

آموزش رم افزار

MATLAB

1st ed.

Learning pakage

commands

functions

graphics

2d 3d plot

gui programming

compiling - create exe



contained useful reference & papers

بنام خدا

این مطالب به هیچ عنوان نمی تواند مرجع مناسبی برای یک کاربر
هرفه ای باشد ولی طبیعتاً برای افرادی که تمایل به یادگیری برنامه
نویسی متلب را دارند مفید و موثر فواهد بود.

با این همه سعی بر این بوده که مطالبی ساده و در عین حال تقریباً
کامل گرد آوری شود تا مورد استفاده هر چه بیشتر علاقمندان قرار گیرد.

مطالب و مثال های ارائه شده ، همه در v.7 MATLAB اجرا شده
است و همه مثالها و دستورات در این نسخه قابل اجرا است و اگر در
نسخه بالاتر و یا پایین تر اجرای دستور با فقط مواجه شدممکن
است در نسخه مربوطه فرم بگارگیری دستورات تفاوت داشته باشد که با
بگارگیری HELP میتوانید این نوع مشکلات را رفع نمایید.

برای کاربران هرفه ای وکسانی که این مطالب برایشان کافی نیست ،
لیست مراجع مفید و آدرس آنها و همینطور توضیحات کافی در بخش
آمده است . Reference

این نسخه ، نسخه اول بوده و چند بخش مهم از مطالب ((مانند
پردازش تصویر - برنامه نویسی GUI - منطق فازی - کامپیاپل گردن
وساخت فایل اجرایی از برنامه متلب .)) را به همراه فود ندارد ولی
مطمئن باشید در آینده ای نه پندان دور نسخه کامل به دستتان فواهد
رسید .

در پایان ، از صميم قلب از هر نوع انتقاد و پيشنهاد استقبال ميكنيم .
به اميد روزی که اين مطالب به کمک فرد شما به مرجعی غنی و
قدرتمند تبدیل شود .
منتظر نامه های ارزشمند شما هستیم .

مشکی باف

nm_tafa@yahoo.com

برمکی

Abas_barmaky@yahoo.com

فهرست مطالب

7	الگوریتم و الگوریتم نویسی
9	MATLAB
	نصب متلب
10	پنجره های متلب
12	کار در command window
13	تعریف متغیر و ماتریس
14	اندیس ماتریس
17	عملگرها
	: ; + - * / \ ^ ' .^ ./ \ .^ ! < > <= >= == ~= &
23	پارامتر های اولیه
	Pi i eps nan realmin realmax
25	دستورات ابتدایی
	Input
27	Disp
28Clc
	Home
29Clear
30	Nargin
	Nargout
	Beep
31	m-file
33	Function
36	کنترل در برنامه نویسی
	If ...end
37	Else
	Elself
38	Switch.. Case
39	For ... end
40	While ... end
41	Continue
	Break

42	منطق در شرط
43	format
44	گردکردن
45	توابع عددی
46	Primes Factor Factorial Gcd Lcm
47	توابع مختلف
48.....	Abs Complex Imag Real Angle Conj
49	توابع نمایی
50.....	Sqrt Sqrtn Nthroot Power Pow2 Exp Log
51	توابع مثلثاتی
52	دستورات منطقی
53	Isempty Isnumeric Isequal Isreal Isprime
54	توابع زمانی
	Clock Date Tic ... toc Pause
55	توابع آرایه ای
	Numel Length

56	Find Size
57	ماتریس های خاص
58	Magic Rand Eye Ones Zeros
59	توابع ماتریسی
60	Max Min Sort
61.....	Sum Prod Mean
62	Diag Det
63	Trace Rank
64	Flipdim
65.....	Flipr Flipud Rot90
66	کار با فایل
67	Load Open Dlmwrite
68.....	Dlmread Textread
69	ترسیم دو بعدی
	Plot
74	تنظیمات صفحه رسم
75.....	Xlabel Ylabel Title Legend
76.....	چند ترسیم در یک صفحه
	Subplot

78 ترسیمات سه بعدی

Plot3

79 رسم سطح ولایه

Peaks
Meshgrid
77 Mesh
Contour
82 Meshc
Surf
79 Surf
Surfc
contour3
85 Plot3
86 View
87 ترسیم توابع

Ezplot
88 Ezplot3
Ezmesh
90 Ezsrf
91 نمودار های آماری

Bar
Hist
Stairs

92 چند جمله ای ها

Root
93 Poly
Polyval
Polyfit
94 Ginput
.Polyder
Polyint
Conv
Deconv
95 توابع سمبلیک

Syms
Eval
97 Limit
Diff
Int
98 Compose
Symsum
Finverse
99 Jacobian
100..... Reference

الگوریتم والگوریتم نویسی

به جرات می توان گفت بیشتر کسانی که می خواهند برنامه نویسی را شروع کنند، یک کتاب تهیه کرده و شروع به یاد گیری و حفظ دستورات کتاب می کنند ولی مطلبی را که باید مد نظر داشت این است که برنامه نویسی چیزی جز حل مسئله نیست یعنی به عبارتی در مرحله اول اصلا لازم نیست سراغ نرم افزار و دستورات آن برویم . فقط باید مسئله را حل کنیم به هر روش و راهی فقط صحیح و دقیق .

البته اگر راه حلی که ارائه کردیم به صورت توصیفی و باشد ، باید به صورت ریاضی در آورده شود .

مرحله بعدی نوشتمن الگوریتم است .

الگوریتم یعنی انجام مرحله هر کار

می توان گفت مهمترین و اساسی ترین قسمت یک برنامه و برنامه نویسی الگوریتم والگوریتم نویسی آن است

باید این مطلب را اشاره کنیم که برنامه نویسان موفق در ابتدا الگوریتم نویسان خوبی هستند .

حال از روی مسئله حل شده الگوریتم مربوطه را می نویسیم باید توجه کرد که باید کوچکترین جزئیات نیز باید در نظر گرفته شود .

برای تفهیم بهتر به ارائه مثالی می پردازیم(روش حل و محاسبه فاکتوریل)

همه می دانیم فاکتوریل چیست عدد ارائه شده را در یکی کمتر در یکی کمتر و....تا یک ضرب می کنیم .

حاصل فاکتوریل عدد مربوطه است ، ولی این یک تعریف بود و همانطور که گفتیم باید به صورت ریاضی بیان شود .

می توانیم بدین گونه کار را پیش ببریم .

$$N! = n * n-1 * n-2 * \dots * 1$$

$$N! = \prod (1-n) N$$

ما فرمول ریاضی آن را هم نوشتیم ولی چگونه به الگوریتم تبدیل نماییم
گفتیم باید جزئی ترین اعمال نیز مد نظر قرار گیرد

الگوریتم فوق را می توان بدینگونه نوشت

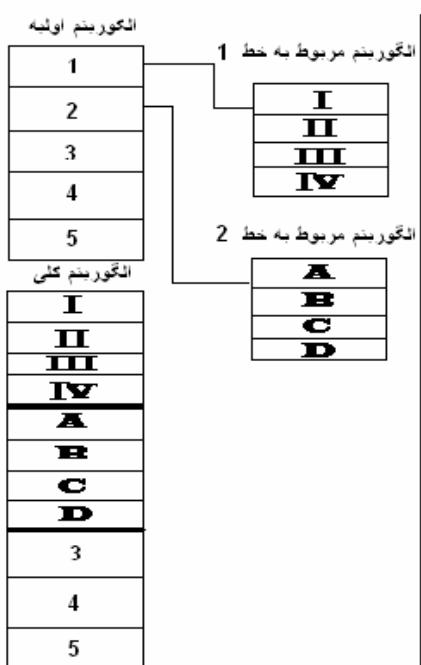
- 1- در نظر گرفتن عدد
- 2- در نظر گرفتن عدد دوم یکی کوچکتر از عدد اصلی
- 3- ضرب دو عدد
- 4- گزاردن حاصل به جای عدد اصلی
- 5- اگر عدد دوم یک است محاسبه تمام شده است
- 6- گزاردن یک عدد کوچکتر از عدد دوم به جای عدد دوم
- 7- تکرار محاسبه از سه (برو به سه)

حال می خواهیم این الگوریتم را پیاده کنیم

در بعضی نرم افزار ها حتی جمع و ضرب نیز تعریف نشده است یعنی ما باید الگوریتم های مربوط به جمع و ضرب را نیز بنویسیم (در متلب این چنین نیست و تقریبا همه توابع ریاضی موجود است).

بنا به جمله بالا باید برای هر سطر نیز الگوریتم خود را جای دهیم مثلا برای سطر 3 ضرب دو عدد را تعریف کنیم و البته برای هر سطر الگوریتمی طراحی شد کلا در جای سطر مربوطه قرار میگیرد و بعد بقیه سطر ها نوشته می شود .

شكل و توضیحات زیر، می تواند کمکی برای بهتر جا افتادن ترتیب در اجرای نوشتن الگوریتم باشد.



شکل (وبدو) را میتوان به یک الگوریتم نسبت داد
به گونه ای که هر فانه ای از جدول به یک عملیات
افتصاص دارد (بلوک های ۱۲۳۴۵)

در صورتی که هر کدام از عملیات . فود شامل
چندین عملیات باشد همه آنها به جای عمل ۹
دستور مربوطه قرار فواهد گرفت
بلوکهای سمت راست فود عملیاتی برای بلوکهای
۱ و ۲ سمت چپ هستند که باید این بلوکها در
الگوریتم اصلی به جای فقط و یا دستور مربوطه
های داده شود .

در یک جمله بگوییم که باید همه کارها را به ترتیب باید پیش برد .
هر پارامتر و متغیری که در رابطه ای استفاده میشود باید قبل تعریف شده باشد .

MATLAB

بدون توضیح در مورد متلب و چگونگی پیدایش و گسترش آن ، به نحوه کاربرد و استفاده آن می پردازیم .

نصب متلب

با گذاشتن CD نصب متلب برنامه نصب به طور خودکار شروع به کار می کند . البته ممکن است بدليل نقص سیستم عامل قبل از نصب متلب ، نرم افزار java virtual machine نصب شود و پس از آن کامپیوتر دوباره راه اندازی شود .

پس اجرای برنامه نصب کامپیوتر یک کد PLP خواهد خواست که در CD نصب وجود دارد و کدی شبیه12345-12345-13 که دو رقم اول شماره نسخه و بقیه اعداد پنج حرفی است

پس از آن مسیر و نوع نصب درخواست میشود که سه نوع نصب وجودارد :

1 – فقط برنامه (product only)

2 – فقط (documentation only) help

3 – هم برنامه و هم (documentation and product) help

پس از اعمال تنظیمات مربوطه کامپیوتر شروع به نصب خواهد کرد که اگر به صورت کامل نصب کنید کمی بیشتر از پانزده دقیقه طول خواهد کشید که البته CD دوم (در صورتی که بسته نصب دو cd داشته باشد) پرونده های HELP نرم افزار است که می توان آن را نصب ننمود (skip documentation) و یا

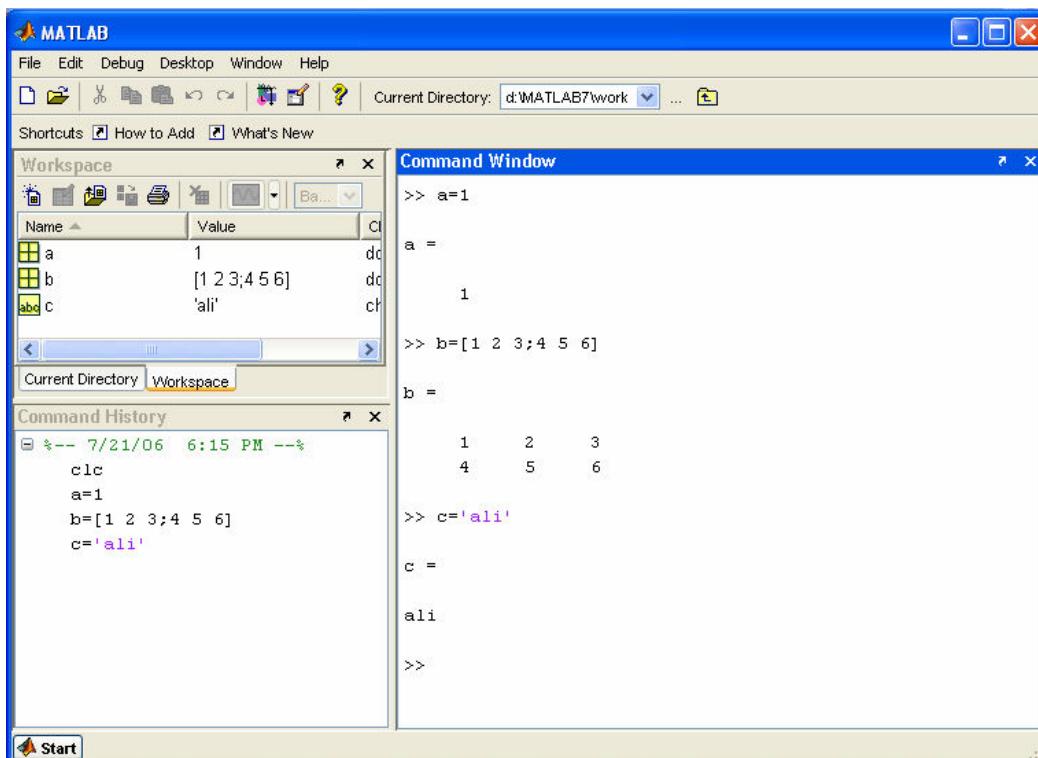
(skip 2) ولی توصیه می کنیم که نرم افزار را به صورت کامل نصب نمائید .

البته نصب نرم افزار بسته به نسخه مربوطه از 1 GB تا 2 GB به فضای خواهد داشد پس متلب را در درایو ویندوز نصب ننمائید .

در صورتی که می خواهید فضای کمی استفاده کنید در حالت انتخابی نصب را دنبال کنید و ابزار هایی که احتیاج ندارید را نصب نکنید .

پنجره های متلب

اگر متلب را برای اولین بار باز کرده باشید صفحه ای مانند شکل زیر را خواهید دید(البته در این صفحه نشان داده نشده است)



وقتی متلب را برای اولین بار راه اندازی میکنیم پنج پنجره را خواهیم دید .

command window – 1

پنجره دستور ... که می توانیم همه دستورات متلب را ، البته به صورت سطحی (فقط یک دستور) در آن اجرا کنیم و همینطور پاسخ اجرای دستورات در اینجا نمایش داده می شود .
(پنجره سمت راست در شکل بالا)

command history – 2

پنجره ای است که همه دستورات اجرا شده در command window را بایگانی می کند .
(پنجره کوچک پایین و سمت راست در شکل بالا)

work space – 3

مکانی است که همه پارامترها و ماتریس‌های تعریف شده در آن نگهداری می‌شود.

(پنجره کوچک بالا و سمت چپ در شکل بالا)

البته اگر بر روی هر کدام از پارامترها دبل کلیک کنیم پنجره‌ای بنام editor باز خواهد شد که می‌توانیم همه پارامترها را ویرایش کنیم.

launch pad – 4

ابزاری برای دسترسی آسان به دمو‌ها جعبه ابزارها، راهنمایی و... است که البته در متلب 7 و بالاتر توسط دکمه start خود مطلب در سمت چپ پایین صفحه می‌توان آن را دید که با کلیک بر روی آن همه ابزارها، دموها و... لیست شده‌اند.

(در شکل بالا launch pad نشان داده نشده است ولی در سمت چپ و پایین میتوانید دکمه start را ببینید)

current directory – 5

این پنجره مکان فایلی که برنامه در حال اجرا در مطلب در آن قرار دارد را نشان میدهد که البته همه فایل‌های موجود را نیز نمایش میدهد. این شاخه عموماً و البته در زمان راه اندازی به مسیر شاخه work منتقل می‌شود که در مسیر MATLAB7\work \ قرار دارد.

هر دستور ورودی و خروجی در این فایل انجام خواهد شد (یعنی اگر پارامتری را بخواهیم ذخیره کنیم در این شاخه و این فایل ذخیره خواهد شد) و البته اگر بخواهیم برنامه‌ای را اجرا کنیم باید در این شاخه باشد که البته در صورت یکی نبودن شاخه‌ها مطلب خود شاخه را تغییر خواهد داد. current directory را می‌توان هم از پنجره current directory وهم فشار دادن دکمه مربوطه در نوار ابزار، تغییر داد.

کار در Command window

همانطور که گفتم command window یا پنجره دستور ، توانایی انجام همه دستورات متلب را دارد و همینطور پاسخ همه دستورات و برنامه های اجرا شده در command نمایش داده می شود .

(در صورتی که در جایی command استفاده شد همان command window است .)

حال چگونه در command کار کنیم :

وقتی متلب را اجرا کردید (باز کردید) تا وقتی در command علامت >> ظاهر نشده است، کامپیوتر آماده نیست و باید منتظر شد

وارد کردن دستوری در command بدین صورت است که دستور را مقابل >> می نویسیم و Enter می کنیم مثلا برای تعریف یک پارامتر

>>a=3{enter}

(همین سه حرف را بنویسید و Enter کنید)

-- {enter} به معنی فشار دادن کلید Enter میباشد

بلا فاصله بعداز فشار دادن کلید Enter پیغام روبرو ظاهر میشود و به معنی قرار دادن عدد 3 در a می باشد و باز بلا فاصله علامت >> ظاهر میشود که بیانگر انتظار کامپیوتر برای دستور بعدی است .

a=

3

>>

حال این پارامتر را با یک عدد جمع کنید :

>>a+5{enter}

ans=

8

ملاحظه می کنید که در پاسخ دومین دستور ans=8 چاپ شد به طوریکه در اولین دستور a=3 شده بود این تفاوت به این دلیل است که در دستور اولی یک پارامتر تعریف کردیم و کامپیوتر پاسخ داد که عدد را در پارامتر قرار دادم ولی در دومی یک عمل محاسباتی انجام دادیم و کامپیوتر پاسخ داد جواب این است (ans مخفف answer) .

بدین صورت میتوانید از command به عنوان یک ماشین حساب استفاده کنید .

بد نیست این مطلب را به خاطر داشته باشید که دستورات در متلب با حروف کوچک وارد میشود .

تعریف متغیر و ماتریس

در مطلب همه پارامترها و اعداد به صورت ماتریس شناخته می شوند حتی اگر یک عدد معمولی باشد به عنوان یک ماتریس 1^*1 شناخته خواهد شد و همه اعمال ریاضی بر اساس قوانین جبر ماتریسی است . شاید برایتان جالب باشد اگر بگوییم همه محاسبات ریاضی که ما خودمان انجام میدهیم بر اساس جبر ماتریسی است ... (بدین دلیل که همه پارامتر های ما عدد معمولی بوده - ماتریس 1^*1 - که به خودی خود معادلات محاسبات ، ساده شده به روابط ساده ریاضی تبدیل می شود) .

