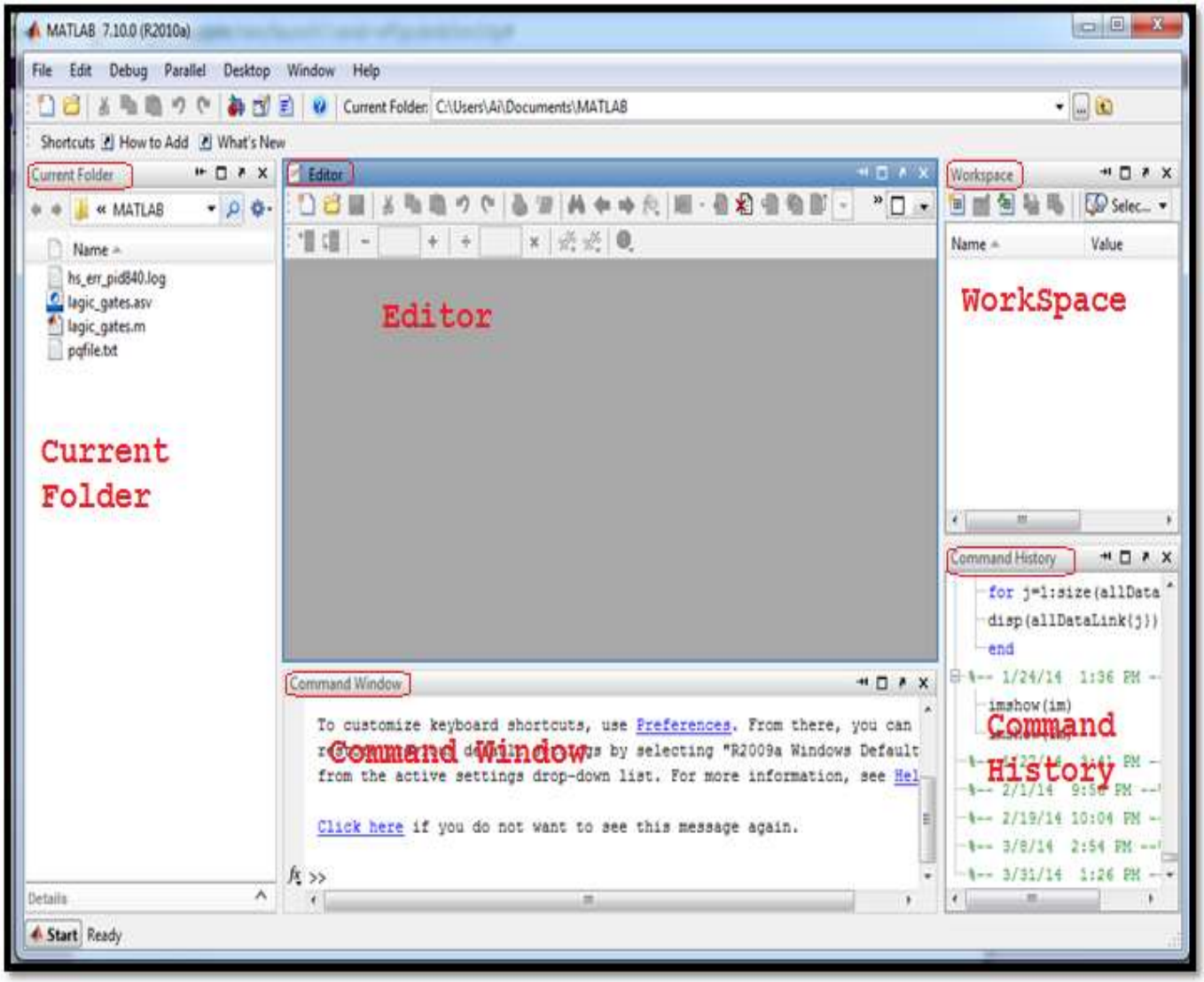
✓ معرفی نرم افزار متلب:

متلب یک سیستم جامع نرم افزاری برای محاسبات ریاضی و محاسبات تکنیکی می باشد. کاربردهای نرم افزار متلب بسیار وسیع می باشد. شما با متلب حتی می توانید صدا و یا انیمیشن بسازید .
شرکت توسعه دهنده نرم افزار متلب، شرکت MathWorks می باشد.

✓ آشنایی با محیط اصلی نرم افزار متلب:

هنگامی که وارد محیط نرم افزار متلب بشوید، پنجره اصلی نرم افزار، خود شامل پنجره های زیر است :



- پنجره : Command

در پنجره Command می توانیم دستورات خود را نوشته و سپس با فشار دادن کلید enter از کیبورد، نتایج اجرای دستورات را، در همین پنجره، مشاهده کنیم .

- پنجره : Workspace

در پنجره Workspace ، لیستی از متغیرهایی که به وسیله دستورات در متلب تعریف شده است، نمایش داده می شود .

- پنجره : Current Folder

در پنجره Current Folder می توانیم پوشه ای که در آن فایل های متلب مورد نظرمان وجود دارد را مشاهده کنیم و به آسانی پوشه و یا فایل های مورد نظرمان را بیابیم .

- پنجره : Command History

در پنجره Command History ، لیستی از دستوراتی که در متلب اجرا کرده ایم، نمایش داده می شود.

✓ نحوه اجرای دستورات در متلب:

بهترین روش برای یادگیری متلب این است که از دستورات ساده شروع کنید و نتایج آن را مشاهده کنید، هرگاه دستوری را در متلب اشتباه وارد کنید، متلب پیغام خطایی در پنجره Command نمایش می دهد که نوع خطا و همچنین محل خطا را برای شما مشخص می کند. پس با دستورات ساده شروع کنید، نتایج و یا پیام های خطا را مشاهده کرده و به تدریج به سراغ دستورات پیچیده تر بروید.

✓ نوشتن دستورات در پنجره: Command

ابتدا باید در پنجره Command کلیک کنید تا فعال شود، سپس می توانید دستورات مورد نظر خود را وارد کرده و با فشار دادن کلید enter از کیبورد، نتیجه اجرای دستورات را در همان پنجره ببینید.

مثال : دستور ساده زیر را وارد می کنیم:

```
۱+۳
```

نتیجه به صورت زیر در پنجره Command نمایش داده می شود:

نتیجه :

```
ans =
      ٤
```

ansنمایشگر ابتدای کلمه answer است. هنگامی که در متلب برای نتیجه یک محاسبه، نامی انتخاب نشده باشد، خود نرم افزار متلب، نام ans را برای آن انتخاب می کند، یعنی این که متغیری به نام ans با مقدار ۲ ایجاد می کند. چنانچه برای محاسبات بعدی به این عدد احتیاج داشته باشید، باید حتما نام دیگری برای آن انتخاب کنیدمثلا، a=(۱+۳) ،زیرا نرم افزار متلب به دستور بعدی که نام متغیری برای آن در نظر گرفته نشده باشد، دوباره نام ans را اختصاص می دهد و عملا مقدار قبلی آن پاک می شود.

نکته :

چنانچه تمایل داشته باشید که دستورات قبلی و نتایج آنها که در پنجره Command نمایش داده شده اند، پاک شوند، تنها کافی است که بر روی قسمتی از پنجره Command ، کلیک سمت راست کرده و گزینه Clear Command Windowرا انتخاب نمایید. باید دقت داشته باشید که با این کار، تنها دستورات و نتایج نشان داده

شده در پنجره Command پاک می شوند، اما متغیرهایی که توسط این دستورات در متلب تعریف شده اند، همچنان وجود دارند و می توان از آنها استفاده نمود. دستوری که متغیرها را به طور کامل از متلب پاک می کند، دستور clear می باشد که در دروس بعدی در مورد آن صحبت خواهیم کرد. همچنین چنانچه برنامه متلب را ببندید، تمامی متغیرهای تعریف شده در متلب پاک می شوند و دفعه بعد که متلب را باز کنید، هیچ متغیری در آن تعریف نشده است.

نکته :

چنانچه بخواهید نتیجه اجرای خطی از دستورات در پنجره Command نمایش داده نشود، تنها کافی است که در پایان آن خط از دستورات، علامت ; را بنویسید. با به کار بردن این علامت در پایان هر خط، متلب دستورات آن خط را اجرا کرده و نتیجه محاسبات را در متغیرها ذخیره می کند، اما نتیجه محاسبات را در پنجره Command نمایش نخواهد داد.

انواع متغیرها و مقداردهی به متغیرها در متلب:

یکی از ویژگی های متلب این است که احتیاجی نیست که حتما نوع متغیر را در همان ابتدای برنامه مشخص کنیم و با مقادیری که در طول برنامه به متغیر نسبت داده می شود، نوع متغیر به صورت خود به خود تعیین می شود .

در تعریف نام متغیرها باید دقت داشته باشید که متلب نسبت به کوچک یا بزرگ بودن حروف حساس می باشد.

