فصل اول

مقدمه ای بر MATLAB

®MATLAB یک بسته نرم افزاری بسیار قدرتمند است که دارای ابزارهای توکار بسیاری برای حل مشکلات و توسعه تصاویر گرافیکی است. ساده ترین روش برای استفاده از محصول MATLAB به صورت تعاملی است. یک عبارت توسط کاربر وارد می شود و MATLAB بلافاصله با نتیجه پاسخ می دهد. همچنین امکان نوشتن اسکریپت ها (متن ها) و برنامه ها در MATLAB وجود دارد که در اصل گروهی از دستورات هستند که به صورت متوالی اجرا می شوند.

این فصل بر روی اصول اولیه، از جمله بسیاری از عملگرها و توابع توکار که می‌توانند در عبارات تعاملی استفاده شوند، تمرکز خواهد کرد.

1.1 ورود به MATLAB

MATLAB یک بسته نرم افزاری ریاضی و گرافیکی با قابلیت های عددی، گرافیکی و برنامه نویسی است. این شامل یک محیط توسعه یکپارچه، و همچنین ساختارهای برنامه نویسی رویه ای و شی‌گرا است. MATLAB دارای توابع توکار برای انجام بسیاری از عملیات، و جعبه‌ابزارهایی وجود دارند که می‌توانند برای تقویت این توابع (به عنوان مثال برای پردازش سیگنال) اضافه شوند. نسخه های موجود برای پلتفرم های سخت افزاری مختلف، هم در نسخه‌های حرفه‌ای و هم در نسخه‌های دانشجویی وجود دارد. MathWorks سالانه دو نسخه از MATLAB را به نام‌های سال مدنظر و a یا b منتشر می‌کند. این کتاب نسخه‌های منتشر شده از طریق نسخه R2018a را پوشش می دهد. در مواردی که در سال‌های اخیر تغییراتی رخ داده است، به این موارد اشاره می شود.

هنگامی که نرم افزار MATLAB راه اندازی می شود، پنجره ای باز می شود که قسمت اصلی آن پنجره فرمان است (شکل 1.1 را ببینید). در پنجره فرمان، باید علامت زیر را ببینید:

>>

<< اعلان نامیده می شود. در نسخه دانشجویی، اعلان به شکل زیر است:

EDU>>

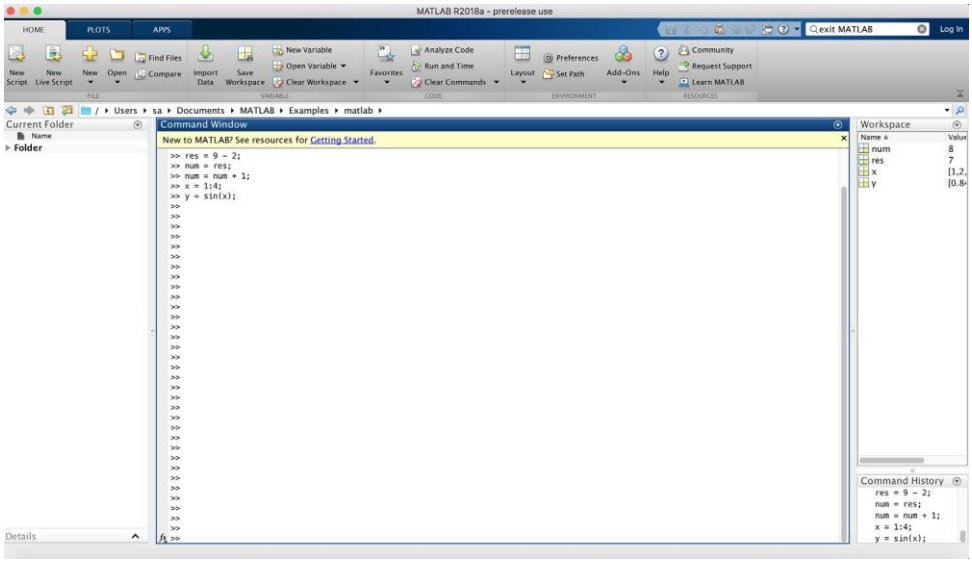
در Command Window می توان از MATLAB به صورت تعاملی استفاده کرد. در اعلان، هر دستور یا عبارت MATLAB را می توان وارد کرد، و MATLAB بلافاصله با نتیجه پاسخ خواهد داد.

همچنین امکان نوشتن برنامه هایی در MATLAB وجود دارد که در فایل‌های اسکریپت (متنی) یا فایل‌های کد MATLAB قرار دارند. برنامه ها در فصل 3 معرفی خواهند شد.

دستورات زیر می توانند به عنوان مقدمه ای برای MATLAB عمل کنند و به شما کمک کنند:

- demo مثال‌هایی از MATLAB را در مرورگر راهنما، که نمونه هایی از برخی از ویژگی های MATLAB را دارد، نمایش می دهد
- help هر تابعی را توضیح می دهد. help help توضیح خواهد داد که چگونه help کار می‌کند
- lookfor از طریق help یک کلمه یا عبارت خاص را جستجو می کند (توجه: این ممکن است زمان زیادی طول بکشد)
- doc یک صفحه مستندات را در مرورگر راهنما باز می کند

برای خروج از MATLAB، عبارت quit یا exit را تایپ کنید یا بر روی علامت قرمز رنگ "x" در بالای پنجره کلیک کنید.