در مطلب مانند ریاضیات ماتریس را با (کروشه یا برآکت ... []) نشان می دهند .
اعدادی که داخل برآکت قرار می گیرند معرف درایه های ماتریس مربوطه می باشند .
دریک ماتریس ستونها را با ، (ویرگول) و یا (فاصله) جدا می کنیم و سطر ها را با ; (سمیکولون) و یا زدن enter ورftن به سطر بعد می توانیم جدا کنیم .
چهار ماتریس زیر با هم برابرند :

$$[1 \ 2 \ 3; 4 \ 5 \ 6; 7 \ 8 \ 9]$$

$$[1,2,3;4,5,6;7,8,9]$$

$$\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix}$$

$$\begin{bmatrix} 1,2,3 \\ 4,5,6 \\ 7,8,9 \end{bmatrix}$$

همیشه به خاطر داشته باشید که مطلب برروی حروفات بکار رفته در پارامتر ها حساس است بدین صورت که حروفات بزرگ و کوچک یکسان شناخته نمی شوند .
برای مثال دو پارامتر a و A با هم برابر نیستند و اگر در جایی حرف کوچک و در جایی حرف بزرگ بکار ببریم کامپیوتر حتما خطأ خواهد گرفت.

خودتان چند ماتریس با سطر و ستون های متفاوت بسازید(البته با نام) تا در ادامه مطالب از آنها استفاده کنید .

اندیس ماتریس

حال یک ماتریس ساخته ایم ولی می خواهیم عضوی ام آن را پیدا کنیم ...

در متلب در هر ماتریسی ، هر درایه ای را می توانیم آدرس دهی کنیم .

اندیس گذاری در متلب به دو صورت است 1- شماره درایه 2- اندیس ماتریس(سطر و ستون)

آدرس هر درایه با عددی داخل پرانتز(n) مقابله نام پارامتر (name) نشان داده میشود .

NAME(n)

برای مثال نخست یک ماتریس تعریف میکنیم

>>A=[1 2 3 4 5 6 7 8 9 0];{enter}

>>A(2){enter}

این دستور درایه دوم ماتریس A را میخواهد

Ans=

2

اگر ماتریس دو بعدی باشد در دستور مربوطه باید بعد را نیز مشخص کنیم .

NAME(dim,n)

و باز اگر چند بعدی باشد :

NAME(dim1,dim2,...,n)

که می توانیم این دستور را به صورت ماتریسی بدینگونه نوشت :

>>A(1,2){enter}

این دستور یعنی سطر اول و ستون دوم

برای مثال :

>>A=[1 2 3;4 5 6;7 8 9]{enter}

>>A(2,3){enter}

Ans=

6

اگر ماتریس چند بعدی را فقط با یک عدد اندیس گذاری کنیم متلب اندیس را بر حسب شماره درایه محاسبه خواهد کرد.

شماره درایه به شماره مکان درایه در ماتریس گفته می شود که اندیس ۱ از بالا و سمت چپ شروع و آخرین اندیس پایین و سمت راست خواهد بود و اندیس در یک ستون با پایین رفتن سطر اضافه میشود و در صورتی که به سطر آخر رسید از سطر اول ستون بعدی ادامه پیدا می کند.

می توانیم سطر و یا ستون خاصی را آدرس دهی کنیم:
ستون $a(:,n)$ با $a(n,:)$ و سطر $a(m,:)$ با $a(:,m)$ نشان داده میشود
در مثال بالا سطر دوم:

`>>A(2,:){enter}`

Ans=
4 5 6

اگر بخواهید همه درایه های ستون اول را برابر با یک قرار دهید:

`>>A(:,1)=1{enter}`

Ans=
1 2 3
1 5 6
1 8 9

در صورتی که بخواهیم سطر یا ستون یک ماتریس را حذف کنیم، فقط باید سطر یا ستون مربوطه را برابر [] (ماتریس تهی) قرار دهیم.

`>>A(3,:)=[]{enter}`

Ans=
1 2 3
1 5 6

مالحظه می کنید که در ماتریس A که ستون آن را یک قرار داده بودیم سطر سوم حذف می شود.

میتوانیم اندیس ماتریس را به شیوه دیگری نیز بیان کنیم و آن بیان کردن بوسیله یک بردار است.
(نحوه تشکیل و موارد استفاده از این بردار به طور کامل شرح داده خواهد شد)

`>>A(1:2,1:2:3){enter}`

این دستور یعنی درایه های A، سطر ۱ تا ۲ و ستون ۱ تا ۳ با پرش ۲ (یعنی یکی در میان)

Ans=
1 3
1 6

((از این به بعد نوشته نهی شود و خطا نماید))
`>> شروع شده باید {enter} کرد`

عملگرها و توابع

اگر توضیحات ارائه شده در مورد دستور و یا تابعی کافی نبود و یا در اجرای دستور با مشکل مواجه شدید می توانید با استفاده از دستور help توضیحات مربوط به خود دستور (واقع در خود تابع) را مطالعه کنید که توضیحات در command نمایش داده خواهد شد .

```
>>help input
```

ویا این دستور را اجرا کنید :

```
>>help *
```

ویا از help خود مطلب استفاده کنید (این پرونده ها اصلا ربطی به مطلب ندارند حتی ممکن است تابعی را از آن در آورده باشید ولی در tool box نباشد و اجرا نشود) که با کمک دستور doc متوانیم به این ابزار دست یابیم .

```
>>doc input
```

می بینید که در اجرای اولین و دومین مثال اطلاعات در command چاپ شد ولی در سومین مثال پنجره ای به نام help باز می شود که شامل ابزاری برای جستجو و ... است .

عملگر ها

[] براکت محدود کننده آرایه

هر عدد و یا هر رشته ای در داخل براکت قرار گیرد به عنوان درایه های ماتریس شناخته می شود
برای مثال آرایه ای از اعداد را که ماتریس می گوییم
[1,2,3,4]
و آرایه ای از رشته (کاراکتر)
['ali']

در هر آرایه ای ستون را با ، (ویرگول) و یا () (پرانتز) و سطر را با ; (سمیکولون) و یا با {enter} (فشار دادن کلید enterالبته تا هر وقت براکت را نبندیم ماتریس بسته نمی شود)

کولون :

این عملگر برای تعیین محدوده در یک آرایه بکار می رود راحتتر بگوییم این عملگر به معنی تا است
یعنی اگر بخواهیم یک ماتریس (از ... تا) عدد معینی بسازیم این عملگر به راحتی این کار را انجام می دهد
برای مثال از 1 تا 9 میخواهیم یک آرایه بنام q بسازیم

>>q=1:9

q=
1 2 3 4 5 6 7 8 9

حال اگر بخواهیم ماتریس دیگری بنام w از 1 تا 9 به صورت یک در میان (1 و 3 و 5) تشکیل دهیم
برای اعمال پرسش در ساخت بردار کافیست ماین ابتدا وانتها (1:9) عدد پرسش را بگذاریم (1:2:9)
>>W=1:2:9

W=
1 3 5 7 9

برای نشان داده نشدن نتیجه دستور بکار میرود
برای روشن شدن مفهوم این جمله به مثالهای زیر توجه فرمائید

>>A=3

A=

3

>>A=3;

>>

می بینیم نتیجه دومین دستور که انتهای آن ; وجود دارد نشان داده نشده است و بلافاصله کامپیوتر منتظر دستور بعدی است .

جمع و تفریق

- +

عملگر جمع و تفریق

اگر برای جمع و تفریق دو ماتریس استفاده می شود ، دو ماتریس باید هم مرتبه باشد .
و اگر یک عدد به ماتریسی اضافه یا کسر گردد آن عدد به تمام درایه های ماتریس اثر خواهد کرد .

>>[1 2]+[3 4]

Ans=
3 6

>>[1 2 3;4 5 6]+4

Ans=
5 6 7
8 9 10

باید حتما درجه دو ماتریس با هم سازگار باشند.

```
>>A=[1 2 3];
>>B=[1;2;3];
```

```
>>A*B
```

```
Ans=
14
>>B*A
```

```
Ans=
```

```
1 2 3
2 4 6
3 6 9
```

دلیل تفاوت پاسخ در قواعد ضرب ماتریسی است

```
A(n,m)*B(m,n)=ans(n,n)
B(m,n)*A(n,m)=ans(m,m)
```

```
F(q,w)*H(w,e)=ans(q,e)
```

تقسیم از چپ به راست /

حاصل تقسیم مقدار سمت چپی را بر راستی بر می گرداند که باز در صورت ماتریس بودن باید سازگار باشد.

```
>>10/2
```

```
Ans=
5
```

تقسیم از راست به چپ \

دقیقا مانند تقسیم چپ به راست است با این تفاوت که فقط حاصل تقسیم مقدار راستی بر چپی را بر می گرداند.

```
>>10\2
```

```
Ans=
.2
```

مقدار توان هر مقداری را برمی گرداند

>>2^3

Ans=

8

ترانهاده ماتریس ' کوتیشن

ترانهاده یعنی تعویض سطر و ستون هر درایه در ماتریس که ماتریس دقیقا حول محور اصلی می چرخد.

>>A=[1 2 3 4]

>>A'

Ans=

1

2

3

4

ضرب درایه به درایه *.

حاصل ضرب درایه به درایه دو ماتریس هم مرتبه را برمی گرداند.

>>A=[1 2 3 4]

>>a.*a

ans=

1 4 9 16

همان $[1 \ 2 \ 3 \ 4] * [1 \ 2 \ 3 \ 4]$ بوده که بصورت درایه به درایه ضرب شده است.

تقسیم درایه به درایه \ ./

دقیقاً مانند ضرب درایه به درایه است فقط دو عدد تقسیم می‌شوند.

>>[1 3 4 7]./[2 1 4 3.5]

Ans=
.5 3 1 2

توان درایه به درایه .^

توان درایه به درایه را برابر می‌گرداند

>>[1 2 3].^[1 2 3]

Ans=
1 4 27

ترانهاده آرایه (غیر مزدوج) !

ترانهاده آرایه را برابر می‌گرداند.

(آرایه می‌تواند هر چیزی باشد مثلاً رشته یا کاراکتر..... ماتریس آرایه‌ای از اعداد است)

کوچکتر	<
بزرگتر	>
کوچکتر مساوی	\leq
بزرگتر مساوی	\geq
مساوی (برابر)	$=$
نامساوی	\sim
و	&
یا	

این عملگر ها بیشتر در بیان و ترکیب شرط استفاده میشود .

لطفا به این مثال توجه فرمائید:

```
>>A=3;
>>A==0
```

یعنی آیا A صفر است یا نه (در صورت مثبت بودن 1 و در صورت منفی بودن 0)

```
Ans=
0
>>B=[1 2 3];
>>A>=3 & b(2)==2
```

```
Ans=
1
```

باید همیشه به خاطر داشته باشیم که در رابطه های ترکیبی تقدم عملگر ها را رعایت کنیم بدینگونه که اول توان عمل میکند بعد (ضرب و تقسیم با هم) و بعد (جمع و تفریق با هم) ...

برای مثال این رابطه در ریاضی $(2\sin^2x+2^2)$ است اگر بخواهیم این رابطه را وارد کنیم باید بینیم \sin^2x+2^2 در داخل sin است یا نه

والبته اگر بخواهیم $2x+2^2$ را که همان 2^2+x^2 است را تحلیل کنیم چگونه می شود همانطور که گفتیم اول توان x^2 بعد ضرب x^2 و بعد جمع عمل میکند

ولی اگر بخواهیم بجز این ترتیبی که گفته شد بنویسیم باید از پرانتز استفاده کنیم پرانتز عملگری است که عملیات داخل ، با خارج از آن ارتباطی ندارد .

پارامتر های اولیه

در مطلب چند پارامتر به عنوان پارامتر های اولیه معرفی شده اند مانند عدد π ...تعریف نشده ...بی نهایت و.....که باز می توانیم بوسیله مقدار دهی ...مقدار آنها تغییر داد و برای برگرداندن آنها به مقدار اولیه کافیست دستور clear را به کار ببریم (دستور clear) پارامتر ها را پاک کرده و پارامتر های اولیه را به حالت اول برمی گرداند .

عدد π π

عدد π 3.1415 را با دقت خیلی زیاد ارائه می کند .

i ثابت موهومی

ثابت موهومی یا همان رادیکال $\sqrt{-1}$ است که در اعداد مختلط ، ثابت موهومی را تشکیل می دهد .

عدد مختلط را می توان بدینگونه تعریف کرد که $x+iy$ که x قسمت حقیقی و y قسمت موهومی است و i همان ثابت موهومی است .

```
>>sqrt(-1)
```

```
ans=
```



```
0 + 1.0000i
```

```
>>i^2
```

```
ans=
```



```
-1
```

اپسیلون eps

کوچکترین عدد ممکن یا همان اپسیلون

>>eps

Ans=
2.2204e-016

در پاسخ بالا $2.2204e-016$ $2.2204e-016$ یعنی ده به توان منفی شانزده.

بینهایت inf

این متغیر بیشتر در پاسخی که کامپیوتر می دهد دیده میشود.
یعنی از محدوده شناخته شده برای کامپیوتر خارج است.

>>200^200

Ans=
inf

مبهوم NaN

مخفف not a number است که در صورت $0/0$ رخ می دهد.

>>0/0

Ans=
NaN

کوچکترین و بزرگترین عدد صحیح مثبت realmin & realmax

>>realmin,realmax

Ans=
2.2251e-308
Ans=
1.7977e+308

دستورات ابتدایی

قبل از همه چیز این مطلب را یاداوری میکنیم که حتما وحتما باید دستورات را با حروف کوچک تایپ کنید در غیر این صورت کامپیوتر پیغام خطأ خواهد داد به مثال زیر توجه فرمائید
((اگر در این نوشته ها هم با حروف بزرگ تایپ شده باشد ، اشتباه است ،
شما با حروف کوچک تایپ نمایید))

```
>>disp 'ali'  
ali
```

جلو تر خواهیم دید که disp دستور نمایش دادن است
حال اگر فقط یک حرف آن را با حرف بزرگ بنویسیم
>>Disp 'ali'{enter}

پیغام خطای معنی کنید ((دستور یا تابع تعریف نشده)) که بدلیل اشتباه وارد کردن دستور رخ داده است .

در یافت مقداری، باید متغیر input

این دستور در موقع اجرا مقداری را از کاربر درخواست می کند و در زمان اجرا مدامی که عددی وارد نشود سیستم منتظر می ماند(منتظر ماندن سیستم را می توانید از ناپذید شدن <> متوجه شوید) فرم بکار گیری این دستور بدین صورت است :

```
input('string')  
input('string','kind')
```

سطر اول بکار گیری معمولی دستور را نشان می دهد
در این دستور به جای string هر چیزی می توانیم بنویسیم این نوشته در هنگام اجرای دستور چاپ خواهد شد به مثال زیر توجه فرمائید :

```
input(' ')
```

سیستم بدون هیچ علامت، منتظر در بافت ورودی، است.

```
>>>input(' please enter the number :')  
please enter the number number
```

سیستم عبارت داخل ' چاپ شده و منتظر دریافت ورودی میماند .

اگر بخواهیم در پاسخ دستور یک عبارت رشته ای وارد شود می توانیم پاسخ وارد شده را در داخل

'وارد نماییم'

```
>>input('please enter your name')  
please enter your name  
'ali'
```

Ans=
ali

در مثال فوق سطر سوم پاسخ وارد شده است.

ولی اگر بخواهیم عبارت را بدون 'وارد کنیم' باید نوع ورودی را نیز معین کنیم که از فرم دستور زیر استفاده میکنیم.

```
>>input('please enter your name','s')  
please enter your name  
ali
```

Ans=
ali

ملاحظه می شود که در پاسخ سطر سوم دیگر 'استفاده نشده است و آن بدلیل معرفی کردن نوع ورودی در دستور در قسمت انتهائی دستور..('s)... این را می رساند که مقدار ورودی (s) یا همان رشته است.

اگر در پاسخ این نوع دستور(صرفاً موقعي که ورودی را رشته تعریف کرده ایم) عدد وارد کنیم به عنوان رشته ای از اعداد شناخته خواهد شد و هیچ ارزش و معنی ریاضی نخواهد داشت.

```
>>input('please enter the number','s')  
please enter the nunmber  
123
```

Ans=
123

می بینید که نتیجه اعلام شده برابر 123 است شاید تصور کنید که دستور عدد را گرفته است... ولی باید این را بگوییم که سیستم یک دو سه شناخته است نه صد و بیست و سه.

با همه این توضیحات این را باید بگوییم که تا به حال فقط دستور را اجرا کردیم ولی اگر بخواهیم آن را بکار ببریم باید مانند تعریف پارامتر عمل کنیم و مقدار دریافتی را در متغیری قرار دهیم.

```
>>A=input('please enter the number :')  
Please enter the number  
123
```

A=
123

اگر در دستور `input` میخواهید نوشه های چاپ شده چندین سطر باشد می توانید از `\n` استفاده کنید
بدین صورت که هر جایی از جمله `\n` باشد بقیه جمله از خط بعد چاپ می شود .

```
>>A=input('hello \n how are you \n please enter the number')
Hello
How are you
Please enter the number
```

می بینید که در هر جایی که `\n` بکار رفته چاپ از خط بعدی ادامه پیدا کرده است .

نمایش آرایه یا متغیر `disp`

برای نشان دادن یک آرایه و یا یک متغیر عددی و یا رشته ای بکار می رود .

البته برای نشان دادن آرایه می توانیم بدینگونه عمل کنیم که نام آرایه را وارد کنیم و `{enter}` کنیم تا نام آرایه و مقدار آن نمایش داده شود . ولی وقتی بخواهیم نام آرایه نمایش داده نشود چکار باید بکنیم .
`>>a=3;`
`>>a`

```
a=
3
```

قسمت دوم جواب ارائه شده می باشد .

حال دستور `disp` این کار را انجام میدهد .

```
>>a=3;
>>disp(a)
```

3

که در پاسخ این دستور فقط عدد 3 چاپ می شود .

حال می خواهیم یک جمله را قبل از عدد مربوطه چاپ کنیم .

```
>>a=3;
>>disp 'your answer is';disp(a)
```

```
your answer is
3
```

همانطور که در مثال می بینید در وارد کردن دستور (قسمت اول مثال بالا) دو دستور را پشت سر هم نوشته ایم و یک ; ما بین آنها قرار داده ایم :

در هر جایی ، هر دستوری را می توان مانند روش بالا در
یک خط نوشت که در صورتی می خواهیم نتیجه نمایش
داده نشود با ; جدا می کنیم و اگر بخواهیم نتیجه
نمایش داده شود با ، دستورات را از هم جدا می کنیم .