✓ انواع متغیرها:

در نرم افزار متلب، انواع مختلفی از متغیرها وجود دارد. برخی از آنها را شرح می دهیم:

- **متغیرهای عددی:**

این متغیرها می توانند دارای مقادیر عددی باشند. به مثال زیر توجه کنید:

مثال :

فرض کنید بخواهیم به متغیر A ، مقدار ۲ را نسبت بدهیم. باید بنویسیم:

```
A=۲
```

نتیجه :

```
A =
     ۲
```

- **متغیر های رشته ای: (string)**

چنانچه متغیری را بخواهیم به صورت یک رشته از حروف تعریف کنیم، باید از علامت ' استفاده کنیم. به مثال زیر توجه کنید:

مثال :

```
s='this is a string'
```

نتیجه :

```
s =
this is a string
```

نکته :

دقت شود که استفاده از علامت ' برای ایجاد رشته ها ضروری است و چناچه از این علامت استفاده نشود، با پیغام خطا مواجه می شویم. این موضوع را در مثال زیر نشان داده ایم:

مثال :

```
s=this is a string
```

نتیجه :

```
??? s=this is a string
          |
Error: Unexpected MATLAB expression.
```

✓ دستور: clear

این دستور برای پاک کردن متغیرهای تعریف شده در متلب به کار می رود. این دستور را می توان به شیوه های زیر به کار برد:

clear	تمامی متغیرهای تعریف شده در متلب را پاک می کند
clear all	تمامی متغیرهای تعریف شده در متلب را پاک می کند
clear x y	تنها متغیرهای x و y را پاک می کند

نکته بسیار مهم :

یک برنامه نویس متلب، معمولا اولین دستوری که در برنامه خود به کار می برد، دستور clear all می باشد، زیرا امکان این که متغیرهایی که قبلا در متلب، توسط برنامه های قبلی، تعریف شده اند در برنامه جدید اختلال ایجاد کنند، زیاد است. این نکته در اجرای برنامه های طولانی و پیچیده، از اهمیت زیادی برخوردار است.

✓ پاک کردن اطلاعات نمایش داده شده در پنجره command با clc :

دستور clc در متلب، برای پاک کردن اطلاعات نمایش داده شده در پنجره command به کار می رود. یعنی هر زمان که خواستید تمامی اطلاعات نمایش داده شده در پنجره (command البته تا این لحظه و نه برای اطلاعاتی که در آینده نمایش داده می شوند) ، پاک شوند، تنها کافی است که دستور زیر را بنویسید:

```
clc
```

نکته :

بد نیست به این نکته اشاره کنم که یک شروع خوب برای کدهای یک برنامه متلب، به صورت زیر می باشد:

```
clear all
close all
clc
```

که در آن دستور clear all ، تمامی متغیرهایی که قبلا در متلب تعریف شده اند را پاک می کند، دستور close all، تمامی پنجره های شکلی که قبلا در متلب باز شده اند را می بندد و دستور clc نیز تمامی اطلاعاتی که قبلا در پنجره command نمایش داده شده است را پاک می کند. بنابراین این سه خط کد، یک شروع خوب برای هر برنامه متلب می باشد تا برنامه بدون هیچ مشکل و تداخلی با اطلاعات قبلی اجرا شود.

عملگرهای ریاضی در متلب:

در متلب برای استفاده از عملگرهای ریاضی، دو شیوه به کار رفته است. شیوه اول برای عملگرهای بسیار رایج می باشد، مانند جمع، تفریق، ضرب، تقسیم و ... که متلب برای هر کدام از آنها یک نماد را در نظر گرفته است، یعنی شما در هنگام نوشتن دستورات، تنها کافی است که از کیبورد، نماد مربوط به آن عملگر را در خط دستورات وارد کنید (مثلا زدن دکمه مربوط به نماد * از کیبورد برای عملگر ضرب)

شیوه دوم، استفاده از تابع است. متلب برای عملگرهای ریاضی که میزان استفاده از آنها زیاد نیست، دستوراتی را ساخته است و شما باید نام آن دستورات و نحوه استفاده از آنها را بدانید (مثلا برای عملگر رادیکال ۲ ، دستور sqrt در نظر گرفته شده است)

در ادامه عملگرهای مربوط به هر دو شیوه را شرح خواهیم داد.

• عملگرهای ریاضی مشخص شده با نماد در متلب:

همان طور که گفتیم برای هر یک از این عملگرهای ریاضی، در متلب یک نماد خاص اختصاص یافته است. در جدول زیر، نمادهای مربوط به عملگرهای ریاضی پر کاربرد در متلب، نوشته شده است:

نماد در متلب	عملگر
+	جمع
-	تفریق
*	ضرب
/	تقسیم
^	به توان

مثال :

```
۳*۲-(۳*٤)/۲+٤^۲
```

نتیجه :

```
ans =
     ۱٦
```

مثال :

برای به توان ۲ رساندن عدد ۳ ، این گونه می نویسیم:

```
۳^۲
```

نتیجه :

```
ans =
     ۹
```

• عملگرهای ریاضی مشخص شده با دستور در متلب:

در متلب برای هر کدام از این عملگرهای ریاضی، یک دستور در نظر گرفته شده است. برای نمونه تعدادی از این عملگرها را در جدول زیر آورده ایم:

دستور در متلب	عملگر
sqrt	رادیکال ۲
log	Ln (لگاریتم طبیعی)
log۱۰	Log (لگاریتم بر مبنای ۱۰)

مثال : با دستور sqrt ، رادیکال ۲ عدد ٤ را محاسبه می کنیم:

```
A=sqrt(٤)
```

نتیجه :

```
A =
     ۲
```

مثال : با دستورlog۱۰ ، لگاریتم بر مبنای ۱۰ عدد ۱۰۰ را محاسبه می کنیم:

```
A=log۱۰(۱۰۰)
```

نتیجه :

```
A =
     ۲
```

✓ نمایش مقدار متغیر بدون نام متغیر با دستور: disp

نرم افزار متلب، برای نمایش مقدار یک متغیر، ابتدا نام متغیر را می نویسد، سپس یک علامت تساوی (=) را نوشته و در زیر آن، مقدار متغیر را نمایش می دهد. ممکن است در مواردی بخواهیم که تنها مقدار متغیر، در پنجره Command ، نمایش داده شود و نام متغیر، نمایش داده نشود. در این صورت، باید از دستور disp استفاده کنیم. به مثال زیر توجه کنید:

مثال :

برای مقایسه، ابتدا روش معمولی برای نمایش مقدار متغیر را به کار می بریم:

```
A=۲;
A
```

نتیجه :

```
A =
     ۲
```

حال از دستور disp استفاده می کنیم:

```
A=۲;
disp(A)
```

نتیجه :

```
     ۲
```

مشاهده می کنید که تنها مقدار متغیر A نمایش داده شده است و دیگر از نمایش نماد A و علامت تساوی (=) خبری نیست.

m-fileها در متلب:

چنانچه بخواهید برنامه ای طولانی و پیچیده بنویسید، دیگر پنجره Command جوابگوی نیاز شما نیست و به محیطی فراتر از آن برای نوشتن دستورات و تصحیح کردن آنها نیاز دارید. متلب برای این گونه موارد، امکان ساخت m-file ها را فراهم کرده است. شما می توانید در یک m-file ، تمامی دستورات خود را نوشته و تنها بر روی یک دکمه گرافیکی کلیک کرده و سپس نتیجه اجرای دستورات را در پنجره Command ببینید.

➢ ساخت یک m-file در متلب:

برای ساخت یک m-file جدید می توانید از هر یک از روش های زیر استفاده کنید :

۱-در بالای پنجره اصلی نرم افزار متلب، بر روی گزینه New script کلیک کنید. این گزینه به شکل  می باشد .

۲-با نگه داشتن کلید Ctrl و فشار دادن کلید N از کیبورد، این کار را انجام دهید .

هر یک از روش های بالا را که انتخاب کنید، نتیجه این است که متلب یک پنجره خالی باز می کند که می توانید در آن، دستورات خود را اجرا کنید .

توصیه می شود اولین دستوری که در یک m-file می نویسید، دستور clear all باشد تا تمامی متغیرهایی که قبلا در متلب تعریف شده است را پاک کند و اختلالی در روند اجرای برنامه ایجاد نشود .

باید دقت داشته باشید که در نرم افزار متلب، m-fileها برای دو هدف اصلی به کار می روند،:

۱) کاربرد اول آن نوشتن برنامه های پیچیده و طولانی

۲) کاربرد دوم آن ساخت تابع می باشد.

ساخت تابع با استفاده از m-file را در مباحث بعدی توضیح خواهیم داد. در این مبحث تنها در مورد نوشتن برنامه در m-file ها صحبت خواهیم کرد .

پس از آنکه دستورات برنامه را در m-file نوشتیم، ابتدا باید با استفاده از گزینه Save در بالای همان پنجره m-file، آن را ذخیره کنیم. همچنین با نگه داشتن کلید Ctrl و فشار دادن کلید S ، می توانید این کار را انجام دهید .

سپس برای اجرای برنامه، باید بر روی گزینه Save and run که به شکل  می باشد، کلیک کنید تا نتایج برنامه در پنجره Command نمایش داده شود. همانطور که از نام این گزینه مشخص است، این گزینه، عمل ذخیره کردن را هم انجام می دهد، یعنی اگر تغییراتی در برنامه ایجاد کنید و سپس بر روی این گزینه کلیک کنید، این تغییرات در m-file ذخیره می شود. اگر قبلا فایل ذخیره نشده باشد، ابتدا از شما می خواهد که نامی برای آن انتخاب کرده و سپس آن را ذخیره کنید .

m-fileها دارای پسوند m می باشند .

