操作基本相同。

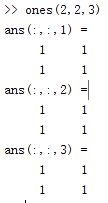
这里主要是介绍三维数组。

三维数组的第三维称为“页”，即一个三维数组由行、列和页组成，其中一页包含一个由行和列构成的二维数组，并且每一页的二维数组必须有相同的维数。

## 三维数组的创建

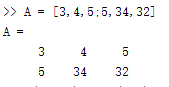
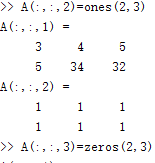
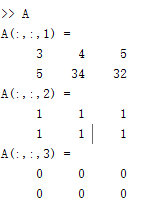
* 1. 利用矩阵生成函数生成

利用ones 、zeros 、rand、randn等。

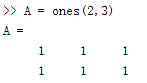


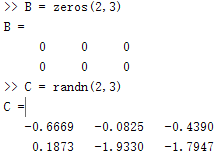
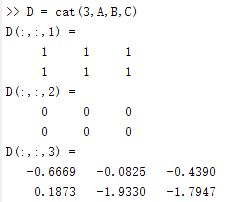
* 1. 利用索引方式

先生成一个二维的，然后利用索引扩展。

* 1. 利用cat函数



## 多维数组的其他运算

利用函数完成，如 reshape 、 size 、 squeeze 、 sub2ind、shiftdim等。