در دستور بالا نیز بدین دلیل پشت سرهم وارد کرده ایم که اگر بتنهایی وارد می شد پاسخ هر دستور
بطور مجزا چاپ شده و پاسخ ارائه شده در بالا (اول چاپ رشته و بعد عدد) بصورت بالا نمی شد .

پاک کردن صفحه نمایش clc

صفحه نمایش (command window) را پاک کرده و مکان نما را به اولین خط صفحه می برد .

توجه فرمائید که این دستور اصلا به متغیر ها کاری ندارد و فقط صفحه را پاک می کند .

```
>>A=3;  
clc
```

با این دستورات اول متغیری وارد کردیم پس از آن صفحه را پاک کردیم .

```
>>A
```

```
A=  
3
```

پس از پاک کردن صفحه A را فرا خوانی میکنیم و می بینیم که متغیر پاک نشده است .

بردن مکان نما به اول صفحه home

این دستور مانند `clc` عمل میکند ولی با این تفاوت که در `home` صفحه پاک نمی شود و می توانیم با
موس به دستورات و پاسخ های چاپ شده نگاه کنیم ولی در `clc` کل صفحه پاک می شود .

پاک کردن متغیر clear

این دستور متغیر ها را از work space پاک می کند.

فرم بکارگیری این دستور به این شکل می باشد:

clear

clear all

clear parameter

دستورات دو سطر اول و دوم تقریباً برابرند ولی سطر سوم پارامتر مشخص شده را پاک می کند.

به مثال زیر توجه کنید :

```
>>a=1;b=2;c=3;d=4;e=5;f=6;
```

با این دستور شش پارامتر معرفی شده اند که میتوانید با وارد کردن نام آنها پی به وجود آنها ببرید.

```
>>clear a b c
```

با این دستور سه پارامتر اول پاک می شوند که با وارد کردن نام آنها می توان به عدم وجود آن پی برد.

```
>>a
```

```
??? Undefined function or variable 'a'
```

این پیغام یعنی یا دستور وارد شده اشتباه است و یا متغیر وارد شده وجود ندارد ((معنی لفظ به لفظ آن

متغیر یا تابع تعریف نشده است)) که ملاحظه می کنید که با دقت کردن به پیام های سیستم به راحتی

میتوان خطأ و اشتباه موجود را فهمید و رفع نمود.

که در اینجا بدلیل وجود نداشتن پارامتر a این پیغام داده شده است.

حال اگر این دستور را بکار ببریم :

```
>>clear
```

همه پارامتر ها پاک خواهد شد.

تعداد ورودی های تابع nargin

این دستور با وارد کردن نام تابع تعداد متغیر های ورودی تابع را ارائه می کند .

```
>>nargin('sin')
```

Ans=

1

```
>>nargin('*')
```

Ans=

2

تعداد خروجی های تابع nargout

دقیقاً مانند دستور قبلی است با این تفاوت که تعداد خروجی تابع را می دهد .

```
>>nargout('sin')
```

Ans=

1

```
>>nargout('*')
```

Ans=

1

تولید صدای بیپ beep

با اجرای این دستور یک صدای کوتاه بیپ تولید میشود که در برنامه ها می توان درصورت بروزخطا و استفاده کرد .

```
>>beep
```

m.file پنجره ای

هر برنامه نویسی می خواهد بر نامه ای را که می نویسد ، نگهداری کند ، تا در صورت لزوم از آن استفاده کند .

گفتم هر دستوری را می توان در window command اجرا کرد و هر برنامه ای ، اجرای پشت سر هم دستورات می باشد . پس می توان بر نامه ای را بگونه ای که خط به خط دستورات آن را در command وارد کرد ، نوشت . ولی برای استفاده بار دوم و ... چکار باید کرد ؟

در مطلب پنجره ای بنام m-file editor وجود دارد که فضایی شبیه به word pad و یا note دارد که بر اساس کاربرد ابزار هایی بیشتر و یا کمتر دارد .

برنامه ای که ساخته می شود در m-file editor نوشته شده و ذخیره می گردد ---- ولی اگر این پنجره از داخل مطلب باز شده باشد چند ابزار دیگر نیز به این پنجره افزوده می شود که می توان بر نامه نوشته شده را اجرا کرد و پاسخ آن را در command مشاهده کرد (دوباره تکرار می کنیم اگر editor از داخل مطلب باز شده باشد قادر به اجرای برنامه است) .

برای اجرای m-file editor بر روی کلید سمت چپی روی نوار ابزار (شکلی مانند صفحه سفید) کلیک کنید تا این پنجره باز شود و یا از منوی file گزینه new و بعد m-file را انتخاب نمایید .

حال برای نوشتن برنامه در این صفحه باید مرتب و به ترتیب از بالا به پایین دستورات را بنویسید (همانگونه که در الگوریتم نویسی گفته شد) .

پس از اتمام نوشتن برای اجرای برنامه از منوی debug کلید run را فشار دهید و یا کلید F5 در صفحه کلید فشار دهید (البته اگر برنامه ذخیره نشده باشد و یا تغییری در آن ایجاد شده باشد به جای Run کلید save & Run را مشاهده خواهید کرد که البته وقتی برنامه تغییر داده شده باشد یک علامت ستاره در نوار عنوان در کنار اسم برنامه دیده می شود و در صورتی که برای اولین بار کلید را فشار دهیم ، سیستم نام و مسیر ذخیره کردن فایل را خواهد خواست .

این پنجره دارای چندین ابزار حرفه ای برای اجرا و خطایابی و ... ، در برنامه است که بدلیل تخصصی بودن به آنها نمی پردازیم ولی افرادی که تمایل به فرآگیری آن دارند به شاخه \MATLAB\demos\ رجوع کنند البته این فایلها در CD همراه وجود دارد .

اگر نام برنامه (البته در current directory باشد) را در command وارد شود برنامه اجرا خواهد شد .

باید بگوییم که هر برنامه نوشته شده را میتوان به عنوان یک تابع استفاده نمود .

حال اگر این برنامه ورودی و خروجی نداشته باشد می توان بدون هیچ تغییری به عنوان تابع استفاده کرد ولی اگر بخواهید برنامه را به صورت تابع استفاده کنید و یا یک تابع بنویسد باید قوانینی را رعایت کنید که در ادامه به آنها می پردازیم .

فرض کنید برنامه‌ی را که نوشته‌اید و آن را اجرا کرده‌اید و پاسخ صحیح گرفته‌اید به کسی می‌دهید تا از آن برای اهداف خود استفاده کند.....یا از روی آن برنامه‌ای بنویسد و یا آن را تغییر دهد . برنامه خیلی حجمی و بزرگ است و درک کلی آن نیاز به صرف وقت و انرژی فراوان دارد باید راهی وجود داشته باشد که درک و تفسیر برنامه را راحت‌تر کند و حتی تغییر و اصلاح برنامه را سرعت بخشد .

در m-file هر حرف و جمله‌ای که پس از % نوشته شود در اجرای برنامه تاثیر ندارد یعنی می‌توان گفت در موقع اجرای برنامه خوانده نمی‌شود و همیشه این نوشته‌ها به رنگ سبز نمایش داده می‌شود .

پس بدین صورت می‌توانیم اجزای برنامه را از هم جدا کنیم .

یک ابزار در متلب دستور help است که با وارد کردن نام تابع مقابل آن مشخصات و نحوه استفاده از تابع مر بوطه را نمایش میدهد (یک تابع را امتحان کنید)
>>help factorial

تابع factorial همان فاکتوریل میباشد که با وارد کردن دستور بالا توضیحات مربوطه در command چاپ می‌شود .

باید بگوییم توضیحات چاپ شده همه در داخل m-file مربوطه ، با % نوشته شده است شاید جمله بالا برایتان کمی گیج کننده باشد . راحت‌تر می‌گوییم : در هر m-file توضیحاتی -(با % نوشته میشوند)- که در اول برنامه نوشته می‌شوند با اجرای help برای آن m-file همان نوشته‌ها چاپ می‌شوند . برای درک راحت موضوع به یکی از m_file ها در toolbox رجوع کنید و به نوشته‌های سبز رنگ توجه کنید و بعد در command نام فایلی را که نگاه کرده بودید را همراه با دستور help وارد کنید (مانند مثال ارائه شده در بالا) می‌بینید که همه نوشته‌های سبز رنگ چاپ می‌شود .

این بخش را بدین دلیل بعد از پنجه‌های متلب نگفتم که بدون هیچ پایه‌ای [تابع و] ، شروع کردن به برنامه نویسی کمی غیر منطقی و کاری اشتباه است حال می‌توانید با این دستورات ابتدایی اولین برنامه خود را بنویسید .

علامت سوال را از بالای سر خود دور کرده و m-file را باز کرده و شروع کنید ، اول صفحه پاک شود . چند متغیر معرفی کنید (ثابت) . چند متغیر در حین برنامه معرفی کنید (input) . اعمال ریاضی را بر روی آنها انجام دهید و بعد پاسخ را چاپ کنید (disp) . ((فقط همه کارها را به ترتیب انجام دهید))

تابع Function

می توان گفت بیشتر دستور ها و توابعی که در متلب به کار می گیریم دارای یک m-file هستند بدین معنی که برنامه ای برای تابع و یا دستور مربوطه نوشته شده است . ولی همه این توابع از سیستم خاصی که function آنها را موظف می کند پیروی می کنند . پس باید برای ساختن یک تابع کمی متفاوت تر از برنامه معمولی عمل نماییم .

در صورتی که می خواهید مثالی از این توابع ببینید ، میتوانید به شاخه MATLAB7\toolbox\ رفته واز داخل یکی از جعبه ابزار ها یک m-file را باز کنید . میبینید که هر کدام بر نامه ای کامل هستند . ولی چیزی که شاید توجهتان را جلب کندسطر اول همه برنامه ها است که با شروع می شود function output=function name(input) پارامتر ورودی و خروجی هستند و output نام تابع است) . اگر برنامه ای را بصورت تابع نوشته ایم می توانیم در هرجایی و هر برنامه ای استفاده کنیم .

حال چگونه یک تابع بسازیم :

شاید با عنوان کردن این موضوع (ساختن تابع) این سوال برایتان پیش آید که چگونه برنامه نوشته شده را به تابع تبدیل کنیم ؟ باید بگوییم تبدیل برنامه نوشته شده به تابع بجز اضافه و کم کردن چند سطر کاری ندارد یعنی به عبارتی اگر بخواهید یک برنامه کامل را به تابع تبدیل کنید چند دقیقه بیشتر طول نخواهد کشید البته اگر روند ساخت تابع را بدانید این کار را راحت و با تسلط کامل انجام خواهید داد .

همه میدانیم ورودی تابع را با خود تابع وارد می کنیم و در حین اجرای تابع هیچ مقداری وارد نمی کنیم . پس در برنامه نوشته شده برای تابع ، از دریافت هرگونه مقدار و ورودی در حین اجرا باید بدور باشیم و همه ورودی ها را در اول برنامه در دستور function تعریف نماییم . هیچ تابعی پس از اجرا جمله وبا علائم اخباری چاپ نمی کند پس باید از چاپ و نوشته های راهنمایی و ... باید دوری کنیم و همه خروجی ها را باز در دستور function تعریف میکنیم .

خط اول هر برنامه تابعی بدينگونه نوشته می شود :

function output=name(input)

در عبارت بالا :

name ، نام تابعی است که میسازیم که باید این نام با نام فایل ذخیره شده(همان فایلی که است m-file و تابع را در آن نوشته ایم) یکی باشد و گرنم در هنگام استفاده کردن تابع خطای خواهد داد . ((تکرار می کنیم که حتما نام تابع با نام فایل ذخیره شده آن یکی باشد)) .

البته وقتی در سطر اول function را بکار ببریم سیستم به صورت خودکار نام تابع را برای نام فایل قرار خواهد داد.

همان مقادیر ورودی است که در برنامه ها بیشتر با input دریافت میشود. این مقادیر می تواند یک یا چند مقدار و حتی ماتریس و... باشد که به ازای هر کدامیک پارامتر در

داخل پرانتز قرار میدهیم. به این مثال توجه فرمائید:

((از اول یک تابع جمع سه عدد می نویسیم و نام آن را numadd می گذاریم))

(این دستورات به ترتیب در m-file نوشته میشوند)

اولین خط برنامه چنین خواهد بود :

```
function d=numadd(a,b,c)
```

می بینید که در دستور function نام و ورودی و خروجی را تعریف نمودیم ولی اگر می خواستیم این دستورات را با input بنویسیم باید فقط سه بار دستور input استفاده می کردیم.

حال سه عدد ورودی را با هم جمع کنیم و پاسخ برنامه را نمایش دهیم.

```
d=a+b+c;
```

یادتان باشد تابع هایی را که می سازید در انتهای سطر ها ((;)) قرار دهید تا نتیجه اجرای سطر نمایش داده نشود (البته برای بار اول و خطایابی طبیعتاً اینگونه نیست).

با اضافه کردن سطر دوم برنامه تابع جمع سه عدد تمام شد اثرا با فشار دادن کلید save در منوی file ذخیره می کنیم اگر دقت کرده باشید می بینید که با فشار کلید save پنجره ای برای دریافت نام و مسیر از شما می خواهد که مسیر را نباید تغییر داد و البته نام نمایش داده شده نیز نام وارد شده برای تابع است که گفتم باید یکی باشد پس باید بدون هیچ تغییری ذخیره کنیم.

اگر function دارای ورودی نباشد میتوانیم از m-file اجرا نماییم(با فشار دادن F5). برای اجرای یک تابع باورودی ، به هیچ عنوان نمی توانیم مانند اجرا در m-file عمل کنیم بلکه باید نام و ورودی را در command وارد کنیم ((به این دلیل که ورودی را باید در دستور وارد کرد ولی اگر تابع ورودی نداشته باشد میتوان از داخل m-file اجرا نمود)).

```
>>numadd(1,2,3)
```

```
Ans=
```

6

که با اجرای برنامه پاسخ نمایش داده می شود .

اگر توضیحاتی در مورد برنامه می خواهید بدھید باید پس از دستور function (خط اول) باید باشد .

```
function d=numadd(a,b,c)
% this function is additional of the three number
d=a+b+c;
```

گفتنیم چگونه m-file و یا همان برنامه نوشته شده را به function تبدیل نماییم ؟

به نظر می رسد تا حدی با نحوه انجام کار آشنا شده باشید .

ولی در چند جمله کوتاه می گوییم که :

همه ورودی ها (input) پاک شده و پارامتر مربوطه در ورودیتابع قرار داده شود .

همه خروجی ها و هر گونه چاپ و علائم و نوشته ها (disp) باید حذف گردد .

پس از اجرای برنامه تغییر داده شده و بدون خطأ ، در پایان هر خط علامت ; گذارده شود .

کنترل ها در برنامه نویسی

این توابع برای تصمیم گیری برای انجام کاری و یا انجام به دفعات به کار می رود .

این توابع به چند دسته تقسیم می شوند :

توابع شرطی

این تابع در صورت صادق بودن شرط وارد شده دستورات معین شده را if یکبار اجرا میکند .

این تابع تا وقتی که شرط وارد شده برقرار است دستورات وارد شده انجام خواهد شد و تعداد تکرار مهم نیست .

این دستور تصمیم گیری در میان چندین موضوع همسان را بر عهده دارد switch تا حدی شبیه به دستور if می باشد .

توابع چرخه ای

بوسیله این دستور می توانیم قسمتی از برنامه را به تعداد معلوم تکرار for نماییم .

توضیح این دستور در بالا داده شد که تکرار به دفعات نا متناهی را می توان انجام داد .

در پایان همه این دستورات دستور end استفاده می شود که نشان دهنده پایان چرخه می باشد .

if.....end

گفته‌یم این تابع این تابع برای اجای قسمتی از برنامه در صورت صادق بودن شرط وارد شده به کار می رود .

شکل این دستور بدینصورت می باشد :

```
if شرط تعیین شده
    دستورات وارد شده
    :
    :
end
```

اگر شرط صدق کند دستورات داخل این عبارت اجرا می شود .

ولی اگر شرط صدق نکند کل عبارت if نادیده گرفته می شود و اجرای دستورات به بعد از خط end منتقل می شود .

می توان چندین عبارت if را در داخل هم استفاده کرد .

به مثال زیر توجه فرمائید(در m-file نوشته می شود):

```
a=input('please enter 2 and press enter');
if a==2
    disp ('your entered number is 2');
end
```

اجرای برنامه:

Running the program

Please enter 2 and press enter
2

Your entered number is 2

در غیر این صورت else

این دستور در داخل عبارت if استفاده می گردد و در صورتی که شرط وارد شده صدق نکند دستورات وارد شده اجرا می شوند .

به مثال زیر توجه فرمائید(در m-file نوشته می شود):

```
a=input('please enter a number : ');
if a == 2
    disp('your entered number is 2');
else
    disp(' your entered number is not 2');
end
```

اجرای برنامه :

Running the program :

Please enter a number
4

Your entered number is not 2

چند شرط همزمان elseif

ممکن است در یک زمان چند شرط را همزمان و پشت سر هم وارد کنیم این کار را بوسیله elseif انجام می دهیم .

به مثال زیر توجه فرمائید(در m-file نوشته می شود):

```
a=input('please enter a number :')
if a == 2
    disp('your entered number is 2')
elseif a == 3
    disp('your entered number is 3')
elseif a == 4
    disp('your entered number is 4')
else
    disp('your entered number is not 2 3 4 ')
end
```

که اجرای برنامه نیز مانند دو مثال قبل می باشد .

Switchcaseend

گفتیم که این دستور برای تصمیم گیری برای حالت های مختلف بکار می رود .

برای مثال آخرین مثال عبارت قبل را با استفاده از switch مینویسیم :

```

a=input('please enter a number :');
switch a
    case 2
        disp('your entered number is 2')
    case 3
        disp('your entered number is 3')
    case 4
        disp('your entered number is 4')
    otherwise
        disp('your entered number is not 2 3 4 ')
end

```

این برنامه دقیقاً برابر با آخرین برنامه ارائه شده برای if می‌باشد.

for.....end

تکرار به تعداد معین

برای تعیین تعداد این تابع باید یک ماتریس سطروی با همان تعداد درایه معرفی کرد که این ماتریس را عموماً با (:) مشخص می‌کنند که ماتریسی با فاصله درایه‌های یک می‌سازد.

for i=1:10

دستورات
end

می‌بینید که در خط اول تعیین تعداد تکرار با تعریف ماتریس I انجام می‌شود.