➢ اجرای دستورات درون یک m-file بدون باز کردن آن:

حتی بدون باز کردن یک m-file نیز می توان برنامه نوشته شده در آن را اجرا کرد. برای این منظور باید ابتدا پنجره Current Folder به گونه ای باشد که فولدر حاوی m-file مورد نظرمان را نمایش بدهد.فرض کنید نام m-fileحاوی برنامه، program.m باشد، بنابراین باید نام فایل program.m را در پنجره Current Folder ببینیم. سپس تنها کافی است که در پنجره Command بنویسیم program و سپس کلید enter از کیبورد را فشار دهیم. دقت شود که نباید پسوندm. نوشته شود. نتیجه اجرای دستورات در پنجره Command نمایش داده خواهد شد.

➢ نوشتن توضیحات در: m-file

زمانی که یک برنامه طولانی بنویسید، به دلیل حجم زیاد دستورات، ممکن است بخشی از روند برنامه نویسی را فراموش کنید. با استفاده از ۲ تکنیک زیر، می توان این مشکل را تا حد زیادی برطرف کرد :

۱-انتخاب هوشمندانه نام متغیرها به گونه ای که هدف استفاده از آنها را بتوان از نامشان به طور کامل درک کرد .

۲-می توانیم هنگام نوشتن برنامه، توضیحاتی را در کنار کدها بنویسیم، تا با خواندن آنها خود برنامه نویس یا هر شخص دیگری به راحتی درک کند که روش های استفاده شده در برنامه چیست.

در متلب چنانچه از علامت درصد (%) استفاده کنیم، تمامی نوشته های بعد از علامت درصد، به صورت توضیح در نظر گرفته می شوند. به مثال زیر توجه کنید:

مثال :

```
x=۲
% www.ketabestan.blogfa.com
y=۳
```

نتیجه :

```
x =
     ۲
 y =
     ۳
```

همان طور که مشاهده می کنید، www.Ketabestan.blogfa.com به عنوان دستور در نظر گرفته نشده و در خروجی نیز نمایش داده نشده است.

باید دقت داشته باشید که متلب نوشته های بعد از علامت درصد را تنها در خط فعلی، به صورت توضیح در نظر می گیرد و نوشته های خط بعد را به صورت دستور (نه توضیح) در نظر خواهد گرفت. بنابراین چنانچه بخواهیم توضیحاتی را در چند خط پشت سرهم بنویسیم، باید در ابتدای هر کدام از آن خط ها، از علامت درصد استفاده کنیم. به مثال زیر توجه کنید:

مثال :

```
x=۲
% www.ketabestan.blogfa.com
% this is a simple code
y=۳
```

نتیجه :

```
x =
     ۲
y =
     ۳
```

بردارها و ماتریس ها در متلب:

نرم افزار متلب، بر اساس کار کردن با بردارها و ماتریس ها ساخته شده است.

✓ بردارها در متلب:

یک بردار، فهرستی از اعداد می باشد که پشت سرهم چیده شده اند. به هر کدام از این اعداد، یک عنصر بردار می گوییم. بردارها را با روش های مختلفی می توان در متلب تعریف نمود .

روش اول : استفاده از علامت کاما (،) برای جدا کردن عناصر بردار و قرار دادن آنها درون براکت

مثال :

```
A=[۱,۲,۳,٤]
```

نتیجه :

```
A =
     ۱     ۲     ۳     ٤
```

روش دوم : استفاده از فاصله برای جدا کردن عناصر بردار و قرار دادن آنها درون براکت

مثال :

```
A=[۱ ۲ ۳ ٤]
```

نتیجه :

```
A =
     ۱     ۲     ۳     ٤
```

✓ ساخت بردارهایی با عناصر قاعده مند:

فرض کنید بخواهیم برداری تعریف کنیم که عناصر آن شامل اعداد ۱ تا ۹ باشد که به ترتیب پشت سرهم باشند، برای این گونه موارد، نرم افزار متلب شیوه ای از علامت گذاری را به کار می برد که در مثال زیر نشان داده شده است و دیگر لازم نیست که این ۹ عدد را به شیوه های ذکر شده قبلی در بردار تعریف کنیم:

مثال :

```
A=۱:۹
```

نتیجه :

```
A =
     ۱     ۲     ۳     ٤     ٥     ٦     ۷     ۸     ۹
```

عدد ۱ ، اولین عدد است و آن قدر به آن، یک واحد یک واحد، اضافه می شود تا به آخرین عدد که برابر ۹ است برسیم .

حال فرض کنید که به جای اعداد ۱ تا ۹ می بایست اعداد ۱ تا ۹۰۰ را به عنوان عناصر بردار تعریف می کردیم، مطمئنا نوشتن اعداد ۱ تا ۹۰۰ کار ساده ای نیست، اما شیوه فوق، این کار را به راحتی انجام می دهد.

برای شیوه ذکر شده، نوع دیگری از علامت گذاری نیز وجود دارد که در مثال زیر نشان داده شده است:

مثال :

```
A=۱:۱:۹
```

نتیجه :

```
A =
     ۱     ۲     ۳     ٤     ٥     ٦     ۷     ۸     ۹
```

تنها تفاوت در این است که میزان افزایش که برابر ۱ واحد است را در علامت گذاری ذکر کرده ایم. در واقع دستور A=۱:۹ نوعی اختصار برای دستور A=۱:۱:۹ در متلب می باشد.

مثال :

فرض کنید بخواهیم اعداد زوج بین ۰ تا ۱۰ را به عنوان عناصر یک بردار در متلب تعریف کنیم، می نویسیم:

```
A=۰:۲:۱۰
```

نتیجه :

```
A =
     ۰     ۲     ٤     ٦     ۸    ۱۰
```

اولین عنصر بردار برابر ۰ است و هر بار، ۲ واحد به آن اضافه می شود تا عنصر بعدی بردار ساخته شود، تا زمانی که به عدد ۱۰ برسیم که آخرین عنصر بردار می باشد.

نکته :

میزان افزایش عناصر بردار می تواند یک عدد اعشاری باشد. به مثال زیر توجه کنید:

مثال :

فرض کنید بخواهیم اعداد زوج بین ۰ تا ۱۰ را به عنوان عناصر یک بردار در متلب تعریف کنیم، می نویسیم:

```
A=۱:۰,۲:۲
```

نتیجه :

```
A =
    ۱,۰۰۰۰    ۱,۲۰۰۰    ۱,٤۰۰۰    ۱,٦۰۰۰    ۱,۸۰۰۰    ۲,۰۰۰۰
```

نکته :

میزان تغییر منظم عناصر بردار می تواند به صورت کاهشی باشد. برای این منظور باید یک عدد منفی را، متناسب با میزان کاهش، مشخص کنیم. به مثال زیر توجه کنید:

مثال :

```
A=۹:-۱:۱
```

نتیجه :

```
A =
     ۹     ۸     ۷     ٦     ٥     ٤     ۳     ۲     ۱
```

✓ اشاره به عناصر یک بردار:

برای آن که یک عنصر از عناصر برداری را مشخص کنیم، تنها کافی است که شماره آن عنصر را بدانیم. به مثال زیر توجه کنید:

مثال :

```
A=[۱ ۲ ۳ ٤]
B=A(۳)
```

نتیجه :

```
A =
     ۱     ۲     ۳     ٤
 B =
     ۳
```

دستور B=A(۳)، متغیر B را برابر عنصر سوم از بردار A قرار می دهد .

زمانی که برداری را به شیوه های قبلی، در متلب تعریف کنید، نرم افزار متلب به طور خودکار آن بردار را به صورت بردار ردیفی در نظر می گیرد (دارای یک ردیف و چندین ستون)، در صورتی که بخواهیم بردار به صورت بردار ستونی (دارای یک ستون و چندین ردیف) باشد، تنها کافی است که از علامت ' در انتهای نام بردار استفاده کنیم. به مثال زیر توجه کنید:

مثال :

```
A=[۱ ۲ ۳ ٤]
B=A'
```

نتیجه :

```
A =
     ۱     ۲     ۳     ٤

 B =
     ۱
     ۲
     ۳
     ٤
```

ماتریس ها در متلب:

یک ماتریس، آرایه ای مستطیلی از اعداد می باشد .

قبلا بردارها را معرفی کردیم، باید دقت داشته باشید که یک بردار ستونی در واقع ماتریسی می باشد که تنها دارای یک ستون است و یک بردار ردیفی در واقع ماتریسی می باشد که تنها دارای یک ردیف است.

