eye ماتریس همانی

این دستور یک ماتریس همانی با درجه وارد شده میسازد .

eye(n)

n درجه ماتریسی است که می خواهیم بسازیم .

```
>>eye(3)

Ans=

1 0 0

0 1 0

0 0 1
```

اگر در این دستور دو پارامتر نعریف شود (سطر و ستون) ماتریسی با این درجه ساخته خواهد شد که همه درایه های آن صفر و درایه های موجود در قطر اصلی(و یا n*n) یک می با شد .

```
>>eye(3,4)

Ans=

1  0  0  0

0  1  0  0

0  0  1  0
```

ones ماتریس یک

ماتریسی با درایه های یک می سازد .

در صورتی که پارامتر تعریف شده یک عدد باشد ، ماتریس مربع با درجه همان عدد ساخته خواهد شد.

ولی در صورتی که دو عدد وارد شود ماتریس بر اساس درجه وارد شده تشکیل می شود .

```
>>ones(3)
```

```
Ans=

1 1 1

1 1 1

1 1 1
```

```
>>ones(3,2)
```

```
Ans=

1 1

1 1

1 1
```

zeros ماتریس صفر

ماتریسی با درایه های صفر می سازد .

عملکرد این دستور دقیقا مانند **ones** می با شد .

```
>>zeros(2)
```

```
ans=

0  0

0  0
```

max بزرگترین درایه در ماتریس

این دستور بزرگترین درایه در یک ماتریس را در بعد خاصی (سطر یا ستون و...)پیدا می کند .

این دستور بدینگونه نوشته می شود :

max(a,DIM)

این دستور ماکزیمم ماتریس **a** را در طول بعد **DIM** پیدا می کند .

اگر **DIM** وارد نشودو یا بیشتر از بعد ماتریس باشد ...سیستم عدد یک را در نظر میگیرد که همان ماکزیمم در ستون می باشد .

```
>>s=[1 2 3;4 5 6];
>>max(s,2)
```

```
Ans=
2  2  3
4  5  6
```

min کوچکترین درایه ماتریس

کوچکترین درایه ماتریس را می دهد .

نحوه استفاده از دستور دقیقا مانند **max** است .

```
>>s=[1 2 3;4 5 6];
>>min(s)
Ans=
1  2  3
```

در این دستور DIM وارد نشده و سیستم در ستون مینیمم را پیدا میکند .

sort مرتب کردن درایه ها

این تابع درایه ها ی مو جود بر روی سطر یا ستون و... را مرتب می کند .

نحوه بکارگیری این دستور مانند دستور های بالا میباشد .

```
>>k=[3 1 2 ;7 3 3;8 2 6];
>>sort(k,2)
Ans=
1  2  3
3  3  7
2  6  8
```

دومین ابزاری که در این دستور گذارده شده است ، مکان در ماتریس اصلی است ،

یعنی پس از مرتب نمودن مکان درایه ها را در ماتریس اصلی نشان میدهد .

به این مثال توجه کنید :

```
>>k=[3 1 2 ;7 3 3;8 2 6]

>>[m,n]=sort(k,2)

k=

3  1  2

7  3  3

8  2  6


m=

1  2  3

3  3  7

2  6  8


n=

2  3  1

2  3  1

2  3  1
```

در دستور بالا : منظور از m ماتریس مرتب شده است .

منظور از n ماتریس مکان است . بدینگونه که درایه متناظر در این ماتریس با ماتریس مرتب شده شماره درایه را در ماتریس اصلی را نشان می دهد که جا به جا شده است .

k ماتریس اصلی.....m ماتریس مرتب شده .....n ماتریس مکان

sum مجموع درایه ها

این دستور مجموع سطر ها یا ستونها و... را در یک ماتریس محاسبه می کند .