هر بار تکرار مربوط به یک درایه می‌باشد و در آن مرحله پارامتر مشخص شده مقدار درایه را دارد.
مثلاً در مثال بالا در اولین مرحله مقدار ۱ می‌باشد و در دومین مرحله مقدار ۲ و ...

حال به این مثال توجه کنید:

```

for i=[1,4,8,4,3]
    disp('level')
    disp(i)
end

```

که این برنامه در مرحله اول مقدار ۱ را که در پارامتر ۱ است چاپ می‌کند و در مرحله دوم مقدار دومین

درایه را در داخل ۱ قرار داده و در خط سوم چاپ می‌کند که ۴ است و در مرحله سوم ۸ و ...

تکرار تا صادق بودن شرط while.....end

در این دستور دقیقاً مانند if شرطی وارد می‌شود ولی نحوه کنترل روند برنامه بدین صورت است که تا وقتی که شرط صادق باشد دستورات داخل عبارت while تکرار خواهد شد. (اگر برنامه اشتباهی داشته باشد که همیشه شرط برقرار باشد سیستم در شرط باقی خواهد ماند).

```
a=input('please enter a number :');
while a ~=1
    if a>1
        a=a-1;
    end
    if a<1
        a=a+1;
    end
disp(a);
end
```

برنامه ارائه شده را اجرا کنید و پس از دیدن پاسخ نحوه کار while را بررسی کنید.

برای اجرای برنامه‌ها کافیست همه را copy کرده و در past.... m-file کنید و F5 را بزنید.

continue

این دستور در چرخه های تکرار استفاده می شود .
در مواقعی که می خواهیم مرحله ای از چرخه انجام نشود و مراحل بعدی انجام شود از این دستور انجام می کنیم .

```
for i=1:10
    if i==5
        disp('This level was jump');
        continue;
    end
i
end
```

می بینید که مرحله پنجم را نادیده گرفته و ادامه می دهد .

break

با اجرای این دستور مسیر برنامه در هر حالتی که باشد کاملا از داخل حلقه خارج خواهد شد .

به این مثال توجه فرمایید :

```
for i=1:10
    if i==5
        disp(' in this level path of running was jump out of statement!');
        break;
    end
i
end
```

می بینید که چاپ رشته تا 10 باید ادامه داشته باشد ولی در مرحله پنجم از حلقه بیرون می پرد .

منطق در شرط

گاهی اوقات ممکن است لازم باشد چندین شرط را ترکیب کنیم . این عمل با استفاده از عبارت های منطقی قابل انجام است .

اگر بخواهیم دو شرط در یک زمان برقرار باشد از (&) و در صورتی که بخواهیم یکی از دو یا چند شرط صدق کند از (|) استفاده می کنیم .

به مثال های زیر توجه فرمایید :

```
a=input('please enter a number');
b=input('please enter a number');
if a>0 & b>0
    disp('a and b are positive')
elseif a<0 & b<0
    disp('a and b are negative')
elseif a>0 | b<0
    disp('a is positive or b is negative ')
elseif a<0 | b>0
    disp('a is negative or b is positive')
end
```

این برنامه را اجرا کنید و جواب ارائه شده را با تحلیل خودتان از برنامه مقایسه کنید .

FORMAT

بوسیله این تابع می توانیم دقت پاسخ ارائه شده از طرف سیستم را تنظیم کرد .

این دستور بدین گونه استفاده می شود :

format kind

که باید format را بنویسیم و بر اساس دقت مربوطه نوع فرمت آن را مینویسیم .
نوع دقت معرفی شده می باشد . Kind

format short

این دستور مقادیر را تا چهار رقم اعشار نشان میدهد .

حال به انواع مختلف فرمت پردازیم .

نوع فرمت

نحوه تاثیر

مثال (تاثیر بر روی عدد پی)

format bank

دو رقم اعشار

3.14

format short

چهار رقم اعشار

3.1416

format short e

چهار رقم اعشار ممیز شنا

3.1416e+000

format short g

بهترین حالت

3.1416

format long

پانزده رقم اعشار

3.14159265358979

format long e

پانزده رقم اعشار ممیز شناور

3.141592653589793 e+000

format long g

بهترین حالت

3.14159265358979

format hex

بر اساس هگزا دسیمال

400921 fb54442d18

format rat

بصورت کسر منطقی

355 / 113

format +

بر اساس تابع علامت sign

+

نحوه کاربرد format هم به این صورت است که فقط دستور را بنویسیم و enter کنیم این دستور هیچ پاسخی ندارد ولی دستور اجرا شده است .

```
>>format long  
>>pi  
Ans=
```

3.14159265358979

```
format short  
Ans=
```

3.1415

بعضی اوقات لازم است که مقادیر را بر اساس مقادیر خاصی گرد کنیم .
در متلب جعبه ابزار تقریبا کاملی برای این کار تهیه شده است .
در پایین این توابع را می بینیم .

تابع	عملکرد تابع	عملکرد برای 2.4	عملکرد برای -2.4
fix	عدد را به سمت صفر گرد می کند	2	-2
floor	عدد را به سمت منفی بینهایت گرد می کند	2	-3
ceil	عدد را به سمت مثبت بینهایت گرد می کند	3	-2
round	عدد را به سمت نزدیکترین همسایگی گرد	2	-2

مثال زیر نحوه استفاده از توانع بالا را نشان می‌دهد.

```
>>fix(2,4)
```

$$\begin{aligned} \text{Ans} &= \\ &2 \end{aligned}$$

نوعی دیگر از این نوع توابع که مقدار باقیمانده از بالا و پایین را نشان می‌دهد.

mod باقیمانده پس از تقسیم از بالا
rem باقیمانده پس از تقسیم از یاری

```
>>mod(-5,3)
```

$$-5+2 \cdot 3 = 1$$

Ans=

>>rem(-5,3)
-5*1+3=-2

$$\text{Ans} =$$

>>rem(5,3) پاسخ این دو دستور برابر است.

```
>>mod(5,3)
```

در صورتی که علامت دو متغیر ورودی برای باشد پاسخ برای mod یکی است.

توابع عددی

اعداد اول primes

این تابع اعداد اول از صفر تا عدد وارد شده را ارائه میکند .

>>primes(11)

Ans=
2 3 5 7 11

تجزیه به اعداد اول factor

این تابع عدد وارد شده را به اعداد اول تجزیه می کند .

>>factor(100)

Ans=
2 2 5 5

فاکتوریل factorial

مقدار فاکتوریل عدد وارد شده را می دهد .

>>farctorial(6)

Ans=
3628800

بزرگترین مقسوم عليه مشترک gcd

بزرگترین مقسوم عليه مشترک دو عدد وارد شده را ارائه می دهد .

>>gcd(12,36)

Ans=
12

>>gcd(12,20)

Ans=
4

کوچکترین مضرب مشترک lcm

کوچکترین مضرب مشترک دو عدد وارد شده را محاسبه می کند .

>>lcm(6,22)

Ans=
66

>>lcm(2,5)

Ans=
10

تواجع مختلط

abs قدر مطلق

مقدار قدر مطلق (ثبت) مقدار ورودی را می دهد .

```
>>abs(-2)
```

Ans= 2

```
>>abs(-i)
```

Ans= 1

در مثال دوم منظور از i همان ثابت موهومی است .

ساخت عدد مختلط complex

با وارد کردن دو عدد به عنوان عدد حقیقی و موهومی ، عدد مختلط مربوطه را می سازد .

```
>>A=complex(2,4)
```

A= 2.0000 + 4.0000i

تعداد ممیز در مثال بالا فقط به format بستگی دارد .

قسمت موهومی عدد مختلط imag

با این دستور می توان به قسمت حقیقی عدد مختلط دست یافت .

```
>>a=complex(2,3);  
>>imag(a)
```

Ans= 3

در این مثال مقدار $a = 2+3i$ شد که مقدار موهومی آن 3 می باشد .

قسمت حقیقی عدد مختلط real

این دستور قسمت حقیقی عدد مختلط را به ما نشان می دهد .

```
>>real(a)
```

Ans=
2

مقدار زاویه قطبی مختلط angle

این تابع مقدار زاویه قطبی را در دستگاه مختلط ارائه می دهد .

ورودی این تابع یک عدد مختلط است که زاویه بردار تشکیل شده بر اساس عدد وارد شده و مبدا را
می دهد .

```
>>angle(1+0i)
```

Ans=
0

```
>>angle(-1+0i)
```

Ans=
3.1415

```
>>angle(1+i)
```

Ans=
0.7854

مزدوج مختلط conj

مقدار مزدوج مختلط را ارائه می کند .

```
>>conj(2-4i)
```

Ans=
2.0000+4.0000i

توابع نمایی

ریشه دوم sqrt

مقدار ریشه دوم یا همان جذر مقدار وارد شده را می دهد .

```
>>sqrt(9)
```

Ans= 3

ریشه دوم ماتریس sqrtm

ریشه دوم ماتریس را محاسبه می کند .

```
>>a=[1,2;1,2];  
>>sqrtm(a)
```

Ans=

0.5774 1.1547
0.5774 1.1547

```
>>a=[2,2;2,2];  
>>sqrtm(a)
```

Ans= 1.0000 1.0000
1.0000 1.0000

ریشه n ام عدد nthroot

ریشه n ام عدد را محاسبه می کند ..

```
>>nthroot(8,3)
```

Ans= 2

توان power

مقدار اول را به توان مقدار دوم می رساند .

```
>>power(2,3)
```

Ans=

8

توان بر مبنای دو pow2

دو را به توان عدد وارد شده می کند .

```
>>pow2(3)
```

Ans=

8

تابع نمایی exp

مقدار تابع نمایی یا همان e به توان x را محاسبه می کند .

```
>>exp(1)
```

Ans=

2.7183

مقدار عدد نپر بدست آمد(همان e^1 را نوشتم) .

لگاریتم log log2 log10

log همان لگاریتم طبیعی یا بر مبنای e است .

log2 لگاریتم بر مبنای دو است .

log10 لگاریتم بر مبنای ده است .

```
>>log(2.7183)
```

ans=

1

لگاریتم ماتریس را میدهد . Logm

توابع مثلثاتی

همانند دیگر ابزار ها در مطلب یک جعبه ابزار خیلی قوی و کامل در مورد توابع مثلثاتی وجود دارد که رفته رفته کامل می شود .

توابع $\sin \cos \tan \cot$ مقادیر سینوس کسینوس تانژانت و کتانژانت مقدار را می دهد .

توابع $\text{asin} \text{acos} \text{atan} \text{acot}$ مقادیر معکوس توابع بالا را می دهد .

توابع $\text{sinh} \text{cosh} \text{tanh} \text{coth}$ مقادیر هیپربولیک توابع بالا را می دهد .

توابع $\text{asinh} \text{acosh} \text{atanh} \text{acoth}$ مقادیر معکوس توابع هیپربولیک را میدهد .

نحوه استفاده از توابع مثلثاتی بدین گونه است که مقدار را بر حسب رادیان مقابل تابع مینویسیم .

```
>>sin(pi/2)
```

```
Ans=1
```

```
>>asin(1)
```

```
Ans=1.5708
```

که دومین پاسخ همان $\pi/2$ است .

در مطلب 7 و بالاتر توابع مربوط به درجه نیز گذاشته شده است که در ادامه دستور فقط حرف d می گذاریم .

```
>>sind(90)
```

```
Ans=1
```

```
>>asind(1)
```

```
Ans=90
```

دستورات منطقی

خالی است یا نهisempty

بیشتر اوقات از input برای انتخاب یکی از چند موضوع عنوان شده بکار میرود در این موارد باید چند عدد خاص را وارد کرد.

فرض کنید در برنامه ای روش روپرتو را بکار ببرید که با وارد کردن عددی به صفحه ای برود و اگر هیچ عددی وارد نشود و enter شود یک عدد را خود به خود در نظر گیرد.

- 1 go to first page
- 2 go to second page
- 3 exit

این نوشه ها پس از اجرای برنامه چاپ می شوند
بنظرتان برنامه آن چگونه نوشته می شود

Please enter(1-3)[1]

مثال بالا دستور اجرا شده را نشان میدهد که باید در ورودی یکی از سه عدد 1 2 3 را وارد کرد
در آخرین خط می بینید که [1] نوشته شده است این بدین معنی است که اگر بدون وارد کردن عددی {enter} کردیم به عنوان ورودی شماره 1 را بر می دارد.

این کار با دستور isempty انجام می شود بدین گونه که دستور مقدار وارد شده را نگاه می کند اگر خالی باشد در پاسخ 1 و اگر خالی نباشد 0 خواهد داد.

```
>>a=input('please enter the number :');
```

please enter the number

بدون وارد کردن عددی {enter} می کنیم.

```
>>isempty(a)
```

```
ans=1
```

عدد است یا نهisnumeric

این دستور نیز مانند isempty می باشد فقط عدد بودن مقدار را بررسی می کند.

```
>>a=8
>>isnumeric(a)
ans=1
```

```
>>a='ali'  
>>isnumeric(a)
```

```
Ans=  
0
```

برابر است یا نه isequal

این دستور نیز مانند دو دستور قبل هستند ولی دو مقدار را مقایسه می کنند .

```
>>a=3  
>>isequal(a,4)
```

```
Ans=  
0
```

حقیقی است یا نه isreal

حقیقی بودن ورودی را بررسی میکند

همه می دانیم یک عدد مختلط $x+iy$ است که اگر y یعنی قسمت موهومی 0 باشد عدد حقیقی میشود

```
>>A=1  
>>isreal(a)
```

```
Ans=  
1
```

عدد اول بودن isprime

با اجرای این فرمان عدد اول بودن مقدار وارد شده بررسی میشود .

```
>>isprime(3)  
Ans=  
1
```

توابع زمانی

زمان جاری کامپیوتر را به صورت یک بردار نشان می دهد . clock

```
>>clock
```

Ans=

1.0 e+003*

2.006 0.007 0.0260 0.0010 0.0360 0.0179

تاریخ را به صورت نوشتاری می دهد . date

```
>>date
```

Ans=

26-Jul-2006

گرفتن زمان در بازه مشخص . tictoc

از زمان شروع یعنی اجرای tic زمان گیری شروع می شود تا وقتی که toc اجرا شود .

زمان بر حسب ثانیه است .

```
>>tic
```

```
>>.....
```

```
>>toc
```

Elapsed time is 10.031000 seconds.

در مثال بالا یعنی می توان ما بین این دستورات tic.. toc.. دستورات دیگری اجرا نمود .

مکث بر حسب ثانیه pause

```
>>pause(10)
```

بر حسب ثانیه وارد شده ادامه کار سیستم را به تعویق می اندازد .

توابع آرایه‌ای

تعداد اعضای آرایه

numel

تعداد درایه‌های یک ماتریس را می‌دهد.

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

Ans=
6

طول بردار

length

تعداد ستونهای یک ماتریس را ارائه می‌کند.

```
>>a=[1 2 3];  
>>length(a)
```

Ans=
3

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

Ans=
3

درایه خاصی را در یک آرایه پیدا می‌کند.

find

```
[n,m]=find(a==k)
```

```
n=find(a== k)
```

دستورات فوق درایه خاصی را در آرایه پیدا می‌کند:

دستور اول درایه مساوی با k را در ماتریس a ، بر حسب سطر و ستون می‌دهد.

در حالی که دستور دوم دقیقاً مانند دستور اول است با این تفاوت که اندیس ماتریس را بر حسب شماره درایه می‌دهد (در اندیس گذاری توضیح داده شده است).

```
>>n=find(s>3)
```

اندازه گیری مرتبه ماتریس size

این دستور برای معلوم شدن تعداد سطر و ستون و لایه و.... بکار می‌رود.

با اجرای این دستور، در پاسخ یک ماتریس 2^*2 تشکیل می‌شود که درایه اول آن تعداد سطر ماتریس معرفی شده و درایه دوم تعداد ستون واگر لایه داشته باشد درایه سوم تعداد لایه می‌باشد و....

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

```
>>size(s)
```

Ans=

2 3

و باز می‌توانیم دستور را اینگونه بکار ببریم:

```
>>[n,m]=size(s)
```

n=

2

m=

3

اگر بخواهیم فقط اندازه یک بعد از ماتریس را بدانیم بدین گونه پیش می‌رویم:
size(s,DIM)
... در اینجا همان بعد است که سطر بعداً سطون، بعد دوم، لایه بعد سوم و DIM

برای تعداد ستون:

```
>>size(s,2)
```

Ans=

3

ماتریس‌های خاص

ماتریس جادویی magic

ماتریس مریع با درجه وارد شده می‌سازد به گونه‌ای که جمع درایه‌های سطر و ستون آن برابر باشد.

>>magic(4)

Ans=
16 2 3 13
5 11 10 8
9 7 6 12
4 14 15 1

ماتریس تصادفی rand

ماتریس تصادفی با درجه وارد شده می‌سازد.

rand(n,m)

این دستور ماتریس n^*m را می‌سازد که درایه‌های آن تصادفی است.

rand(n)

این دستور ماتریس مریع n می‌سازد که درایه‌های آن تصادفی است.

>>rand(5)

ماتریس همانی eye

این دستور یک ماتریس همانی با درجه وارد شده می‌سازد.

eye(n)

n درجه ماتریسی است که می‌خواهیم بسازیم.

>>eye(3)

Ans=
1 0 0
0 1 0
0 0 1

اگر در این دستور دو پارامتر نعرفی 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^*) یک می باشد.

```
>>eye(3,4)
```

Ans=

1	0	0	0
0	1	0	0
0	0	1	0

ماتریس یک ones

ماتریسی با درایه های یک می سازد.

در صورتی که پارامتر تعریف شده یک عدد باشد، ماتریس مریع 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

ماتریس صفر zeros

ماتریسی با درایه های صفر می سازد.

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

ans=

0	0
0	0

عملکرد این دستور دقیقاً مانند ones می باشد.

توابع ماتریسی

بزرگترین درایه در ماتریس \max

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

این دستور بدینگونه نوشته می شود:

$\max(a, \text{DIM})$

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

اگر DIM وارد نشود 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

کوچکترین درایه ماتریس \min

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.

مرتب کردن درایه ها sort

این تابع درایه های موجود 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.

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

مجموع درایه ها sum

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 و یا 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.

حاصلضرب درایه ها prod

این دستور درایه ها را در هم ضرب می کند.

prod(a,DIM)

کاربرد این دستور دقیقاً مانند sum است

>>k=[1 2 3;4 5 6];

>>prod(k,2)

Ans=

6

120

میانگین درایه ها mean

mean(a,DIM)

میانگین درایه ها را محاسبه میکند.

کاربرد دقیقاً مانند sum است.