✓ نحوه تعریف ماتریس ها در متلب:

برای تعریف ماتریس ها در متلب، چندین روش وجود دارد :

روش اول : تعریف عناصر ماتریس با استفاده از علامت کاما (،) برای جدا کردن عناصر و استفاده از علامت ; برای جدا کردن ردیف ها از یکدیگر. به مثال زیر توجه کنید:

مثال :

```
A=[۱,۲,۳;٤,٥,٦;۷,۸,۹]
```

نتیجه :

```
A =
     ۱     ۲     ۳
     ٤     ٥     ٦
     ۷     ۸     ۹
```

روش دوم : تعریف عناصر ماتریس با استفاده از فاصله برای جدا کردن عناصر و استفاده از علامت ; برای جدا کردن ردیف ها از یکدیگر. به مثال زیر توجه کنید:

مثال :

```
A=[۱ ۲ ۳;٤ ٥ ٦;۷ ۸ ۹]
```

نتیجه :

```
A =
     ۱     ۲     ۳
     ٤     ٥     ٦
     ۷     ۸     ۹
```

روش سوم : نرم افزار متلب برای ایجاد برخی ماتریس های خاص، دارای دستوراتی می باشد. برخی از این دستورها عبارتند از:

➢ دستور: ones

این دستور ماتریسی ایجاد می کند که تمامی عناصر آن دارای مقدار عددی ۱ می باشند. به مثال زیر توجه کنید:

مثال :

ماتریسی می سازیم که دارای ۲ ردیف و ۳ ستون باشد و تمامی عناصر آن دارای مقدار عددی ۱ باشند:

```
A=ones(۲,۳)
```

نتیجه :

```
A =
     ۱     ۱     ۱
     ۱     ۱     ۱
```

➢ دستور: zeros

این دستور، ماتریسی ایجاد می کند که تمامی عناصر آن دارای مقدار عددی ۰ می باشند. به مثال زیر توجه کنید:

مثال :

ماتریسی می سازیم که دارای ۳ ردیف و ۲ ستون باشد و تمامی عناصر آن دارای مقدار عددی ۰ باشند:

```
A=zeros(۳,۲)
```

نتیجه :

```
A =
     ۰     ۰
     ۰     ۰
     ۰     ۰
```

➢ اشاره به یک ردیف یا یک ستون ماتریس:

گاهی لازم است که به یک ردیف یا یک ستون یک ماتریس، اشاره داشته باشیم. برای این منظور، می توان از علامت : استفاده نمود. به مثال زیر توجه کنید:

مثال :

```
A=[۱ ۲ ۳;٤ ٥ ٦;۷ ۸ ۹]
B=A(۲,:)
```

نتیجه :

```
A =
     ۱     ۲     ۳
     ٤     ٥     ٦
     ۷     ۸     ۹

 B =
     ٤     ٥     ٦
```

همان طور که مشاهده می کنید، در دستور B=A(۲,:) ، عدد ۲ برای شماره ردیف و علامت :برای شماره ستون به کار رفته است، یعنی اینکه عناصری مورد نظرمان است که شماره ردیف آنها برابر ۲ باشد، اما شماره ستون آنها می تواند شماره هر ستونی باشد. بنابراین حاصل برابر تمامی عناصر ردیف دوم ماتریس A می باشد.

➢ اشاره به چند عنصر متوالی از یک ردیف یا یک ستون ماتریس:

ممکن است بخواهیم از یک ردیف یا یک ستون، تنها به چند عنصر متوالی آن، اشاره کنیم. به مثال زیر توجه کنید:

مثال :

```
A=[۱ ۲ ۳ ٤;٥ ٦ ۷ ۸;۹ ۱۰ ۱۱ ۱۲]
B=A(۲,۲:٤)
```

نتیجه :

```
A =
     ۱     ۲     ۳     ٤
     ٥     ٦     ۷     ۸
     ۹    ۱۰    ۱۱    ۱۲

B =
     ٦     ۷     ۸
```

در دستور B=A(۲،۲:٤) ، عدد ۲ برای شماره ردیف و عبارت ۲:٤ برای شماره ستون نوشته شده است. بنابراین Bبرابر عناصری از ماتریس A خواهد بود که شماره ردیف آنها برابر ۲ و شماره ستون آنها از ۲ تا ٤ می باشد

مثال :

```
A=[۱ ۲ ۳ ٤;٥ ٦ ۷ ۸;۹ ۱۰ ۱۱ ۱۲]
B=A(۱:۲,۲:٤)
```

نتیجه :

```
A =
     ۱     ۲     ۳     ٤
     ٥     ٦     ۷     ۸
     ۹    ۱۰    ۱۱    ۱۲

 B =
     ۲     ۳     ٤
     ٦     ۷     ۸
```

➢ اشاره به چند عنصر غیر متوالی از یک ردیف یا یک ستون ماتریس:

گاهی ممکن است عناصر مورد نظرمان متوالی نباشند، در اینگونه موارد، نمی توانیم از علامت : استفاده کنیم و باید شماره ردیف یا ستون عناصر مورد نظرمان را درون علامت های] و [قرار بدهیم. به مثال زیر توجه کنید:

مثال :

```
A=[۱ ۲ ۳;٤ ٥ ٦;۷ ۸ ۹]
B=A(۲,[۱ ۳])
```

نتیجه :

```
A =
     ۱     ۲     ۳
     ٤     ٥     ٦
     ۷     ۸     ۹

 B =
     ٤     ٦
```

در دستور B=A(۲,[۱ ۳]) ، عدد ۲ برای شماره ردیف و عبارت [۱ ۳] برای شماره ستون به کار رفته است. عبارت [۱ ۳] برای شماره ستون، به این معنی است که ستون شماره ۱ و ستون شماره ۳ مورد نظرمان بوده است .

➢ انجام عملیات های ریاضی بر روی عناصر یک بردار:

در برنامه های متلب، بسیار زیاد پیش می آید که بخواهیم یک عمل ریاضی را بر روی تمامی عناصر یک بردار انجام دهیم. نرم افزار متلب، برای این موارد، از نوعی علامت گذاری استفاده می کند که در مثال زیر بیان شده است:

مثال :

فرض کنیم بخواهیم تمامی عناصر بردار A را به توان ۲ برسانیم، می نویسیم:

```
A=[۱ ۲ ۳ ٤]
B=A.^۲
```

نتیجه :

```
A =
     ۱     ۲     ۳     ٤

 B =
     ۱     ٤     ۹    ۱٦
```

نکته :

اهمیت علامت نقطه (.) بسیار زیاد است، زیرا این علامت است که مشخص می کند که عمل ریاضی مشخص شده، بر روی هر عنصر بردار صورت گیرد و چنانچه علامت نقطه گذاشته نشود، آن عمل ریاضی بر روی کل بردار انجام می شود (عملیات های برداری) که مطمئنا نتیجه ای غیر از آنچه می خواستیم به ما خواهد داد . در صورت نگذاشتن علامت نقطه (.)، چنانچه آن عمل ریاضی قابل اجرا بر روی کل بردار نباشد، متلب یک پیام خطا را در خروجی نمایش می دهد، زیرا متلب سعی می کند که آن عمل ریاضی را بر روی کل بردار اجرا کند. برای روشن شدن این موضوع، به مثال زیر توجه کنید که در واقع همان مثال قبلی بدون علامت نقطه (.) می باشد:

مثال :

```
A=[۱ ۲ ۳ ٤]
B=A^۲
```

نتیجه :

```
A =

     ۱     ۲     ۳     ٤

??? Error using \ mpower
Inputs must be a scalar and a square matrix.
To compute elementwise POWER, use POWER (.^) instead.
```

پیام خطای فوق به این دلیل است که متلب می خواهد کل بردار A را به توان ۲ برساند و نمی تواند این کار را انجام دهد، زیرا تنها اعداد اسکالر و ماتریس های مربعی را می توان به توان ۲ رساند.

مثال :

فرض کنید عناصر متناظر دو بردار A و B را بخواهیم به صورت تک تک در هم ضرب کنیم، می نویسیم:

```
A=[۱ ۲ ۳ ٤]
B=[۲ ۳ ٤ ٥]
C=A.*B
```

نتیجه :

```
A =
     ۱     ۲     ۳     ٤

B =
     ۲     ۳     ٤     ٥

C =
     ۲     ٦    ۱۲    ۲۰
```

نگذاشتن علامت نقطه (.) در دستور فوق باعث می شود که متلب یک پیام خطا را در خروجی نمایش دهد.

نکته :

برای دو عملگر ریاضی - و + احتیاجی نیست که علامت نقطه گذاشته شود، زیرا نرم افزار متلب به طور خودکار این عملگرها را بر روی تک تک عناصر بردارها اجرا می کند. به مثال زیر توجه کنید:

مثال :

```
A=[٤ ٤ ٤ ٤]
B=[۱ ۲ ۳ ٤]
C=A-B
```

نتیجه :

```
A =
     ٤     ٤     ٤     ٤

 B =
     ۱     ۲     ۳     ٤

C =
     ۳     ۲     ۱     ۰
```

➢ محاسبه اندازه یک ماتریس با دستور size در متلب:

فرض کنید که ماتریسی داریم که می خواهیم بدانیم اندازه آن چقدر است، برای این منظور، دستور size در متلب، مورد استفاده قرار می گیرد. به مثال زیر توجه کنید:

مثال :

```
A=[۱ ۲ ۳;٤ ٥ ٦]
B=size(A)
```

نتیجه :

```
A =
     ۱     ۲     ۳
     ٤     ٥     ٦

B =
     ۲     ۳
```

مشاهده می کنید که دستور B=size(A) ، تعداد ردیف ها و تعداد ستون های ماتریس A را محاسبه کرده است و به ترتیب، عدد مربوط به آنها را در متغیر B ذخیره کرده است.

✓ محاسبه مینیمم (min) یا ماکزیمم (max) یک ماتریس در متلب:

در متلب برای به دست آوردن مینیمم (min) یا ماکزیمم (max) یک ماتریس، به ترتیب از دستور min و دستور maxاستفاده می شود.