نحوه استفاده از این دستور بدینگونه است :

```
sum(a,DIM)
```

در صورتی که DIM وارد نشود ویا اشتباه باشد عدد 1 در نظر گرفته می شود .

```
>>s=[1 2 3;4 5 6];
>>sum(s,2)
Ans=
6
15
```

دستور بالا مجموع درایه های واقع در سطر را باهم جمع می کند .

prod حاصلضرب درایه ها

این دستور درایه ها را در هم ضرب می کند .

```
prod(a,DIM)
```

کاربرد این دستور دقیقا مانند sum است

```
>>k=[1 2 3;4 5 6];
>>prod(k,2)
Ans=
6
120
```

mean میانگین درایه ها

میانگین درایه ها را محاسبه میکند .

mean(a,DIM)

کاربرد دقیقا مانند sum است .

```
>>mean(k,2)

Ans=

2

5
```

diag قطر اصلی

این دستور قطر اصلی ماتریس مربع را به صورت یک ماتریس ستونی می دهد والبته ماتریس مربع متناظر با ماتریس ستونی ویا سطری معرفی شده را نیز می دهد .

diag(a)

a می تواند یک ماتریس مربع ویا ستونی باشد .

```
>>a=[1 2 3;4 5 6;7 8 9];

>>diag(a)

Ans=

1

5

9
```

```
>>b=[1 2 3];

>>diag(b)
```

```
Ans=

1  0  0

0  2  0

0  0  3
```

در صورتی که ماتریس وارد شده غیر سطری وغیر ستونی ومربع نباشد(m*n) .

پاسخ دستور درایه های واقع در مکان های (n*n) و یا(m*m) خواهد بود .

det دتر مینان ماتریس

درترمینان ماتریس معرفی شده را ارائه می کند .

ماتریس معرفی شده باید مربع باشد .

```
>>a=[1 2 3;4 5 6];

>>det(a)
```

```
Error using ==> det???

Matrix must be square .
```

```
>>a=[4 2 3;4 6 6;7 8 10];
>>det(a)
Ans=
22
```

trace تریس ماتریس

مجموع درایه های واقع در قطر اصلی را محاسبه می کند .

```
>>a=[4 2 3;4 6 6;7 8 10];
>>trace(a)
Ans=
20
```

در صورتی که ماتریس مربع نباشد درایه های (n*n) با هم جمع خواهد شد .

```
>>a=[1 2 3;4 5 6];
>>trace(a)
Ans=
6
```

rank مرتبه ماتریس

مرتبه ماتریس یعنی بزرگترین درجه ای که ماتریس مستقل باشد(سطر یا ستون مضرب ویا مجموع با دیگری نباشد) .

```
>>a=[1 2 3;2 4 6;6 7 8];
>>rank(a)
```

```
Ans=
2
```

```
>>a=[1 2 3;2 5 6;6 7 8];
>>rank(a)
```

```
Ans=
3
```

در مثال های بالا :

در اولین مثال سطر دوم دو برابر سطر اول است به همین دلیل در شمارش به حساب نیامده است .

در دومین مثال درایه دوم سطر دوم تغییر داده شده که دیگر سطر دوم و سوم وابسته نیست و در شمارش محاسبه شده است .

flipdim چرخش ماتریس در بعد

ماتریس را حول بعد تعریف شده می چرخاند .

flipdim(a,DIM)

A ماتریس مربوطه و DIM بعد تعریف شده است .

```
>>a=[1 2 3;4 5 6;7 8 9];
>>flipdim(a,2)
Ans=
3  2  1
4  5  6
9  8  7
```

fliplr چرخش چپ راست ماتریس

ماتریس را به صورت چپ و راست می چر خاند .

```
>>a=[1 2 3;4 5 6;7 8 9];
>>fliplr(a)
Ans=
3  2  1
4  5  6
7  8  9
```

میبینید که کارکرد دستور دقیقا مانند flipdim(a,2) میباشد .

flipud چرخش ماتریس به صورت بالا پایین

ماتریس را به صورت بالا پایین می چرخاند .

کاملا مانند **fliplr** است .

کارکرد آن مانند **flipdim(a,1)** میباشد .

```
>>a=[1 2 3;4 5 6;7 8 9];
>>flipud(a)
Ans=
7 8 9
4 5 6
1 2 3
```

rot90 چرخش ماتریس

ماتریس را به اندازه **90** درجه می چرخاند .

```
>>a=[1 2 3;4 5 6;7 8 9];
>>rot90(a)
Ans=
3  6  9
2  5  8
1  4  7
```

برای چرخش در جهت های مختلف میتوان از دستور زیر استفاده کرد .

rot90(a,k)

در اینجا **a** ماتریس مربوطه و **k** درجه بر حسب ضریبی از **90** میباشد که برای چرخش در جهت عکس آن را منفی می گیریم .

```
>>a=[1 2 3;4 5 6;7 8 9];

>>rot90(a,-1)
```

```
Ans=

7 4 1

8 5 2

9 6 3
```