>>mean(k,2)

Ans=

2

5

قطر اصلی diag

این دستور قطر اصلی ماتریس مربع را به صورت یک ماتریس ستونی می دهد و البته ماتریس مربع متناظر با ماتریس ستونی و یا سطربی معرفی شده را نیز می دهد .

diag(a)

a می تواند یک ماتریس مربع و یا 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 سطربی و غیر ستونی و مربع نباشد (m*n). پاسخ دستور درایه های واقع در مکان های (n*m) و یا (m*m) خواهد بود .

دترمینان ماتریس det

دترمینان 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

تربیس ماتریس trace

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 ($n \times n$) با هم جمع خواهد شد .

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

Ans=
6

مرتبه ماتریس rank

مرتبه ماتریس یعنی بزرگترین درجه ای که ماتریس مستقل باشد(سطر یا ستون مضرب و یا 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.

چرخش ماتریس در بعد flipdim

ماتریس را حول بعد تعریف شده می چرخاند.

flipdim(a,DIM) 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
6 5 4
9 8 7
```

چرخش چپ راست ماتریس flipr

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
6 5 4
9 8 7
```

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

چرخش ماتریس به صورت بالا پایین flipud

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
```

چرخش ماتریس rot90

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
```

کار با فایل

ذخیره کردن متغیر save

متغیر معرفی شده را در شاخه current directory ذخیره می کند.

```
>>save parameter  
>>
```

در مثال بالا به جای parameter نام متغیر نوشته می شود.

دستور save پاسخ ندارد.

پارامتر ذخیره شده توسط این دستور فقط با مطلب قابل خواندن می باشد.

خواندن متغیر ها load

این تابع متغیر ذخیره شده را می خواند. همچنین می تواند فایل متنی که به صورت عددی (ماتریسی) نوشته شده بخواند.

```
>>load parameter  
>>
```

که به جای parameter نام پارامتر مربوطه قرار داده می شود.

و در صورتی که بخواهیم از فایل بخوانیم:

```
>>parameter=load('name')
```

در این دستور name نام و فرمت فایل متنی است و parameter نامی است که به متغیر اختصاص میدهیم.

Parameter=

.....
....

بازکردن فایل در editor open

>> open('name') همان نام فایلی است که می خواهیم باز کنیم name

این تابع ماتریس مشخص شده را بر حسب تنظیمات مشخص شده در داخل یک فایل متّنی می‌ریزد.

```
DLMWRITE('file.txt',M,'delimiter','\t','precision','%.6f','newline','pc')
```

خط بالا شکل کلی دستور می‌باشد.

در دستور:

نام فایل خروجی می‌باشد.

M نام ماتریسی است که می‌خواهیم در داخل فایل برجیزیم.

'delimiter' نحوه جدا سازی درایه‌ها در یک ردیف را مشخص می‌کند که کلمه بعدی نیز همان را نشان میدهد.

اگر کلمه بعدی \t باشد (مانند این مثال) درایه‌ها با tab (جلو رفتن به تعداد هشت کاراکتر) ... جدا می‌شود.

و اگر کلمه بعدی حرف باشد (مانند ',') درایه‌ها با ویرگول از هم جدا می‌شوند.

'precision' دقت ارائه شده و چاپ شده در فایل که باز نوع آن را کلمه بعدی معرفی می‌کند.
'%' یعنی تا شش رقم اعشار

'newline' این قسمت طریقه تعریف سطر‌های دوم و سوم و... را در ماتریس اصلی معلوم می‌کند که در صورت تعریف نشدن همه درایه‌ها به صورت پشت سر هم نوشته خواهند شد
'pc' نیز دقیقاً با 'newline' می‌اید (در سیستم‌های بر اساس unix به جای pc، ر
نوشته می‌شود).

با این دستور می‌توانیم هر گونه فایل متّنی را بخوانیم.

دستور فوق بدینگونه مورد استفاده قرار می‌گیرد.

```
dlmread('file name','delimiter')
```

در دستور بالا به جای file name نام و فرمت فایل متّنی مربوطه و به جای delimiter کاراکتری که در فایل متّنی درایه‌ها را جدا می‌کند، را باید وارد کنیم (در صورتی که در فایل متّنی ماین اعداد هیچ کاراکتری وجود ندارد و فقط با فاصله جدا شده اند به جای delimiter نیز باید جای خالی گذاشت).

خواندن فایل متنی textread

این تابع می تواند فایل متنی را بخواند.

کابرد این دستور به صورت زیر می باشد :

```
textread('filename')  
[a,b,c,d,...]=textread('filename','format')
```

دستور اول کا ملا بدیهی است یعنی فقط باید نام فایل را وارد کنیم.

ولی در دستور دوم $a b c d$ همه نام ماتریس های ستونی است که هر کدام از ستونهای فایل متنی به ترتیب از چپ به راست در ماتریس های متناظر ریخته می شود. Format بدين گونه تعریف می شود که به ازای هر ستون یک فرمت بدينگونه نوشته می شود بدينگونه که هر کدام را با یک $d\%$ نشان می دهیم که d نوع عدد موجود در فایل متنی را نشان می دهد.

برای جا افتادن دستور بالا به مثالی ساده می پردازیم:

در شاخه current directory یک فایل متنی مینویسیم که هر سطر آن چهار عدد باشدواعداد با فقط با یک جای خالی جدا شده باشد و با یک نام راحتی ذخیره می کنیم .
 (تعداد اعداد در هر سطر باید حتما چهار باشد و گر نه سیستم پیغام خطای خواهد داد)
 برای خواندن فایل فوق نام فایل را در دستور زیر وارد کنید دستور را به کار بگیرید .
 [a,b,c,d]=textread('file name'),%d %d %d %d')

توجه می کنید که در میان هر کدام از $d\%$ فضای خالی گذاشته شده است یعنی در میان اعداد در فایل متنی هر چیزی وجود دارد بای همان را در میان $d\%$ ها نوشته می شود .
..... یعنی اگر در فایل متنی به جای فضای خالی ویرگول بگذاریم دستور بالا به صورت زیر در می آید.

```
[a,b,c,d]=textread('file name'),'%d,%d,%d,%d')
```

ترسیم دو بعدی

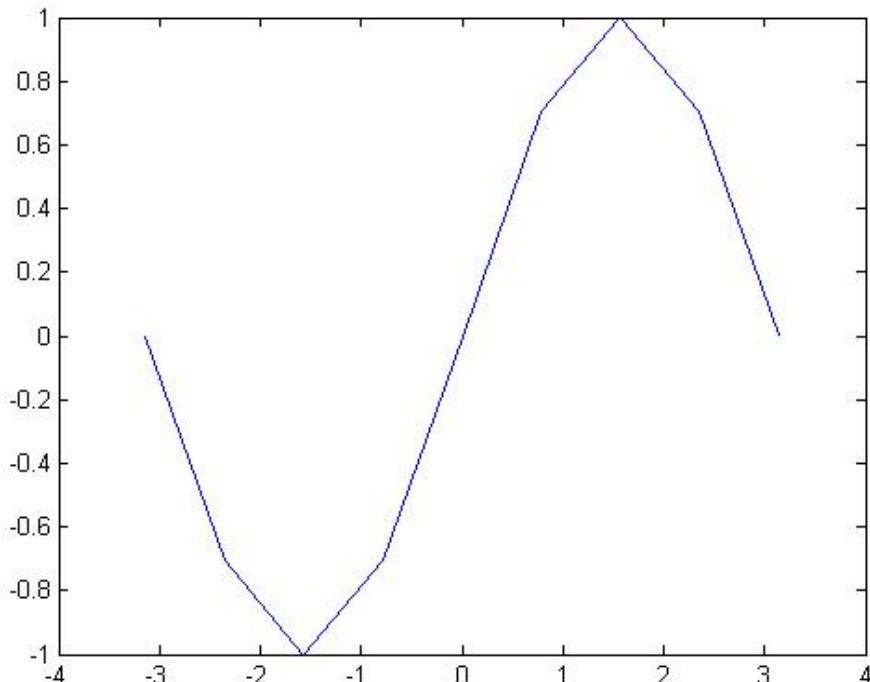
در متلب ابزار کاملی برای ترسیم نمودار های مختلف وجود دارد .
این ابزار شامل نمودار های دو بعدی و سه بعدی و انواع نمودار های فراوانی میباشد .

رسم دو بعدی plot

یکی از ابزار های رسم نمودار روابع دو بعدی این دستور می باشد که نحوه ترسیم آن دقیقاً مانند ترسیم دستی نمودار که خودمان انجام میدهیم بینگونه که در بازه خاصی x را معرفی می کنیم و y منتظر هر کدام را بدست آورده و در پایان بر اساس اعداد محاسبه شده نقاط مربوطه پیاده شده و نقاط به هم وصل می شود . طبیعتاً هر چقدر فاصله نقاط کمتر باشد دقت ترسیم بهتر می شود .

به مثال زیر توجه کنید :

```
>>x=-pi:pi/4:pi;  
>>y=sin(x);  
>>plot(x,y)
```

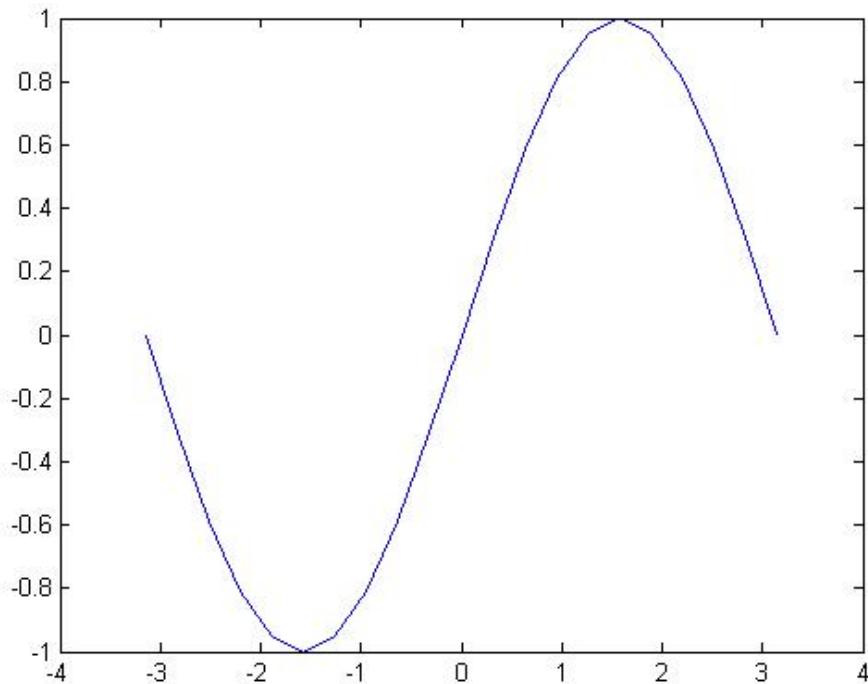


میبینید که نمودار با شکستگی های زیادی همراه است . حال دقت ترسیم را افزایش دهیم :

```

>>x=-pi:pi/10:pi;{enter}
>>y=sin(x);{enter}
>>plot(x,y)

```



در دستورات بالا در خط اول یک بازه به تعداد مشخص تقسیم می کنیم (x ها را معرفی می کنیم) پس از آن در خط دوم مقدار y را برای هر کدام از x ها پیدا می کنیم و در پایان در خط سوم نقاط پیاده شده و به هم وصل می شود .

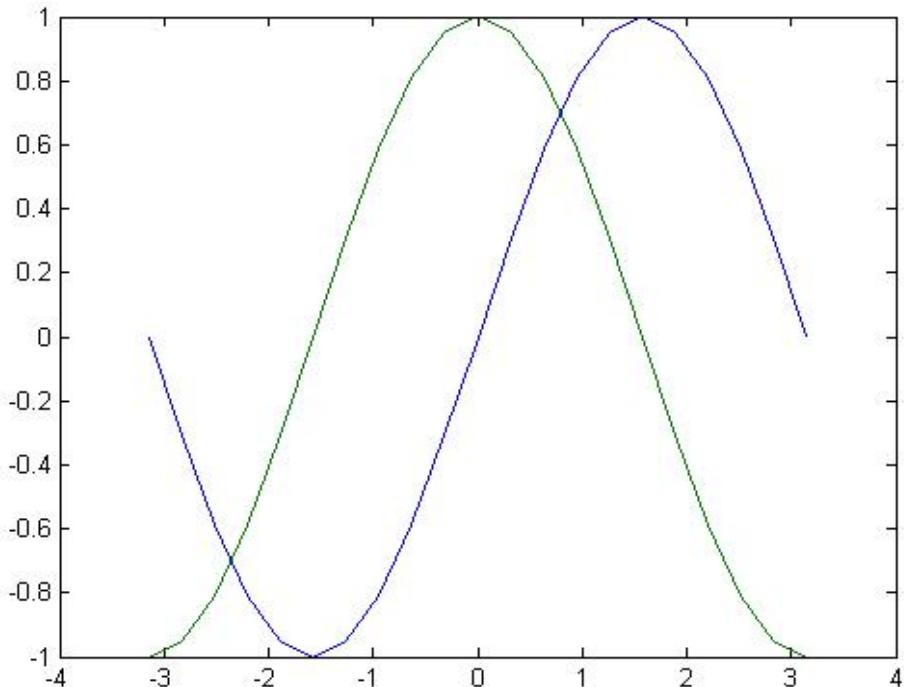
حال اگر بخواهیم دو نمودار \sin ، \cos را کنار هم رسم کنیم ، کافیست در دستور `plot` مقدار محاسبه شده برای \cos را نیز قرار دهیم .

این کار بدینگونه است ، هر ترسیمی را که افزایش میدهیم ، در دستور `plot` دقیقاً مانند دستور ترسیم یک نمودار پارامتر ها را به پشت سر دستور آن می افزاییم به گونه ای که هر جفت پارامتر برای کدام از ترسیمات پشت سر هم قرار گیرد .
باز اگر بخواهیم ترسیم دیگری اضافه کنیم مثل روش بالا اضافه می کنیم .

```

>>x=-pi:pi/10:pi;
>>y=sin(x);
>>z=cos(x);
>>plot(x,y,x,z)

```



لطفا به آخرین سطر دستورات توجه فرمایید.

حال اگر بخواهیم ترسیمات را با رنگ و یا ... خاصی انجام دهیم بدینگونه پیش می رویم.
برای معرفی رنگ در ترسیم از حروفات کلیدی استفاده می شود که داخل کوتیشن در دستور `plot` نوشته می شود.

به مثال زیر توجه کنید:

```
plot(x,y,'r')
```

این دستور ترسیم را با رنگ قرمز انجام می دهد و به نوع ترسیم کاری ندارد.

```
plot(x,y,'^')
```

این دستور به رنگ کاری ندارد و فقط نقاط را با مثلث نشان می دهد.

```
plot(x,y,:')
```

این دستور فقط ترسیم را به صورت خط چین انجام می دهد.

هر کدام از سه حرف کلیدی بالا از گروه خاصی هستند که می توان آنها را با هم ترکیب نمود .

`plot(x,y,'r^:')`

در این دستور سه حرف کلیدی بالا با هم ترکیب شده است .

حرف اول رنگ(قرمز) حرف دوم علامت(مثلث) و حرف سوم نوع خط(نقطه چین) را تعیین می کند .

حروفات ترسیم

حروفات رنگ در plot

خط صاف	-	آبی	b
نقطه چین	:	سبز	g
خط نقطه	- .	قرمز	r
خط چین	--	فیروزه ای	c
(خالی)	بدون ترسیم خط	بنفش	m
		زرد	y
		مشکی	k

حروفات نمایش نقطه

نقطه	.
دایره	o
ضربردر	x
علامت اضافه	+
ستاره	*
مربع	s
لوزی	d
مثلث (به طرف بالا)	^
مثلث (به طرف پایین)	v
مثلث (به طرف راست)	>
مثلث (به طرف چپ)	<
ستاره پنج راسی	p
ستاره شش راسی	h

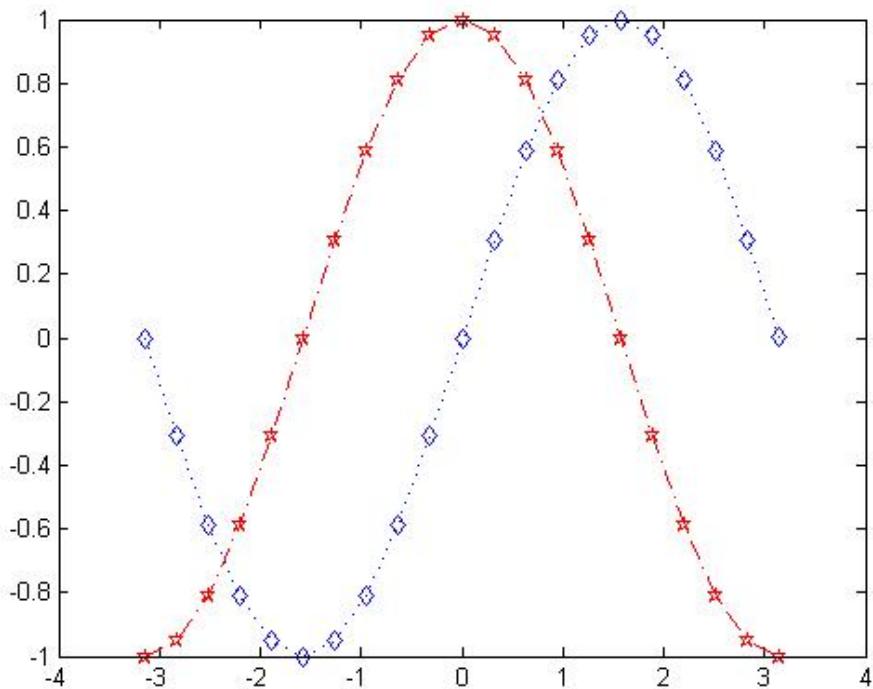
گفتیم که می توان همه این حروفات را با هم ترکیب کرد بدینگونه که ('trsیم ، نقطه ، رنگ ')

حال ترسیم بالایی را با تنظیمات جدید انجام می دهیم .

```

>>x=-pi:pi/10:pi;
>>y=sin(x);
>>z=cos(x);
>>plot(x,y,'b:d',x,z,'rp-')

```



در خط آخر فرمان مر بوط به ترسیم و تنظیمات را می بینیم که تنظیمات مربوط به هر کدام از ترسیم
بلافاصله پس از پارامتر ها می آید .