➢ محاسبه مینیمم (min) یک ماتریس در متلب:

در متلب، برای محاسبه مینیمم (min) یک ماتریس، از دستور min استفاده می شود. اگر A یک ماتریس دو بعدی باشد، دستور min(A) ، برداری را بر می گرداند که در آن مینیمم هر ستون ماتریس A ، مشخص شده است و دستور min(min(A)) ، مینیمم ماتریس A را محاسبه می کند. به مثال زیر توجه کنید:

مثال :

```
A=[۱ ۲;۳ ٤]
B=min(A)
C=min(min(A))
```

نتیجه :

```
A =
     ۱     ۲
     ۳     ٤

B =
     ۱     ۲

C =
     ۱
```

➢ محاسبه ماکزیمم (max) یک ماتریس در متلب:

در متلب، برای محاسبه ماکزیمم (max) یک ماتریس، از دستور max استفاده می شود. اگر A یک ماتریس دو بعدی باشد، دستور max(A) ، برداری را بر می گرداند که در آن ماکزیمم هر ستون ماتریس A ، مشخص شده است و دستور max(max (A)) ، ماکزیمم ماتریس A را محاسبه می کند. به مثال زیر توجه کنید:

مثال :

```
A=[۱ ۲;۳ ٤]
B=max(A)
C=max(max(A))
```

نتیجه :

```
A =
     ۱     ۲
     ۳     ٤

B =
     ۳     ٤

 C =
     ٤
```

➢ محاسبه طول یک بردار با دستور length در متلب:

در متلب، با استفاده از دستور length ، می توانیم طول یک بردار (تعداد کل عناصر بردار) را محاسبه کنیم. به مثال زیر توجه کنید:

مثال :

```
A=[۷ ۸ ۹]
B=length(A)
```

نتیجه :

```
A =
     ۷     ۸     ۹

B =
     ۳
```

البته دستور length ،برای ماتریس ها نیز می تواند مورد استفاده قرار بگیرد. اگر دستور length را برای یک ماتریس به کار ببریم، این دستور، تعداد ردیف ها و تعداد ستون های ماتریس را محاسبه می کند و هر کدام از این دو عدد که بزرگتر باشد را در خروجی نمایش خواهد داد. به مثال زیر توجه کنید:

مثال :

```
A=[۱ ۲ ۳;٤ ٥ ٦]
B=length(A)
```

نتیجه :

```
A =
     ۱     ۲     ۳
     ٤     ٥     ٦

B =
     ۳
```

➢ مرتب کردن عناصر یک بردار یا ماتریس، به صورت صعودی یا نزولی، با دستور sort در متلب:

دستور sort در متلب، برای مرتب کردن عناصر یک بردار یا ماتریس، به صورت صعودی یا نزولی، به کار می رود. چنانچه در دستور sort مشخص نکنیم که می خواهیم عناصر به صورت صعودی مرتب شود یا نزولی، دستور sort، به طور پیش فرض، عناصر را به صورت صعودی مرتب می کند.

به مثال زیر توجه کنید:

مثال :

با دستور sort ، عناصر یک بردار را به صورت صعودی مرتب می کنیم:

```
clear all
close all
clc

A=[٥ ۲ ٤ ۳ ۱]
B=sort(A)
```

نتیجه :

```
A =
     ٥     ۲     ٤     ۳     ۱

B =
     ۱     ۲     ۳     ٤     ٥
```

نکته :

چنانچه بخواهیم دستور sort ، عناصر را به صورت نزولی مرتب کنید، باید عبارت 'descend' را درون پرانتز دستور sortبنویسیم.

به مثال زیر توجه کنید:

مثال :

با دستور sort ، عناصر یک بردار را به صورت نزولی مرتب می کنیم:

```
clear all
close all
clc

A=[٥ ٢ ٤ ٣ ١]
B=sort(A,'descend')
```

نتیجه :

```
A =
     ٥     ٢     ٤     ٣     ١

B =
     ٥     ٤     ٣     ٢     ١
```

نکته :

چنانچه یک ماتریس به دستور sort داده شود، دستور sort عناصر را در یکی از ابعادی که مشخص می کنیم، مرتب خواهد نمود. اگر هیچ بعدی را مشخص نکنیم، دستور sort ، عناصر را در جهت ستون ها مرتب خواهد نمود.

به مثال زیر توجه کنید:

مثال :

عناصر ماتریس A را در جهت ستون ها، مرتب می کنیم:

```
clear all
close all
clc

A=[٥ ٩ ٧; ٣ ١ ٤; ٦ ٢ ٨]
B=sort(A)
```

نتیجه :

```
A =
     ٥     ٩     ٧
     ٣     ١     ٤
     ٦     ٢     ٨

B =
     ٣     ١     ٤
     ٥     ٢     ٧
     ٦     ٩     ٨
```

مثال :

عناصر ماتریس A را در جهت ردیف ها، مرتب می کنیم:

```
clear all
close all
clc

A=[٥ ٩ ٧; ٣ ١ ٤; ٦ ٢ ٨]
B=sort(A,٢)
```

نتیجه :

```
A =
     ٥     ٩     ٧
     ٣     ١     ٤
     ٦     ٢     ٨

B =
     ٥     ٧     ٩
     ١     ٣     ٤
     ٢     ٦     ٨
```

تولید اعداد تصادفی در متلب:

مواردی پیش می آید که شما نیاز به یک یا چند عدد تصادفی در برنامه خود دارید. متلب دستوراتی دارد که با آنها می توانید اعدادی تصادفی ایجاد کنید.

➢ دستور: rand

این دستور قادر است اعداد تصادفی در بازه (۰،۱) بسازد. به مثال زیر توجه کنید:

مثال :

```
A=rand(۲,۳)
```

نتیجه :

```
A =
    ۰,۲۷۸٥    ۰,۹٥۷٥    ۰,۱٥۷٦
    ۰,٥٤٦۹    ۰,۹٦٤۹    ۰,۹۷۰٦
```

دستور rand (۲،۳)، ماتریسی با ۲ ردیف و ۳ ستون می سازد که عناصر آن به صورت اعداد تصادفی در بازه (۰،۱) انتخاب شده اند.

اگر درون پرانتز دستور rand ، تنها یک عدد بنویسیم، مثلا rand (۲)، آنگاه دستور rand ، تعداد ردیف و تعداد ستون ها را برابر آن عدد می گیرد. به مثال زیر توجه کنید:

مثال :

```
A=rand(۲)
```

نتیجه :

```
A =
    ۰,۹٥۷۲    ۰,۸۰۰۳
    ۰,٤۸٥٤    ۰,۱٤۱۹
```

مشاهده می کنید که ماتریس، دارای ۲ ردیف و ۲ ستون می باشد.

➢ ایجاد اعداد تصادفی در بازه ای غیر از (۰،۱) با دستور: rand

برای آنکه اعداد تصادفی، در بازه ای غیر از بازه (۰،۱) باشند، تنها کافی است که از یک فرمول ساده استفاده کنیم. اگر اعداد تصادفی در بازه (۰،۱) را داشته باشیم، آنگاه با فرمول به کار رفته در مثال زیر، می توانیم اعداد تصادفی در بازه (a,b) بسازیم:

مثال :

فرض کنید بخواهیم ماتریسی حاوی اعداد تصادفی در بازه (۸،۱۲) بسازیم، می نویسیم:

```
a=۸;
b=۱۲;
A=a+(b-a)*rand(۲,۳)
```

نتیجه :

```
A =
    ٩,٦٨٧٠    ١١,١٦٨٨    ١٠,٦٢٣٠
   ١١,٦٦٢٩    ١١,٨٣٨٠     ٨,١٤٢٨
```

✓ چیدمان اعداد صحیح ١ تا n به صورت تصادفی در یک بردار با دستور randpermدر متلب:

چنانچه از دستور randperm به صورت randperm(n) استفاده کنیم، آنگاه این دستور در خروجی، یک بردار را برمی گرداند که در آن بردار، اعداد صحیح ١ تا n به صورت تصادفی قرار گرفته اند. به مثال زیر توجه کنید:

مثال :

```
A=randperm(٩)
```

نتیجه :

```
A =
     ٦     ٣     ٧     ٨     ٥     ١     ٢     ٤     ٩
```

اجرای دستورات شرطی با دستور if در متلب:

از دستور if در متلب، برای اجرای دستورات شرطی استفاده می شود. یعنی اینکه در ابتدا شرط یا شرط هایی توسط متلب چک می شود و اگر آن شرط یا شرط ها برآورده شده باشد، آنگاه متلب دستورات مشخص شده را اجرا خواهد کرد. به مثال زیر توجه کنید:

مثال :

```
A=٥
if A>=٠
    B=A
end
if A<=٠
    B=-A
end
```

نتیجه :

```
A =
     ٥

 B =
     ٥
```

همان طور که مشاهده می کنید، از دو دستور if استفاده کرده ایم. هدف این است که مقدار B برابر قدرمطلق Aباشد، بنابراین اگر A مساوی یا بزرگتر از صفر باشد، باید B را برابر A قرار دهیم و اگر A مساوی یا کوچکتر از صفر باشد، باید B را برابر –A قرار دهیم.

دقت کنید که در پایان دستور if ، حتما باید end نوشته شود تا نرم افزار متلب بداند که دستور if پایان یافته است.

➢ دستور if به همراه: else

همان طور که گفتیم، زمانی که از دستور if در متلب استفاده می کنیم، متلب شرط یا شرط هایی را چک می کند و در صورت برآورده شدن آنها، دستورات را اجرا می کند. اما شاید بخواهیم به متلب اعلام کنیم که اگر

شرط یا شرط ها برآورده نشدند، آنگاه چه دستوراتی را اجرا کند. در اینگونه موارد، دستور if را با else به کار می بریم. به مثال زیر توجه کنید:

مثال :

در مثال قبلی، از دو دستور if استفاده کردیم، اما این بار همان مثال را تنها با یک دستور if می نویسیم:

```
A=۰
if A>=۰
    B=A
else
    B=-A
end
```

نتیجه :

```
A =
     ۰

B =
     ۰
```

هدف این بوده است که B برابر قدرمطلق A باشد، ابتدا متلب چک می کند که A مساوی یا بزرگتر از صفر هست یا نه، اگر باشد، آنگاه B را برابر A قرار می دهد و چون شرط برآورده شده است، دستورات نوشته شده برای else را نادیده می گیرد. اما اگر A مساوی یا بزرگتر از صفر نباشد، آنگاه متلب، تنها دستورات مربوط به elseرا اجرا می کند.

➢ دستور if به همراه: elseif

گاهی نیاز داریم که چندین شرط به صورت پی در پی چک شوند، اگر اولین شرط صحیح بود، دستورات مربوط به آن اجرا شوند و دستورات مربوط به سایر شرط ها نادیده گرفته شوند، اما اگر شرط اول برآورده نشده بود، شرط دوم چک شود و در صورت برآورده شدن شرط دوم، دستورات مربوط به آن اجرا شود و دستورات مربوط به شرط های باقیمانده نادیده گرفته شود، در صورت برآورده نشدن شرط دوم، آنگاه شرط سوم چک شود و همین طور تا آخر. در اینگونه موارد باید از دستور if به همراه elseif استفاده کنیم. به مثال زیر توجه کنید:

مثال :

همان مثال قبل را این بار با استفاده از elseif می نویسیم. تنها تفاوت این است که حالت خاص A=۰ را جداگانه بررسی کرده ایم:

```
A=۰
if A>۰
    B=A
elseif A==۰
    B=۰
else
    B=-A
end
```

نتیجه :

```
A =
     ۰

B =
     ۰
```

دقت شود که برای چک کردن شرط تساوی، حتما باید از دو علامت تساوی به صورت == استفاده شود، زیرا علامت = در متلب، برای نسبت دادن مقدار به متغیرها در نظر گرفته شده است و بنابراین برای چک کردن شرط تساوی، مجبوریم از علامت == استفاده کنیم.

ساخت حلقه در متلب با for

گاهی اوقات، تعدادی دستور داریم که باید به دفعات زیاد اجرا شوند، اگر بخواهیم به صورت معمولی آنها را بنویسیم، مجبور می شویم کدهای مربوط به آنها را به دفعات زیاد تکرار کنیم، اما انتخاب مناسب برای اجرای دستورات تکراری، ساخت یک حلقه می باشد. در متلب، ساده ترین روش برای ساخت حلقه، استفاده از for می باشد. در مثال زیر، نحوه استفاده از for را برای ساخت یک حلقه شرح داده ایم:

مثال :

فرض کنید بخواهیم حاصل ۱۰*۹*۸*۷*٦*٥*٤*۳*۲*۱=۱۰! را با نرم افزار متلب محاسبه کنیم، برای این منظور، کدهای زیر را می نویسیم:

```
k=۱;
for m=۲:۱۰
    k=k*m;
end
k
```

نتیجه :

```
k =
     ۳٦۲۸۸۰۰
```

در کدهای فوق، مشخص کرده ایم که مقدار m از ۲ تا ۱۰ باید باشد و در هر بار اجرای دستورات حلقه، mبرابر مقدار یکی از اعداد این بازه خواهد بود (۲ و ۳ و ... و ۹ و ۱۰). ابتدا مقدار k را، قبل از شروع حلقه، برابر ۱ تعریف کرده ایم. سپس حلقه for شروع می شود. ابتدا مقدار m برابر ۲ ، که اولین عدد است، قرار داده می شود، kدر m=۲ ضرب می شود و چون دستورات حلقه تمام شده است، مقدار بعدی m=۳ در نظر گرفته می شود و این بار مقدار جدید k در m=۳ ضرب می شود و همین طور این روند ادامه می یابد تا زمانی که m=۱۰ نیز در مقدار جدید k ضرب شود و چون دیگر مقدار جدیدی برای m وجود ندارد، حلقه پایان می یابد و در آخر مقدار k نمایش داده می شود.

➢ خارج شدن از حلقه for و while با دستور break

زمانی که یک حلقه می سازیم با دستور for یاwhile ، تعداد گام های حلقه مشخص است و به همان تعداد، دستورات حلقه اجرا می شوند. اما گاهی نیاز داریم که با به وجود آمدن یک شرایط خاص، تعیین کنیم که گام های باقیمانده حلقه، اجرا نشوند. برای این منظور، دستور break به کار می رود، یعنی وقتی متلب به دستور breakدر حلقه برسد، از حلقه خارج می شود و دیگر هیچ کدی از حلقه اجرا نخواهد شد.

اکنون می خوایم نحوه استفاده از دستور break در حلقه را شرح بدهیم. ابتدا فرض کنید که یک حلقه را با دستور for و به صورت معمولی بسازیم (این حلقه، تنها مقدار پارامتر حلقه را در خروجی چاپ می کند)

```
clear all
close all
clc

for nn=۱:۷
    disp(nn);
end
```

نتیجه :

```
۱
۲
۳
٤
٥
٦
۷
```

اکنون می خواهیم تعیین کنیم که گام سوم به بعد از حلقه، اجرا نشود. بنابراین با دستور if چک می کنیم که هرگاه مقدار پارامتر حلقه (nn) برابر ۳ بود، آنگاه دستور break اجرا شود تا از حلقه خارج شویم و دیگر هیچ کدی از حلقه اجرا نگردد:

```
clear all
close all
clc

for nn=۱:۷
    if(nn==۳)
        break;
    end
    disp(nn);
end
```

نتیجه :

```
۱
۲
```

نکته :

ممکن است این سوال پیش بیاید که اگر دو حلقه تو در تو داشته باشیم و آنگاه درون حلقه داخلی تر، دستور breakاجرا شود، آیا تنها از حلقه داخلی تر خارج می شویم و یا اینکه از هر دو حلقه خارج خواهیم شد. پاسخ این است که تنها از حلقه داخلی تر خارج خواهیم شد.

ساخت تابع در متلب با استفاده از: m-file

معمولا برنامه نویسان حرفه ای، چارچوب برنامه را در یک فایل اصلی می نویسند و بخش های مختلف برنامه را به صورت تابع هایی می نویسند که یک یا چند ورودی را دریافت کرده و محاسبات لازم را انجام می دهند و سپس یک یا چند خروجی را بر می گردانند. برنامه نویس، عملکرد هر تابع را جداگانه چک می کند و سپس زمانی که از بابت آنها خیالش راحت شد، تمرکز اصلی خود را بر روی فایل اصلی که حاوی چارچوب برنامه می باشد، می گذارد و تنها در فایل اصلی، ارجاعاتی به توابع ساخته شده خواهد داد.
چنانچه شما هم از این شیوه استفاده کنید، پس از مدتی صاحب بانکی از توابع خواهید شد که در نوشتن برنامه های جدید، بسیار به شما کمک می کند.
با مثالی ساده، نحوه ساخت یک تابع با استفاده از m-file را شرح می دهیم:

مثال :

می خواهیم تابعی به نام add بسازیم که در ورودی، دو عدد را دریافت کرده و سپس مجموع آن دو عدد را در خروجی برگرداند. ابتدا یک m-file خالی باز می کنیم و کد زیر را در آن می نویسیم:

```
function [z] = add(x,y)
z=x+y;
```

در کد متلب بالا، کلمه function اعلام می کند که این m-file به صورت یک تابع می باشد. خروجی با نام z و دو ورودی با نام های x و y مشخص شده اند. کلمه ای که بعد از علامت تساوی قرار می گیرد (در اینجا کلمه add) ، نام تابع را مشخص می کند و باید در مرحله بعد، m-fileرا دقیقا با همین نام ذخیره کنیم. بنابراین m-fileرا با نام add ذخیره می کنیم.

همان طور که مشاهده کردید، خط اول، ساختار تابع را مشخص می کند و در خط های بعدی، تنها کاری که باید انجام دهیم، این است که روابط بین ورودی ها و خروجی را تعریف کنیم که در این مثال، تنها یک جمع ساده می باشد.

پس از ذخیره کردن m-file ، دیگر کاری با آن نداریم. باید دقت کنید که اگر بخواهید این m-file را با گزینه Save and runدر بالای پنجره، اجرا کنید، با یک خطا مواجه می شوید، زیرا این m-file یک تابع است و باید ورودی هایی برای آن تعریف کنیم.

اکنون باید تابع ساخته شده را تست کنیم. دستور زیر را در پنجره Command می نویسیم (و یا در یک m-file دیگر)

```
B=۲;
C=۳;
A=add(B,C)
```

نتیجه :

```
A =
     ٥
```

نکته :

باید m-file تابع درون Current folder و یا سایر مسیرهای مشخص شده برای نرم افزار متلب باشد. در غیر این صورت، متلب یک پیام خطا را نمایش می دهد. این پیام خطا، به این دلیل است که متلب نمی تواند m-file با نام add را پیدا کند.

نکته :

در دستور فوق مشاهده کردیم که نام هایی که برای ورودی و خروجی تابع add استفاده کردیم (Aو B وC)، با نام های درون خود کدهای تابع add ، متفاوت است (xو y وz) .بنابراین احتیاجی نیست که نگران نام های به کار رفته در ساختار تابع باشید و از هر نام دلخواهی برای ارجاع به تابع می توانید استفاده کنید.