در خط آخر دستور مقابل را میبینیم :

```
>> plot(x,y,'b:d',x,z,'rp-')
```

پس از تعیین پارامتر ها (x,y) بلافاصله تنظیمات آن آمده است که 'b:d' می باشد که رنگ آن b آبی
و نوع ترسیم آن : نقطه چین و نشان نقطه آن d لوزی است ... و ترسیم دوم که پارامتر های آن (x,z)
و تنظیمات آن با رنگ r قرمز و نوع خط p نقطه و ستاره pنج راسی است .

تنظیمات صفحه رسم

در مواردی لازم است که برای نموداری که ترسیم نموده ایم نام و توضیحات خاصی ارائه کنیم .
این توضیحات ممکن است نام ترسیم نام محور ها نوشتن بر روی ترسیم و .. باشد .

چند نمونه از ترسیمات :

برچسب محور x xlabel

این دستور محور x را نامگذاری می کند .
xlabel('string')

در دستور بالا به جای string کلمه و حروفات مربوطه گذارده می شود .

بر چسب محور y ylabel

این دستور محور y را نامگذاری می کند .
ylabel('string')

در دستور بالا به جای string کلمه و حروفات مربوطه گذارده می شود .

نام ترسیم title

این دستور ترسیم را نامگذاری می کند .
title('string')

به جای string نام مربوطه قرار می گیرد .
هر نامی که می نویسیم در بالای ترسیم نشان داده می شود .

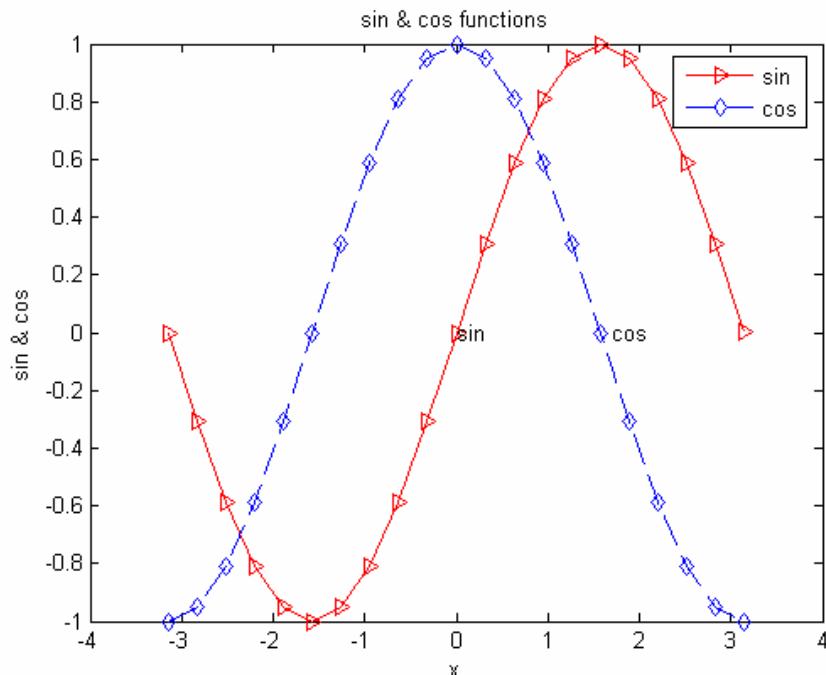
اگر چندین نمودار رسم کرده باشیم ممکن است نتوانیم تشخیص دهیم که کدام ترسیم مربوط به کدام نمودار است بوسیله دستور `legend` می توانیم بر حسب رنگ و نوع ترسیم نمودار ها را از هم تمیز دهیم.

```
legend('string 1','string 2')
```

ترتیب نوشتمنام ها بدینگونه است که در دستور `plot` هر کدام از نمودار ها ترسیم شده در اینجا نیز همانگونه عمل می شود.

به مثال زیر توجه فرمایید :

```
x=-pi:pi/10:pi
y=sin(x)
z=cos(x)
plot(x,y,'r>-',x,z,'bd--')
xlabel('x')
ylabel('sin & cos')
title('sin & cos functions')
legend('sin','cos')
text(0,0,'sin')
text(1.7,0,'cos')
```



در دستورات بالا استفاده شده که برای نوشتمنامه ای در مکان خاصی (مختصات) بکار میرود.
`text(x , y , ' string')`

چند ترسیم در یک صفحه

گفتیم که میتوان در دستور `plot` چندین ترسیم را یکجا انجام داد . ولی اگر نتوانیم بصورت یکجا ترسیم کنیم ویا نخواهیم ترسیمات بر روی هم بیافتد چه کار کنیم .

در دستور `plot` قبل از ترسیم صفحه پاک می شود (میتوان گفت دستور `clf` اجرا میشود) ولی وقتی بخواهیم بر روی ترسیم انجام شده یک نمودار دیگری رسم کنیم باید ترسیم قبلی پاک نشود .

دستور `hold` این کار را انجام می دهد .

نگه داشتن ترسیم `hold`

با استفاده از این دستور مانع پاک شدن صفحه نمایش می شویم تا نمودار های بعدی بر روی نیز بر روی نمودار اولی بیافتد (در این نمودار ها همه رنگها و نوع خط یکسان خواهند بود زیرا به طور جداگانه ترسیم می شوند و از رنگ و نوع خط اولیه استفاده می کنند .

این دستور به صورت روشن و خاموش استفاده می شود
hold on

hold off

تا زمانی که `hold` در حالت روشن است هیچ نموداری پاک نخواهد شد و همه بر روی هم خواهد افتد .

ولی زمانی است که نمی خواهیم نمودار ها بر روی هم بیا فتد .

بوسیله دستور `subplot` صفحه ترسیم را به تعداد مشخص تقسیم می کنیم .

بدینگونه که صفحه ترسیم را تقسیم کرده و قسمت مورد نظر را معلوم کرده و نمودار مربوطه را رسم میکنیم و سپس نام و تنظیمات را انجام می دهیم .

رسم چندین رسم در یک صفحه subplot

بوسیله این دستور می توانیم صفحه ترسیم را به چندین قسمت تقسیم کنیم .

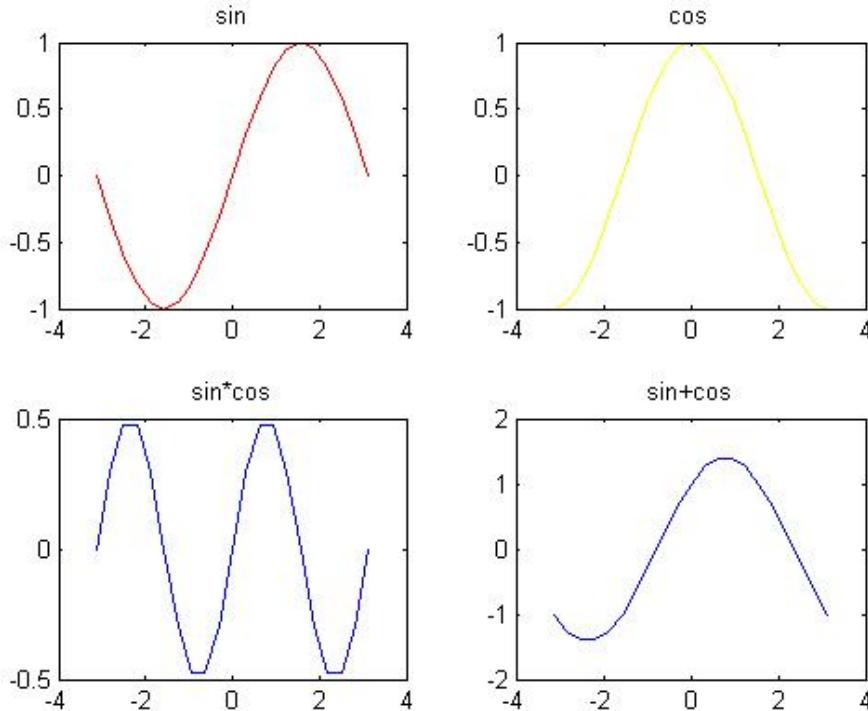
این دستور را بدین گونه استفاده میکنیم .

subplot(m,n,p)

با این دستور صفحه به m سطر و n ستون تقسیم می شود و قسمت p ام را آدرس دهی میکند .
که شما ره قسمت از ردیف اول از بالا شروع می شود .

به مثال زیر توجه کنید :

```
x=-pi:pi/10:pi;
y=sin(x);z=cos(x);t=sin(x).*cos(x);
subplot(2,2,1);plot(x,y,'r');title('sin');
subplot(2,2,2);plot(x,z,'y');title('cos');
subplot(2,2,3);plot(x,t);title('sin*cos');
subplot(2,2,4);plot(x,y+z);title('sin+cos');
```



لطفا به نحوه کابرد subplot و نحوه تنظیم ترسیمات توجه فرماید .

ترسیمات سه بعدی و سطوح

دیدیم که دستور `plot` ابزاری برای رسم نمودار های دو بعدی است ولی موقعی که می خواهیم نمودار های سه بعدی را رسم نماییم چه کار باید بکنیم .

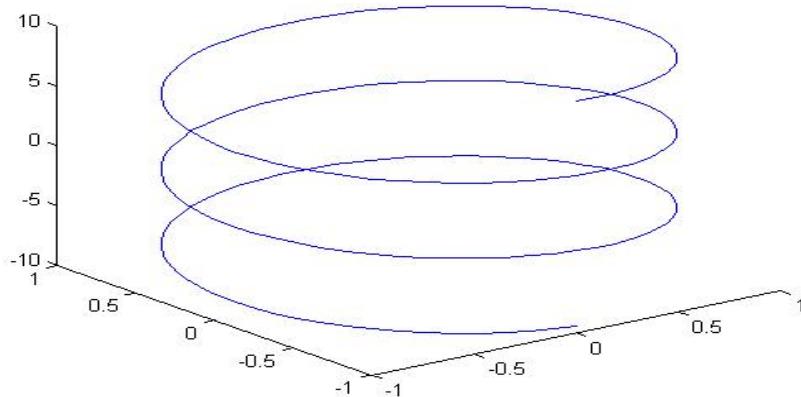
دستور `plot3` این کار را انجام می دهد .

رسم سه بعدی `plot3`

این دستور تابع را در سه بعد رسم می کند .

به این مثال توجه فرمایید :

```
>>t=-3*pi:pi/30:3*pi;  
>>x=sin(t);  
>>y=cos(t);  
>>z=t;  
>>plot3(x,y,z);
```



می بینید که نحوه ترسیم مانند دستور `plot` می باشد .

البته میتوانیم بوسیله دستور `plot3` ، سطوح ولایه ها را ترسیم کنیم .

رسم سطح و لایه

رسم لایه و سطح بانمودار یکسان نمی باشد ، نمودار یک منحنی و ... می باشد(در کل یک خط) ولی سطح اینگونه نمی باشد و یک فضای پیوسته (لایه) می باشد .

خودمان می توانیم هر سطحی که دلمان میخواهد بسازیم ولی برای راحتی کار ، یک تابع سطح ، در خود مطلب قرار داده شده است که با دستور peaks می توان به این سطح دست یافت .

سطح نمونه peaks

گفتیم که یک سطح پیش فرضی در مطلب قرار داده شده که با این دستور میتوان به این سطح دست یافت .

[x,y,z]=peaks(n)

n دقت ترسیم را نشان میدهد (بازه مربوطه به عدد وارد شده تقسیم می شود و چقدر n بیشتر باشد قطعات کوچکتر و همینطور شکستگی نرم تر و رسم دقیقتر خواهد بود)

در صورتی که می خواهید یک سطح بسازید ، باید فاصله همه نقاط موجود بر روی سطح یکسان و به طور مساوی پخش شده باشد و البته z y x را نیز بر حسب توابع بیان میکنیم .

برای ساختن سطح باید شبکه ای کامل و همگن ساخته شود (منظور از شبکه خطوط عمود بر هم است که محل تقاطع مکان نقاط سطح را نشان می دهد) .
در مطلب شبکه را با کمک دستور meshgrid می سازیم .

تشکیل شبکه meshgrid

این دستور یک شبکه برای ایجاد سطوح می سازد .

[x,y]=meshgrid(x) -> = [x,y]=meshgrid(x,x)
[x,y,z]=meshgrid(x,y,z)

اگر یک مثال کوچک برای این دستور بزنیم :

```
>>[x,y]=meshgrid(-2:2)
```

X=

```
-2 -1 0 1 2
-2 -1 0 1 2
-2 -1 0 1 2
-2 -1 0 1 2
-2 -1 0 1 2
```

Y=

```
-2 -2 -2 -2 -2
-1 -1 -1 -1 -1
0 0 0 0 0
1 1 1 1 1
2 2 2 2 2
```

میبینید که این دستور ماتریس 2×2 میسازد – واگر بیشتر دقیق کنید می بینید که هر کدام از سطر y

و یا ستون x می تواند معرف یک خط باشد (یک سری خط موازی عمودی و افقی) که با تعریف z

می توان به سطح دست یافت .

```
>>z=x.^2+y.^2
```

Z=

```
8 5 4 5 8
5 2 1 2 5
4 1 0 1 4
5 2 1 2 5
8 5 4 5 8
```

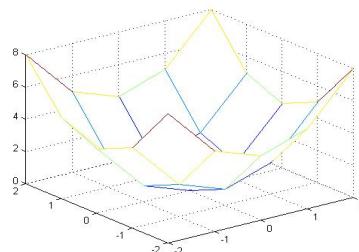
که z را بر اساس مقادیر x و y بدست می آوریم ((می دانیم که x^2 یک تابع دو بعدی کاسه ای

است و y^2 نیز همچنین ووقتی این دو در جهات x و y با هم ترکیب شوند یک کاسه سه بعدی ساخته

خواهد شد)) .

با کمک دستور mesh سطح مربوطه را رسم میکنیم .

```
>>mesh(x,y,z)
```



می بینید که دقیق ترسیم خیلی پایین است و آن هم به این دلیل است که تعداد تقسیمات کم بوده است .

رسم شبکه تشکیل شده

mesh

این دستور شبکه تشکیل شده را ترسیم می کند .

سطح نمایش داده شده توسط این دستور صرفا یک شبکه است .

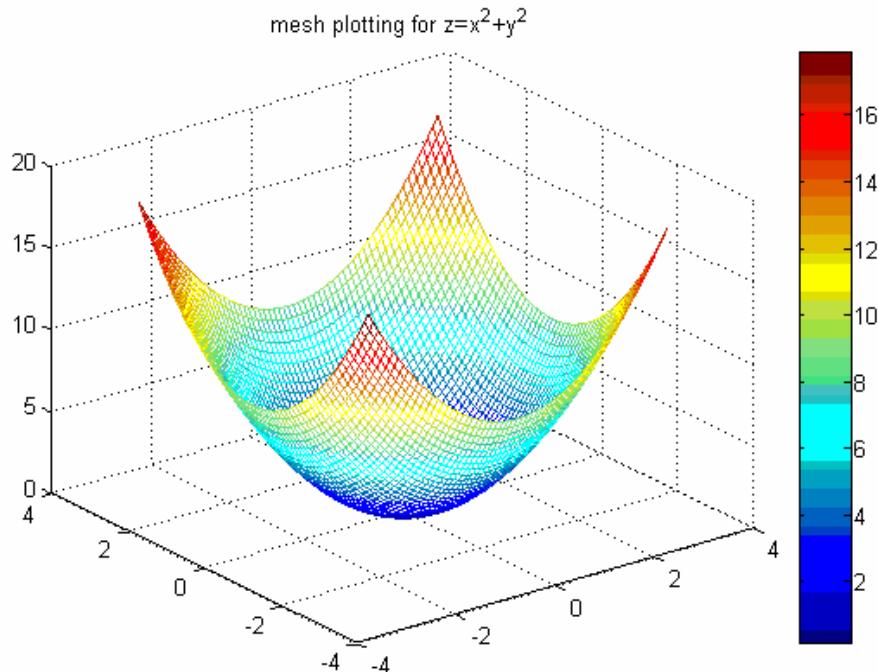
این دستور بدینگونه نوشته می شود:

mesh(x,y,z)

که x, y, z با meshgrid تعیین شده اند .

مثال این دستور در صفحه قبل ذکر شده است . که رسم دقیق آن به صورت زیر است .

```
>>[x,y]=meshgrid(-3:.1:3)
>>z=x.^2+y.^2
>>mesh(x,y,z)
>>colorbar
>>title('mesh plotting for z=x^2+y^2')
```

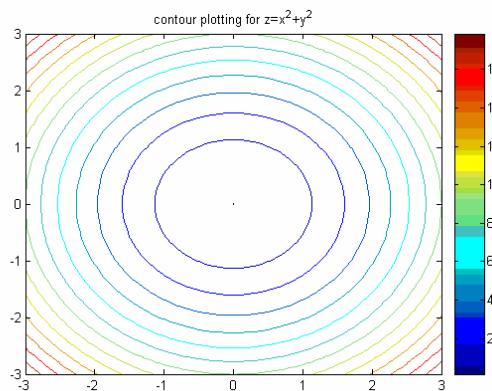


در این مثال می بینیم در خط چهارم دستور colorbar استفاده شده که این دستور باعث ایجاد یک میله رنگی معرف ارتفاع می باشد و سطر آخر عنوان رسم را تشکیل می دهد .

منحنی میزان منحنی است که همه نقاط هم ارتفاع زمین را به هم وصل میکند .

منحنی های میزان همدیگر را قطع نمی کند و کوچکترین محیط بسته ، بلند ترین نقطه است و یا پایین ترین نقطه .

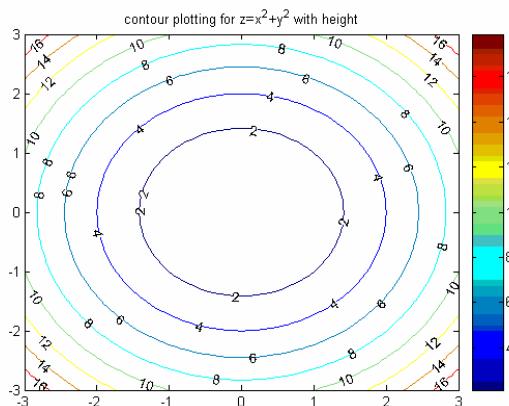
```
>>[x,y]=meshgrid(-3:.1:3)
>>z=x.^2+y.^2;
>>contour(x,y,z,15);
>>colorbar
>>title('contour plotting for z=x^{\{2\}}+y^{\{2\}}')
```



عدد 15 در دستور contour دقت ترسیم را معین می کند .