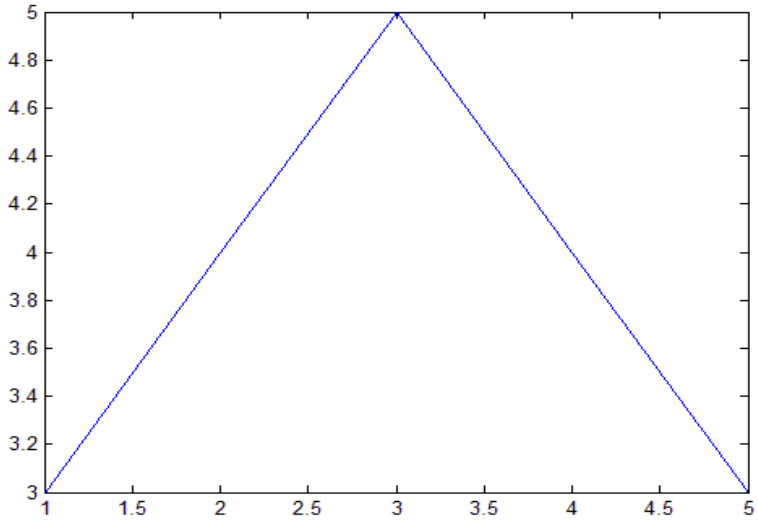
ترسیم شکل در متلب با دستور: plot

دستور plot ، بردارهایی از اعداد را دریافت کرده و آنها را به صورت شکل ترسیم می کند. در واقع دستور plot مقادیر گسسته را که هر کدام به صورت یک نقطه می باشند، پشت سرهم قرار می دهد و سپس آنها را با خط به هم وصل می کند تا بتوانیم آنها را به صورت یک شکل پیوسته ببینیم و بدین ترتیب به ارتباط کلی آنها پی ببریم. به مثال زیر توجه کنید:

مثال :

```
x=[۱ ۲ ۳ ٤ ٥]
y=[۳ ٤ ٥ ٤ ۳]
plot(x,y)
```

نتیجه :

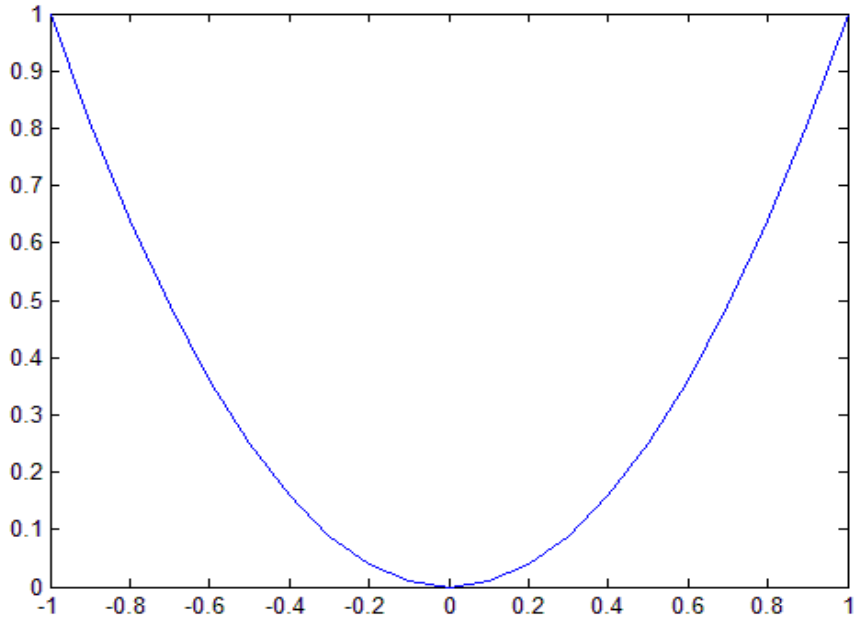


حال فرض کنید بخواهیم تابعی بر حسب متغیر x را برای بازه ای از تغییرات x رسم کنیم، ابتدا باید متغیر x را به صورت برداری از نقاط آن بازه تعریف کنیم. بازه مورد نظر ما پیوسته است و شامل تعداد بینهایت عدد می باشد، اما ما باید تعداد نقاط را به گونه ای انتخاب کنیم که حداقل تعدادی باشند که شکل تابع را به خوبی نمایش بدهند. به مثال زیر توجه کنید:

مثال :

```
x=-۱:۰,۱:۱;
plot(x,x.^۲)
```

نتیجه :



یک علامت نقطه (.) نوشته شده است. وجود این x در دستور فوق، دقت کنید که در تعریف تابع، پس از به x باید به توان ۲ برسد، نه این که کل بردار x علامت، ضروری است و مشخص می کند که هر عنصر بردار .توان ۲ برسد

➢ تعیین عنوان برای محورهای عمودی و افقی شکل با دستور xlabel و دستور: ylabel

همان طور که می دانید، در نرم افزار متلب، دستورات مختلفی همچون plot برای ترسیم شکل و منحنی ها به کار می روند. این دستورات، به محورهای افقی و عمودی، یا عنوان اختصاص نمی دهند و یا اگر عنوان اختصاص بدهند، ممکن است آن عنوان، مد نظر ما نبوده باشد. ممکن است بخواهیم در شکل ترسیم شده، توسط این دستورات، محورهای افقی و عمودی دارای عنوان خاصی باشند. برای این منظور، در متلب از دو دستور xlabel و ylabel استفاده می شود.

از دستور xlabel در متلب، برای تعیین یک عنوان برای محور افقی شکل استفاده می شود و همچنین دستور ylabelنیز برای تعیین یک عنوان برای محور عمودی شکل به کار می رود. برای آشنایی با نحوه استفاده از دو دستور xlabel و ylabel ، به مثال زیر توجه کنید:

مثال :

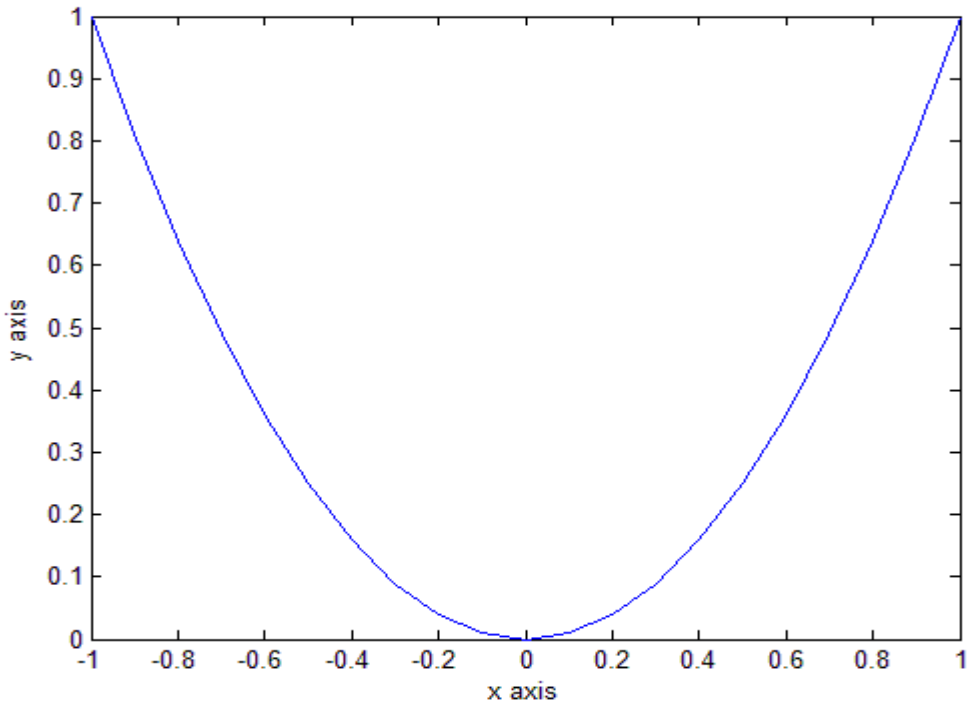
```
x=-۱:۰,۱:۱;
plot(x,x.^۲)
```

نتیجه :

حال با دستور xlabel و دستور ylabel ، برای هر دو محور شکل، عنوان تعیین می کنیم:

```
x=-۱:۰,۱:۱;
plot(x,x.^۲)
xlabel('x axis')
ylabel('y axis')
```

نتیجه :



مشاهده می کنید که دستور xlabel('x axis') تعیین کرده است که محور افقی دارای عنوان x axis باشد و دستور ylabel('y axis') نیز تعیین کرده است که محور عمودی دارای عنوان y axis باشد.

➢ شیوه های مختلف نمایش خطوط منحنی برای دستور plot در متلب:

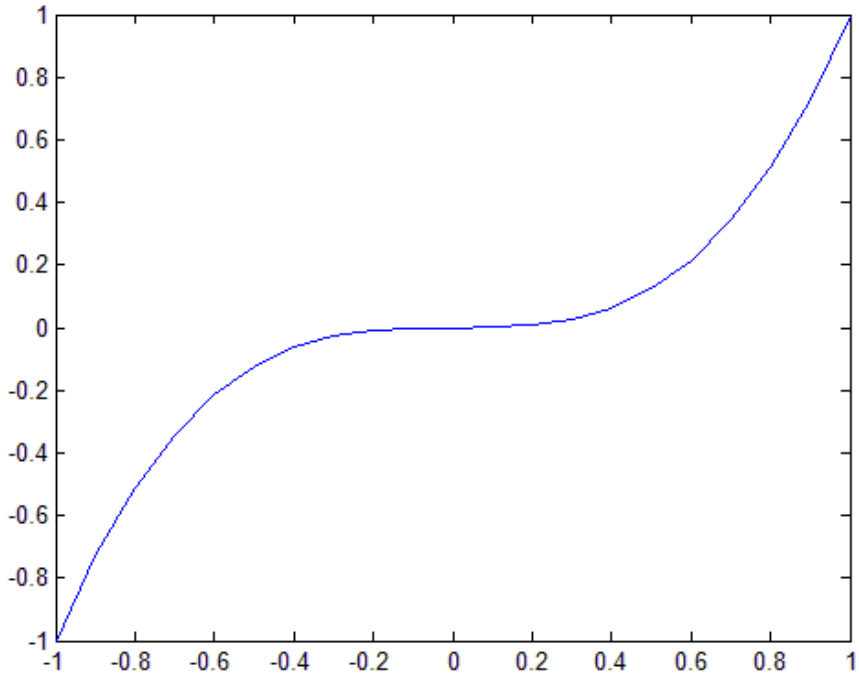
همان طور که می دانید، دستور plot در متلب، برای ترسیم منحنی ها به کار می رود. این دستور برای نحوه نمایش خطوط منحنی ها، دارای یک پیش فرض می باشد، اما می توان این پیش فرض را تغییر داد تا خطوط منحنی ها، با اشکال و شیوه های دیگری نمایش داده شوند. برای این کار، به دستور جدیدی نیاز نیست و تنها باید درون پرانتز دستور plot ، عبارت مربوط به شیوه نمایش مورد نظرمان را بنویسیم. به مثال زیر توجه کنید:

مثال :

فرض کنید بخواهیم تابع زیر را در بازه [-۱،۱] با دستور plot رسم کنیم برای این منظور، می نویسیم:

```
x=-۱:۰,۱:۱;
plot(x,x.^۳)
```

نتیجه :



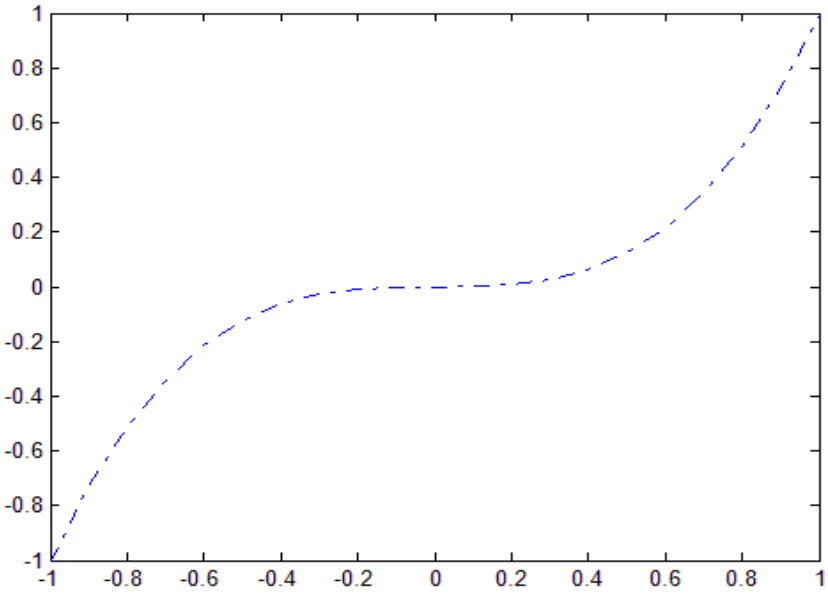
این شیوه نمایش منحنی، شیوه نمایش پیش فرض برای دستور plot می باشد. حال شیوه های دیگر را به کار می بریم:

نمایش خطوط به صورت خطوط نقطه-خط چین: (dash-dot line)

برای نمایش خطوط به صورت خطوط نقطه-خط چین(dash-dot line) ، باید عبارت '-.' را درون پرانتز دستور plot بنویسیم:

```
x=-۱:۰,۱:۱;
plot(x,x.^۳,'-.')
```

نتیجه :



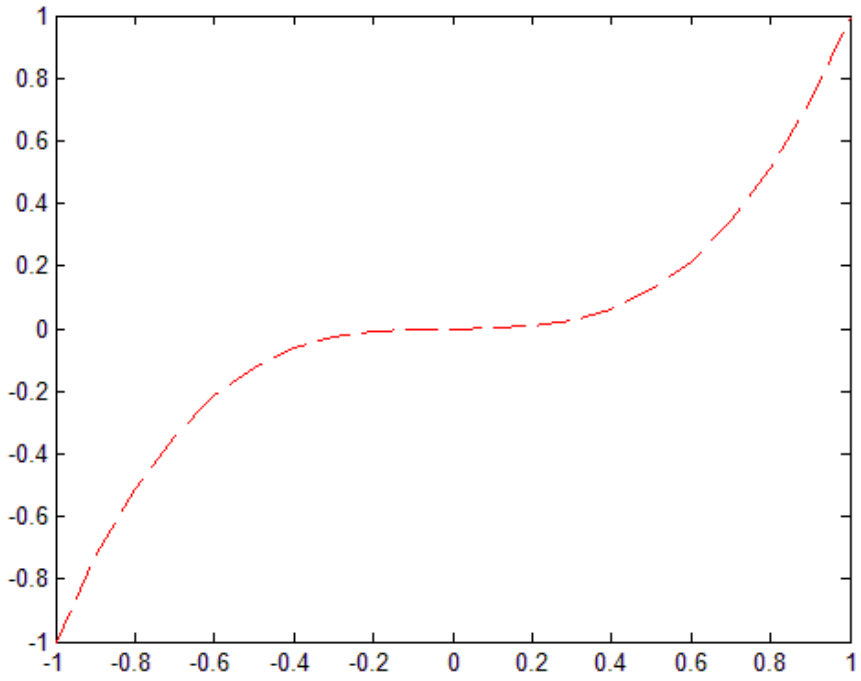
مشاهده می کنید که خطوط منحنی به صورت خطوط نقطه-خط چین (dash-dot line) نمایش داده شده است.

نمایش خطوط به صورت خطوط خط چین قرمز: (red dashed line)

برای نمایش خطوط به صورت خطوط خط چین قرمز(red dashed line) ، باید عبارت'r--' را درون پرانتز دستور plot بنویسیم:

```
x=-۱:۰,۱:۱;
plot(x,x.^۳,'r--')
```

نتیجه :



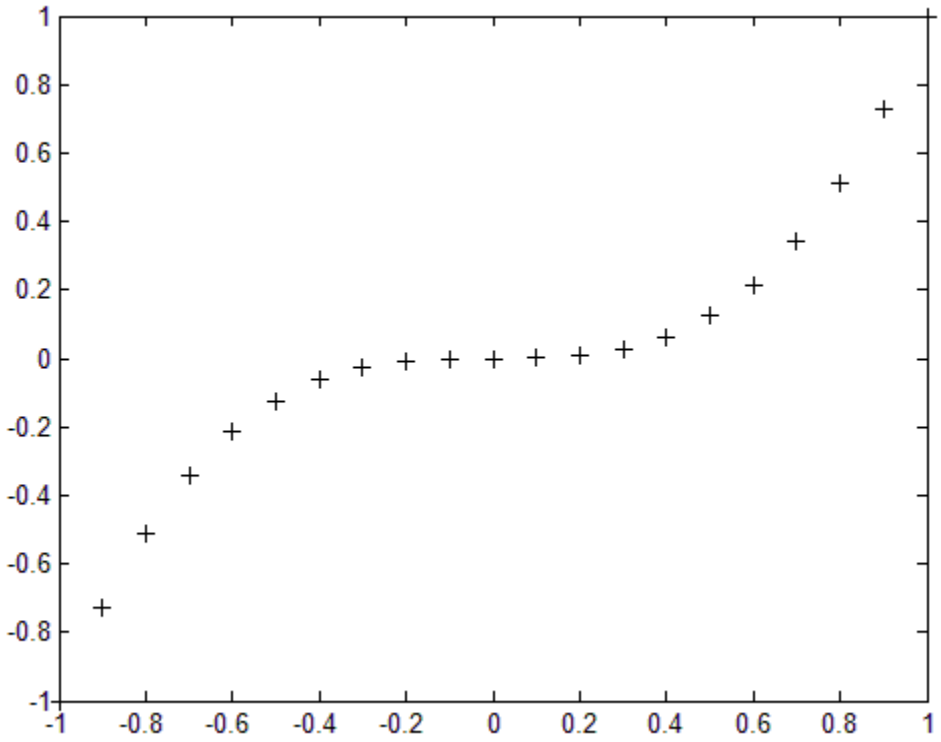
مشاهده می کنید که خطوط منحنی به صورت خطوط خط چین قرمز (red dashed line) نمایش داده شده است.

نمایش خطوط به صورت خطوط با علامت های مثبت سیاه (black plus signs) :

برای نمایش خطوط به صورت خطوط با علامت های مثبت سیاه(black plus signs) ، باید عبارت 'k+' را درون پرانتز دستور plot بنویسیم:

```
x=-۱:۰,۱:۱;
plot(x,x.^۳,'k+')
```

نتیجه :



مشاهده می کنید که خطوط منحنی به صورت خطوط با علامت های مثبت سیاه (black plus signs) نمایش داده شده است.