می بینید که کوچکترین دایره با رنگ آبی یا همان رنگ ارتفاع 2 و یک نقطه نیز در مرکز به رنگ مشکی که ارتفاع 0 می باشد که پایینترین نقطه سطح می باشد .
که البته میتوان ارتفاع را بر روی منحنی میزان نوشت .

```
>>contour(x,y,z,'showtext','on')
```

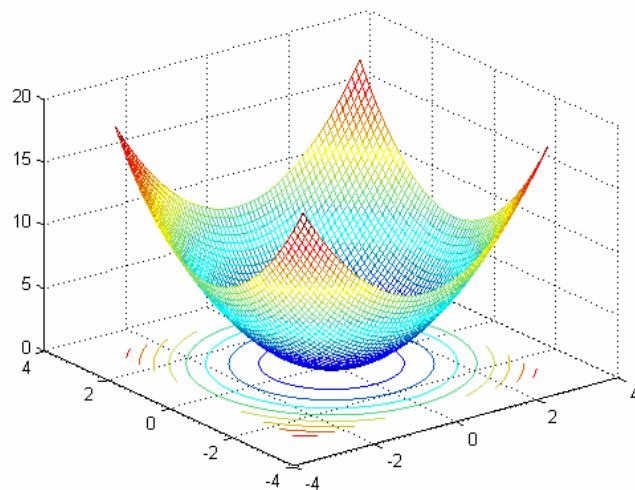


ترسیم شبکه و منحنی میزان

meshc

این دستور mesh و منحنی میزان را در یکجا رسم می کند . به طوریکه mesh رسم می شود و منحنی میزان زیر آن ترسیم می گردد .

>>meshc(x,y,z)



در این مثال می بینیم که mesh رسم شده و منحنی میزان زیر آن قرار دارد .

[در مثال هایی که تعریف پارامتر صورت نگرفته است از پارامتر مثال های قبلی استفاده کنید.]

نمایش سطح بوسیله رنگ و نور

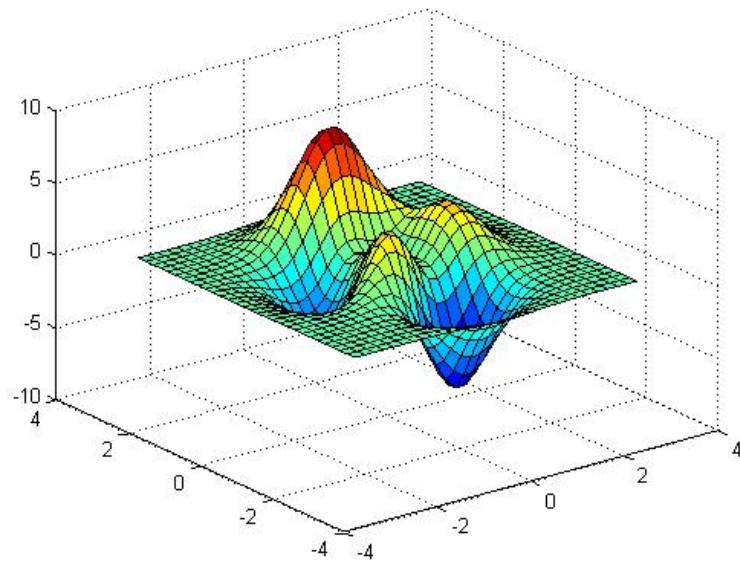
surf

این دستور سطح را با رنگ و نور نمایش میدهد به کونه ای که می توان بوسیله نور پردازی به اشیا عمق داد .

به مثال صفحه بعد توجه کنید :

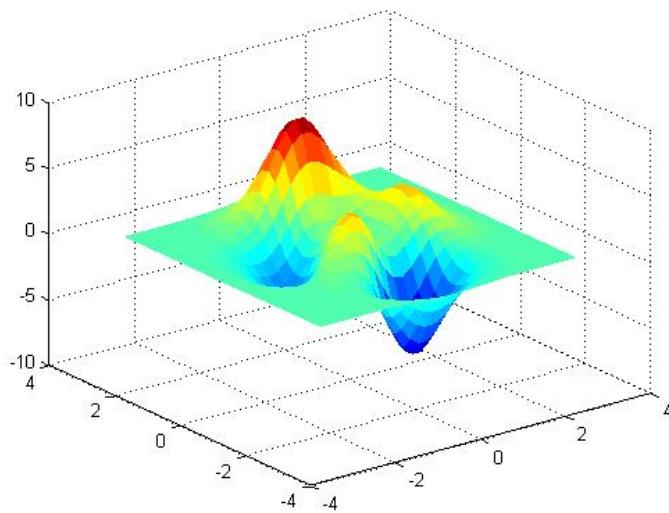
در این مثال برای تشکیل سطح از دستور peaks استفاده شده است .

```
>>[x,y,z]=peaks(30);  
>>surf(x,y,z)
```



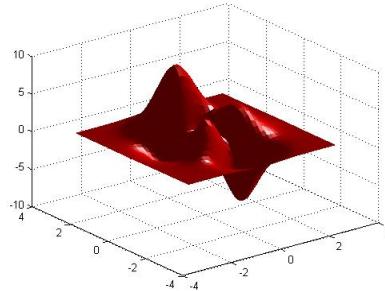
اگر بخواهیم خطوط دیده نشوند باید ویژگی edgecolor را خاموش کنیم .

```
>>surf(x,y,z,'edgecolor','none')
```



همینطور می توانیم دیگر ویژگی ها را نیز تغییر دهیم که دقیقاً مثل دستور قبل پیش می رویم .

```
>> surf(x,y,z,'facecolor','red','edgecolor','none');light
```

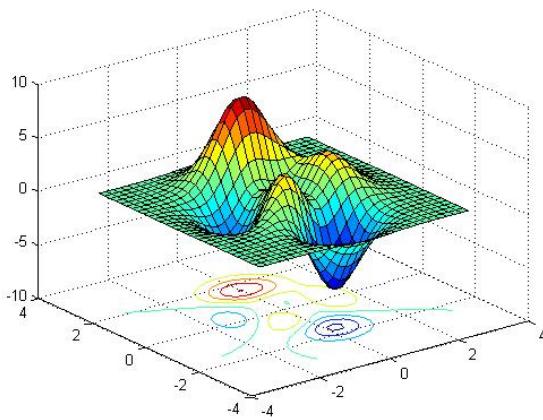


دستور light به سطح یک نوردهی مایل اضافه می کند تا رنگ سطح غیر یکسان و ناهموار دیده شود .

surf سطح با منحنی میزان

این دستور مانند mesh است با این تفاوت که سطح را با منحنی میزان رسم می کند .

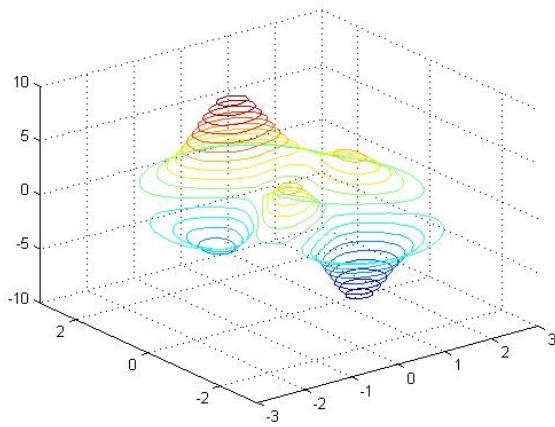
```
>>surf(x,y,z)
```



contour3 منحنی میزان سه بعدی

این دستور منحنی میزان سه بعدی رسم میکند . بطوریکه هر منحنی میزان در ارتفاع مربوط به خود نمایش داده می شود .

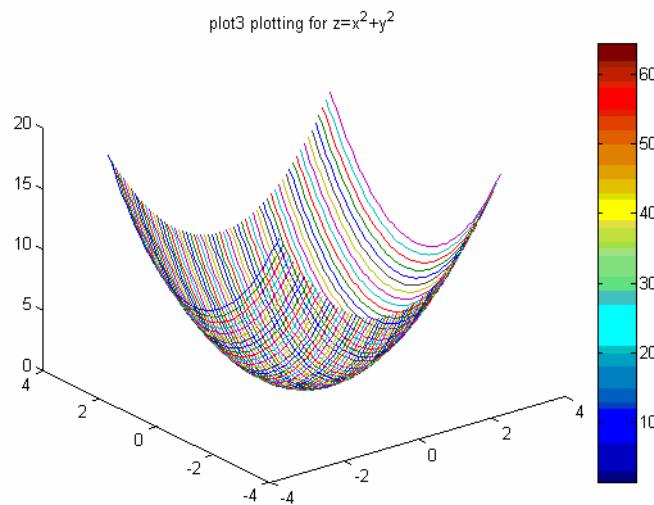
```
>>[x,y,z]=peaks(40);  
>>contour3(x,y,z,20)
```



رسم سه بعدی plot3

این دستور مانند رسم دو بعدی است با این تفاوت که با کنار هم گذاشتن خطوط سطح را نشان می دهد .

```
[x,y]=meshgrid(-3:.1:3)
z=x.^2+y.^2
plot3(x,y,z)
colorbar
title('plot3 plotting for z=x^2+y^2')
```



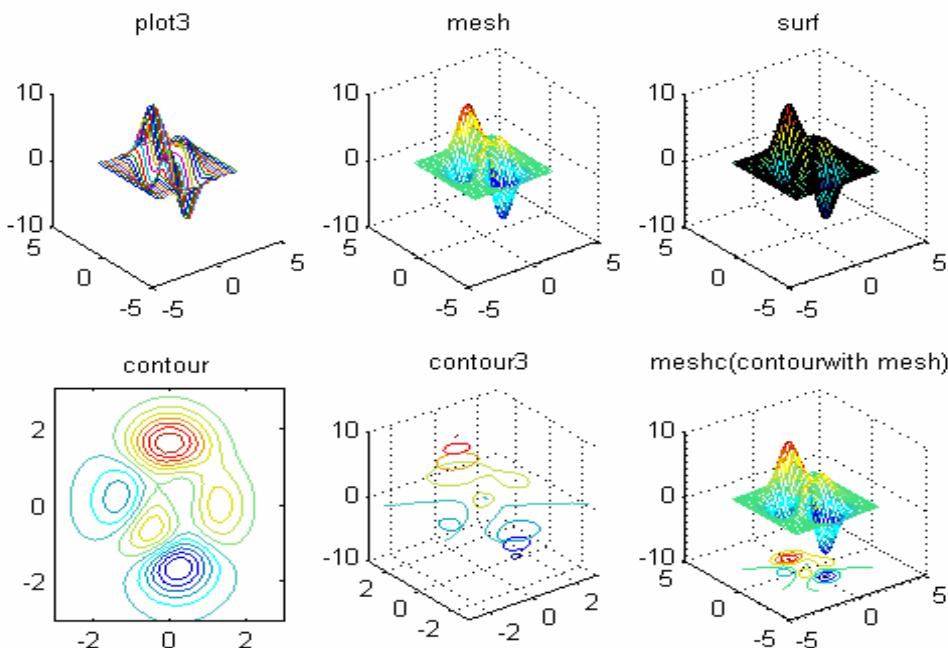
می بینید که سطح با قرار گرفتن خطوط کنار هم نمایش داده می شود .

>>[x,y,z]=peaks(30);
 مکان دوربین را برای نمایش ترسیمات سه بعدی را تعیین میکند.
 >>mesh(x,y,z);
 که می توانید هر موقعیتی را برای دوربین وارد نمایید.
 >>view([4,4,4]);

یک مقایسه :

در این مثال یک سطح را با چندین دستور رسم میکنیم تا خودتان آنها را با هم مقایسه کنید.

```
[x,y,z]=peaks(30)
subplot(2,3,1);plot3(x,y,z);title('plot3')
subplot(2,3,2);mesh(x,y,z);title('mesh')
subplot(2,3,3);surf(x,y,z);title('surf')
subplot(2,3,4);contour(x,y,z,15);title('contour')
subplot(2,3,5);contour3(x,y,z);title('contour3')
subplot(2,3,6);meshc(x,y,z);title('meshc(contourwith mesh)')
```



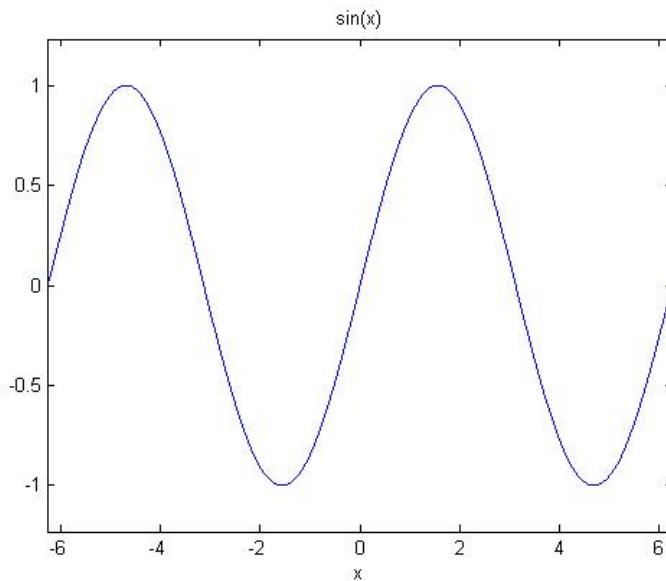
ترسیم توابع

ترسیم توابع که در مطلب به easy plotting معروف است فقط با معرفی تابع نمودار آن را رسم می کند .
در اینجا به چند نمونه از دستورات اشاره می شود .

رسم تابع دو بعدی ezplot

به مثال زیر توجه کنید :

```
>>ezplot('sin(x)')
```



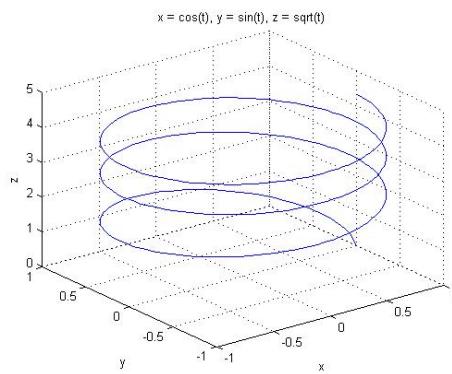
می بینید که برای رسم احتیاجی به تعریف بازه و یا متغیر نیست .

رسم سه بعدی ezplot3

در این دستور باید هر سه مولفه را تعریف نمود (البته باز بر اساس تابع) .

به این مثال توجه فرمایید :

```
>>ezplot3('cos(t)', 'sin(t)', 'sqrt(t)', [0,6*pi])
```

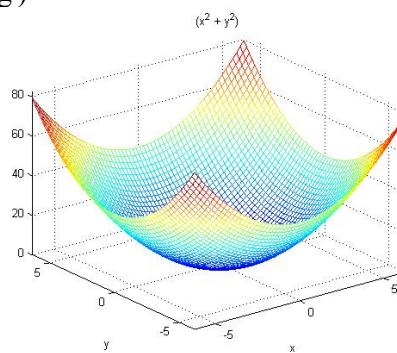


در این دستور سه مولفه x , y , z بر اساس تابع تعریف می شوند و در پایان مقدار پیش فرض برای محاسبه (مقدار پیش روی رسم) معرفی می شود که اگر این مقدار را معرفی نکنیم سیستم خود عددی پیش فرض قرار خواهد داد .

رسم شبکه برای تابع ezmesh

این دستور یک شبکه برای یک تابع سه بعدی تعریف شده می سازد .

```
>>ezmesh(' (x^2 + y^2 )'  
>>colorbar  
>>title('easy mesh plotting')
```

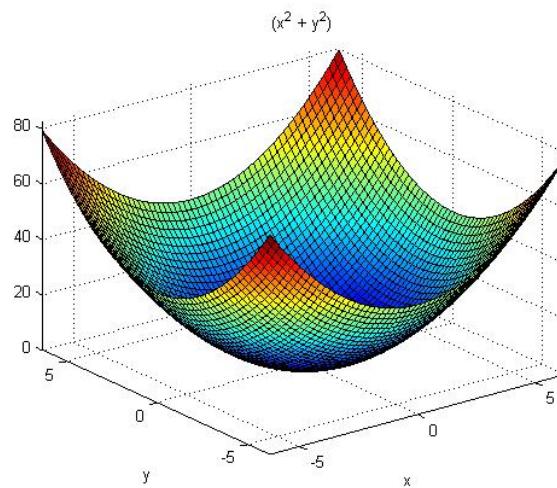


ترسیم سطح برای توابع

ezsurf

دقیقاً مانند دستورات بالا ترسیم سطح را بصورت رنگ و سایه برای سطح را انجام می‌دهد.

```
>>ezsurf('(x^2 + y^2)')  
>>colorbar  
>>title('easy surf plotting')
```

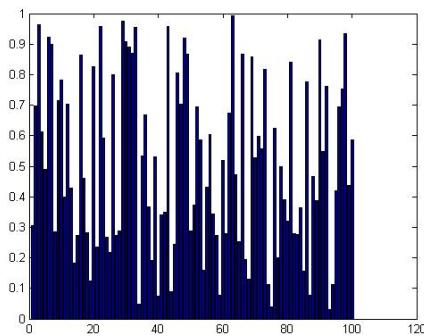


نمودار های آماری

نمودار میله ای Bar

این دستور نمودار میله ای یک مجموعه را رسم می کند .

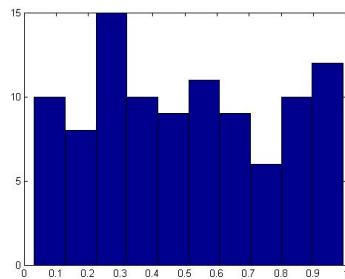
```
>>a=rand(1,100);  
>>bar(a)
```



نمودار فراوانی Hist

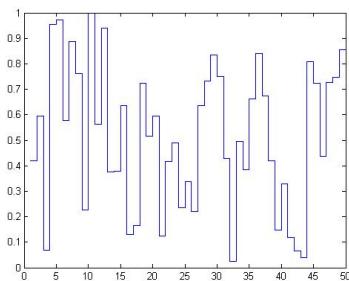
نمودار هیستوگرام مربوط به مجموعه را رسم میکند .

```
>>a=rand(1,100);  
>>hist(a)
```



نمودار پله ای stairs

```
>>a=rand(50,1);  
>>stairs(a)
```



چندجمله ای ها

در مطلب برای محاسبات چند جمله ای جعبه ابزاری تقریبا کامل قرار داده شده است . در اینجا چند نمونه از دستورات و کاربرد آنها را می بینیم .

همه می دانیم چندجمله ای یا polynomial توابعی یک متغیره هستند . ضرایب چند جمله ای از بالا به پایین در یک ماتریس مرتب میشود که درایه اول بالاترین توان و درایه آخر توان صفر یا همان عدد ثابت است . به این مثال توجه کنید :

ماتریس مربوط به چند جمله ای $3x^3 + 2x^2 + 2x + 1$ بدین گونه می شود :

[1 2 3]

می بینید که فقط ضرایب نوشته شده اند .

حال اگر ضرایب چند جمله ای $3x^3 + 4x^2 + 3x + 1$ را بنویسیم به این صورت در خواهد آمد :

[3 0 4 1]

این مثال میبینید که بدلیل نبودن x^2 در چند جمله ای ، در ماتریس مربوطه درایه دوم ک همان ضریب x^2 میباشد را صفر قرار می دهیم .

از این به بعد چند جمله ای را با ماتریس مربوطه نشان می دهیم و در محاسبات چند جمله ای نیز از ماتریس ها استفاده میکنیم .

ریشه های چند جمله ای roots

پاسخ این دستور ریشه های چند جمله ای وارد شده می باشد که ممکن است مختلط نیز باشد .

```
>>a=[1 2 3];  
>>roots(a)
```

```
Ans=  
-1.0000+1.4142i  
-1.0000-1.4142i
```

می بینید که دو ریشه مختلط دارد .

چند جمله ای از روی ریشه poly

این دستور چند جمله ای متناظر با ریشه های وارد شده را می سازد .

برای مثال ریشه های بدست آمده برای مثال قبل را وارد می کنیم :

```
>>r=[-1+1.4142i,-1-1.4142i];  
>>poly(r)
```

Ans=

1.0000 2.0000 3.0000

مقدار گذاری در چند جمله ای polyval

بوسیله این دستور میتوانیم چند جمله ای را مقدار گذاری کنیم .

```
>>a=[1 2 3];  
>>polyval(a,2)
```

Ans=

11

معادله x^2+2x+3 به ازای $x=2$ محاسبه شده است .

لازم به ذکر است که polyvalm نیز معادله را به ازای ماتریس محاسبه می کند .

بهترین برازش منحنی polyfit

نقاط وارد شده را با بهترین حالت بر روی منحنی (با درجه معلوم) برازش میدهد .

این دستور بدینگونه استفاده می شود :

polyfit(x,y,n)

در این دستور x, y مختصات نقاط وارد شده و n درجه چند جمله ای برازش داده شده است .

برای این مثالی را میخواهیم ارائه کنیم که نخست لازم است دستور ginput را فرا بگیریم .

گرفتن مختصات از موس

ginput

با اجرای این دستور سیستم منتظر کلیک می‌ماند (فقط در فضای ترسیم) و وقتی کاربر کلیک می‌کند سیستم مختصات محل مکان نما را به صورت y, x نگه می‌دارد .

این دستور به این روش بکار می‌رود :

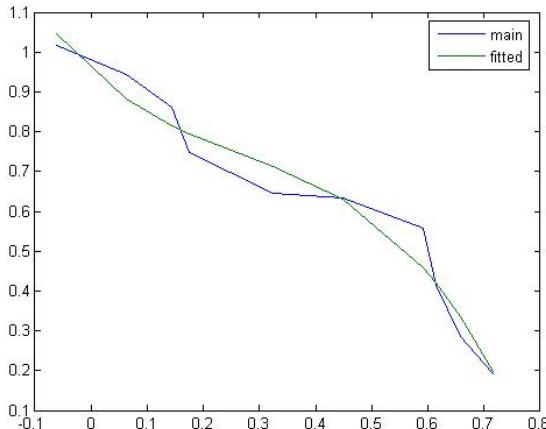
ginput(n)

n در اینجا تعداد نقاطی است که می‌خواهیم بگیریم .

برگرداییم به polyfit و ادامه مثال :

می‌خواهیم ده نقطه در یک امتداد مشخص کنیم و سپس یک منحنی درجه 3 به آن برازش دهیم .

```
>>[x,y]=ginput(10);
>>f=polyfit(x,y,3);
>>z=polyval(f,x);
>>plot(x,y,x,z);
>legend('main','fitted');
```



در صورتی که شما این مثال را اجرا کنید به هیچ عنوان به این شکل نمی‌رسید به این دلیل که در موقع انتخاب نقاط ، نقاط دیگری انتخاب خواهید کرد .

در دستورات بالا سطر اول و دوم مربوط به دریافت نقاط و برازش منحنی است که توضیح داد شد .

پاسخ برازش یک ماتریس خواهد بود که ضریب چند جمله‌ای معادله fit شده است .

در سطر سوم مقدار معادله برازش شده را به ازای x محاسبه می‌کند .

سطر چهارم و پنجم نیز ترسیم و نمایش دو نمودار را انجام میدهند .

مشتق چند جمله ای polyder

این دستور مشتق چند جمله ای وارد شده را محاسبه می کند .

```
>>a=[3 4 2 4];  
>>polyder(a)
```

Ans=

9	8	2
---	---	---

انتگرال چند جمله ای polyint

پاسخ این دستور انتگرال چند جمله ای براساس ثابت وارد شده است .

```
>>a=[1 2 3];  
>>polyint(a)
```

Ans=

0.3333	1.0000	3.0000	0
--------	--------	--------	---

حال اگر بخواهیم ثابت 3 را نیز وارد کنیم :

```
>>polyint(a,3)
```

Ans=

0.3333	1.0000	3.0000	3.0000
--------	--------	--------	--------

ضرب چند جمله ای ponv

بوسیله این دستور می توانیم دو چند جمله ای را ضرب کنیم .

```
>>a=[1 2 3];  
>>b=[1 2];  
>>c=conv(a,b)
```

c=

1	4	7	6
---	---	---	---

تقسیم نیز دقیقا مانند ضرب است . deconv

```
>>deconv(c,a)
```

که در پاسخ این دستور معادله b نمای داده خواهد شد .

تابع جمع و تفریق چند جمله ای را خودتان میتوانید بنویسید .

توابع سمبیلیک

بعضی مواقع ممکن است تابع مورد نظر چند متغیر (سمبل) داشته باشد بدین دلیل نمی توانیم از روش‌های چند جمله‌ای برای محاسبات استفاده کنیم .
این مشکل بوسیله توابع سمبیلیک قابل حل است .

شاید نام تابع سمبیلیک کمی برایتان نا آشنا باشد ولی باید بگوییم که همان تابع چند متغیره معمولی است

مثالا : $y=2*x+3*z$ و یا $y=\sin(x)$

که مطلب این توابع را به صورت پارامتری می شناسد.

برای تعریف یک معادله اول باید پارامتر های آن مشخص شود .

تعریف پارامتر syms

بوسیله این دستور پارامتر های مورد استفاده در توابع را تعریف می کنیم .

```
>>syms x y z  
>>
```

همانطور که می بینید این دستور هیچ پاسخی ندارد .

```
>> sym y=x^2
```

البته میتوانیم یک معادله را با sym وارد کنیم .

حال می توانیم تابع مورد نظر را تعریف کنیم :

```
>>y=x^2+sin(z)
```

```
Y=  
x^2+sin(z)
```

تابع تعریف شد .

مقدار گذاری تابع eval

این دستور مقدار گذاری تابع سمبیلیک را انجام می دهد . به گونه ای که اول مقادیر را تعریف می کنیم

و بعد دستور را به کار می بریم .

(تابع از مثال قبل ساخته شده است)

```
>>x=2;z=pi/2;eval(y)
```

Ans=
10

حد limit

این دستور حد معادله را در همسایگی وارد شده محاسبه می کند.

```
>>syms x y  
>>y=x^2;  
>>limit(y,2)
```

حد y در نقطه $x=2$ را محاسبه می کند.

Ans=
4

مشتق diff

مشتق معادله را محاسبه می کند.

```
>>diff(y)
```

Ans=
2*x

البته مشتق مرتبه n ام به این روش محاسبه می شود.

```
diff(y,n)
```

و اگر معادله چند متغیره باشد و بخواهیم بر اساس یکی مشتق بگیریم ان را داخل کوتیشن بعد از نام تابع می نویسیم.

```
diff(y,'x')  
diff(y,n,'x')
```

انتگرال int

انتگرال تابع وارد شده را محاسبه می کند.

نحوه کاربرد و استفاده از دستور مانند diff است.

```
int(y,'x')  
iInt(y,n,'x')  
int(y,'x',a,b)
```

برای محاسبه انتگرال معین ابتدا و انتهای بازه را پس از این جملات می نویسیم.

ترکیب تابع compose

بوسیله این تابع می توان دو تابع را ترکیب کرد .
همان $f \circ g(x)$ یا همان $f(g(x))$ را محاسبه می کند .

```
>>syms x y z f g  
>>y=x^2;  
>>f=2*g+5;  
>>compose(y,f)
```

Ans=
 $(2*g+5)^2$

مجموع یک سری symsum

مجموع سری (تابع) وارد شده را محاسبه می کند .

symsum(y)

مجموع سری به صورت تابع .

symsum(y,a,b)

مجموع سری از a تا b

معکوس تابع finverse

معکوس تابع وارد شده را محاسبه می کند .

```
>>sym y=sin(x)  
>>finverse(y)
```

Ans=
 $\text{Asin}(x)$

ماتریس ژاکوبین، ماتریسی است که هر سطر مربوط به یک تابع است و هر ستون در هر سطر مربوط به مشتق تابع مربوط به سطر نسبت به پارامتر مربوط به ستون است.
شاید توضیح ارائه شده کمی گنگ باشد. به مثال زیر توجه فرمائید تا درک موضوع راحت‌تر باشد.

```
>>syms a b c d x y z
>>y=a+2*b+4*d;
>>x=b+3*d+c;
>>z=a+d;
>>jacobian([x;y;z],[a,b,c,d])
```

```
Ans=
[0 1 1 3]
[1 2 0 4]
[1 0 0 1]
```

در دستورات:

خطوط اول تا چهارم صرفا تعریف تابع هستند.

خط پنجم دستور jacobian است که در ادامه توضیح می‌دهیم.

همانطور که می‌بینید در قسمت اول دستور $((x,y,z))$ نوشته شده و اگر دقت کنید می‌بینید که در وسط پارامترها؛ گذاشته شده که این عملگر معرف جدا کننده سطر می‌باشد. این قسمت یعنی معادلات x y z و هر کدام در هر سطر.

پس از آن قسمت دوم که $((a,b,c,d))$ یعنی مشتق‌گیری بر اساس این پارامترها انجام شود که این پارامترها نیز با، جدا شده اند که معرف جدا شدن ستون در یک سطر می‌باشد.

با این توضیحات باید جواب دریافت شده سه سطر(برای هر معادله یک سطر) و چهار ستون(برای هر پارامتر یک ستون) داشته باشیم.

برای ساختن این ماتریس به ترتیب عمل می‌کنیم که از معادلات (در سطر مربوطه) بر اساس پارامترها باز به ترتیب مشتق می‌گیریم و در مکان متناظر هر پارامتر قرار می‌دهیم.

به مثال ارائه شده در بالا رجوع کنید و شیوه ساختن ماتریس را مرور کنید.

بیشتر کتابها و جزوایت توابعی برای جمع. تفریق. ضرب و ... از قبیل symdd – symsub – symmul – sympow – symdiv چند تابع دیگر را برای کار با توابع ارائه کرده اند که بیشتر اوقات مطلب خود با استفاده از عملگرها پاسخ می‌دهد و نیازی به این توابع نیست.

REFERENCE

1.matlab reference books

Refbook.pdf

Refbook1.pdf

Refbook2.pdf

2. getting started with matlab

3. programming with matlab

4.matlab toolbox quick reference

5 . کتاب راهنمای جامع matlab 6 غیوری پایتخت

6 . کتاب 7 فکور یکتا جهاد دانشگاهی

7. کتاب آموزش matlab 6.1 دیباگران

نکاتی در مورد انتخاب و تهیه کتاب

در اینجا نمی خواهیم در مورد کتاب و کتاب خوانی صحبت کنیم .

ولی دیده شده بعضی افراد با هدف شروع به یادگیری مثلا مطلب رفته و یک کتاب 1500 صفحه ای گرفته اند و در روز دوم کتاب را کنار گذاشته و کلا دیگر سراغ کتاب و حتی نرم افزار نمی روند .

این اتفاق ممکن است برای هر کسی اتفاق بیافتد که البته دلیل خیلی ساده ای دارد .

وقتی کسی می خواهد نرم افزاری را شروع کند ، هیچ اطلاعاتی در مورد نرم افزار ندارد و چگونگی استفاده از نرم افزار را نمیداند پس طبیعتاً این شخص باید یک کتاب آموزشی تهیه کند (آموزش و یا گام به گام) و کسانی هستند که نرم افزاری را میدانند ولی می خواهند بیشتر فرا گرفته و دانش خود را گسترش دهند باید به دنبال یک مرجع باشند .

فرض کنیم کسی نرم افزاری را در حد معمولی کار کرده باشد و با هدف افزایش توانمندی و دانش یک کتاب گام به گام بگیرد روز اول کتاب را کنار خواهد گذاشت به این دلیل که همه چیز هایی که در کتاب می خواند از قبل می دانسته . ویا بر عکس ، کسی برای آموزش یک مرجع تهیه کند .

مطلوب بالا را همه قبول داریم و همه می دانیم که باید برای هر هدفی و هر نیازی کتابی مجزا را تهیه کرد .

از خود شما یک سؤال می پرسم اگر مبتدی هستید چه کتابی برای آموزش تهیه کرده اید همین لحظه جلد کتاب خود را نگاه کنید اگر آموزش و یا گام به گام باشد که کتابتان را درست انتخاب کرده اید .

ولی متأسفانه بیشتر افراد اینگونه نیستند البته بیشتر اوقات خود فرد نیز مقصو نیست برای مثال ترم اول برنامه نویسی ۰ داشتیم که استاد به ما گفت که همه بروند و کتاب ۵ مهندس نژاد قمی را بگیرند . ما هم رفتم و گرفتیم ولی دریغ از یک کلمه به این دلیل که هیچ چیز نمی فهمیدیم ... آن کتاب مرجع کامل ۰ بود و مرجعی واقعاً قدرتمندی است و در همه دانشگاه ها تدریس می شود ولی کسی که اصلاً الفبای برنامه نویسی را نمی داند به هیچ عنوان نمی تواند از آن استفاده کند .

شاید خودتان هم اشتباه بالا را مرتکب شده باشید .

پس در انتخاب کتابtan دقت کنید که برای چه هدفی تهیه می کنید .

و اما در مورد کتابهای متلب :

((این مطالب به هیچ عنوان با غرض و یا سوء نیت عنوان نشده اند و فقط برای راهنمایی افرادی است که در پی تهیه کتاب هستند بدین دلیل که دیده شده افرادی کتاب گرفته اند ولی به درد آنها نخورده است و نتوانسته اند تعویض نمایند و فقط یک ضرر مالی برایشان بوده .))

تقریباً همه کتابها ((متلب)) مرجع ثابتی دارند .

ولی باید بدانید که با چه هدفی می خواهید کتاب تهیه کنید .

اگر برای آموزش می خواهید (مبتدی هستید) کتاب matlab 6,1 انتشارات دیباگران با جلد آبی و سفید ... انتخاب خوبی می تواند برای شما باشد و البته قیمت مناسبی نیز دارد و در حدود 2000 تومان می باشد ((کتاب آموزشی تقریباً کامل برای افراد مبتدی با تمام توابع ریاضی که سر و کار دارند)) .

این کتاب ترجمه مقاله getting started with matlab (خلاصه) است که البته قسمت GUI و همینطور محاسبات ریاضی نیز به آن اضافه شده است . ((ما این کتاب را برای مبتدیان توصیه می کنیم))

و البته کتاب 7 matlab فکور یکتا انتشارات جهاد دانشگاهی جلد مشکی نیز می تواند گزینه ای مناسب باشد .

این کتاب از دو بخش کلی تشکیل شده که بخش اول آموزش متلب است و البته ترجمه همان مقاله است (البته خیلی خلاصه) و قسمت دوم نیز ترجمه و خلاصه این بخش دستورات و توابع مفید متلب را لیست نموده نحوه استفاده و ... برای هر دستور را توضیح داده است . که واقعاً مجموعه مفیدی را تشکیل می دهند .

این کتاب حدود 5000 تومان است که این را برای افرادی که آشنایی با متلب دارند را به شدت توصیه می کنیم ((کتابی است که بیشتر توابع و کاربرد و روش استفاده آن را گردآوری کرده است)) .

کتابهای دیگر واقعاً کتابهای خوبی هستند ولی هر کدام برای هدف خاصی گردآوری شده که می توانند از آنها بر اساس نیازتان استفاده کنید .

ولی افرادی که حر فه ای هستند :

راحتtan کنیم که هیچ مرجع فارسی مناسب برایتان موجود نیست . البته مرجع های چاپ شده موثر خواهند بود ولی هیچ کدام از ابزار های بدرد بخور و کار بردن را توضیح نداده اند و یا کم توضیح داده شده است .

برای شما که زیاد زبان انگلیسی تان خوب نیست همان کتاب 7 matlab فکور یکتا جهاد دانشگاهی و همینطور کتاب راهنمای جامع matlab را تو صیه میکنیم .

ولی آنهایی که می توانند از متن های لاتین استفاده کنند.

هیچ کتابی نگیرید در مرحله اول یک matlab با ورژن بالا تهیه کنید (من 7 را دارم که جدیدا 8 آمده ولی ندیدم) و به صورت کامل (با help و همه toolbox ها) نصب نماید هر جایی سئوالی برایتان پیش آمد از help استفاده کنید.

اگر help را باز کرده باشید می بینید چندین ابزار دارد که یکی search و یکی index است که برای یافتن در مورد موضوعی از search استفاده کنید و برای گرفتن اطلاعاتی در مورد توابع و دستورها از index استفاده نمایید.

شاید باور نکنید ولی غنی ترین مرجع help هر نرم افزار میباشد.

ولی باز کسانی هستند که بخواهند مرجع دیگری داشته باشند:

این فایلها را search و download کنید.

Getting started with matlab.pdf

Refbook.pdf

Refbook1.pdf

Refbook2.pdf

Matlab toolbox quick reference.pdf

مقاله اول آموزش است که ممکن است از 70 صفحه تا 1000 صفحه باشد ((مانده از کجا dow load کنید))

سه مقاله بعدی مرجع دستورات می باشند که اولی حروفات a-e دومی f-0 و سومی p-z می باشد.

و آخرین مقاله مربوط است به طبقه بندی توابع و دستورات بر اساس toolbox و معرفی آنها.

البته همه این مقالات و همچنین مقالات مفید دیگری در یک CD همراه ارائه شده که امیدواریم مورد استفاده هر چه بیشتر شما قرار گیرد.

امید داریم که این بسته مورد استفاده هر بیشتر شما قرار گرفته باشد.

در پایان :

خواهشمندیم نظرتان را در مورد این بسته به ما اطلاع دهید.