شکل 1.1

MATLAB command window

1.2 محیط دسکتاپ MATLAB

علاوه بر پنجره فرمان (Command Window) ، چندین پنجره دیگر نیز وجود دارد که می توانند باز شوند و ممکن است به طور پیش فرض باز شوند. آنچه در اینجا توضیح داده شده است، طرح پیش فرض این ویندوزها در نسخه R2018a است، اگرچه تنظیمات احتمالی دیگری نیز وجود دارد. نسخه‌های مختلف MATLAB ممکن است تنظیمات دیگری را به‌طور پیش‌فرض نشان دهند و طرح‌بندی همیشه می‌تواند سفارشی شود. بنابراین، ویژگی های اصلی در اینجا به اختصار توضیح داده می شود.

در سمت چپ پنجره فرمان، پنجره پوشه فعلی قرار دارد. پوشه ای که به عنوان پوشه فعلی تنظیم شده است جایی است که فایل ها ذخیره می شوند. این پنجره فایل‌های ذخیره شده در پوشه فعلی را نشان می دهد. این‌ها را می توان به روش‌های مختلفی گروه بندی کرد، به عنوان مثال بر اساس نوع، و به عنوان مثال بر اساس نام طبقه‌بندی کرد. اگر فایلی انتخاب شده باشد، اطلاعات مربوط به آن فایل در پایین نشان داده می‌شود که روی آن «جزئیات» نوشته شده است.

در سمت راست پنجره Command، پنجره فضای کار (Workspace) در بالا و پنجره تاریخچه دستور (Command History Window) در پایین قرار دارد. پنجره تاریخچه فرمان (دستور) فرمان (دستور)هایی را نشان می‌دهد که نه فقط در جلسه فعلی (در پنجره دستور فعلی)، بلکه قبلاً نیز وارد شده‌اند. پنجره Workspace در بخش بعدی توضیح داده خواهد شد.

این پیکربندی پیش فرض را می توان با کلیک کردن بر روی فلش رو به پایین در گوشه سمت راست بالای هر پنجره تغییر داد. این منویی از گزینه ها را نشان می دهد (برای هر پنجره متفاوت است)، از جمله، برای مثال، بستن آن پنجره خاص و باز کردن آن پنجره. پس از باز کردن، با باز کردن منو و سپس کلیک کردن بر روی فلش پیچ خورده که به سمت راست پایین اشاره می کند، پنجره دوباره متصل می شود. برای تبدیل هر یک از این پنجره ها به پنجره فعال، روی ماوس در آن کلیک کنید. به طور پیش فرض، پنجره فعال پنجره فرمان است.

دسکتاپ یک نوار ابزار دارد. به‌طور پیش‌فرض، سه برگه نشان داده می‌شود («HOME»، «PLOTS»، و «APPS»)، اگرچه دیگری، «SHORTCUTS» را می‌توان اضافه کرد. در زیر برگه "HOME"، بسیاری از ویژگی های مفید وجود دارد که به بخش های کاربردی "file"، "variable"، "code"، "ENVIRONMENT " و "RESOURCES " تقسیم می شوند (این برچسب ها در پایین صفحه قابل مشاهده هستند. ناحیه نوار ابزار خاکستری). به عنوان مثال، در زیر "ENVIRONMENT"، با ضربه زدن به فلش رو به پایین در زیر Layout امکان سفارشی سازی پنجره های داخل Desktop Environment وجود دارد. سایر ویژگی‌های نوار ابزار در فصل‌های بعدی با توضیح مطالب مربوطه معرفی خواهند شد.

1.3 متغیرها و دستورهای انتساب

برای ذخیره یک مقدار در یک جلسه MATLAB یا در یک برنامه، از یک متغیر استفاده می شود. پنجره Workspace متغیرهای ایجاد شده و مقادیر آنها را نشان می دهد. یک راه آسان برای ایجاد یک متغیر، استفاده از دستور انتساب است. ساختار یک دستور انتساب به شکل زیر است:

```
variablename = expression
```

متغیر همیشه در سمت چپ است و پس از آن علامت = که عملگر انتساب است (بر خلاف ریاضیات، علامت مساوی منفرد به معنای برابری نیست) و به دنبال آن یک عبارت است. عبارت مورد ارزیابی قرار می‌گیرد و سپس مقدار آن در متغیر ذخیره می شود. در اینجا یک مثال و نحوه نمایش آن پس از فشردن کلید Enter در پنجره فرمان آورده شده است:

```
>> mynum = 6
mynum =
6
>>
```

در اینجا کاربر (شخصی که در MATLAB کار می کند) "mynum = 6"را در قسمت اعلان (prompt) تایپ می‌کند و MATLAB عدد صحیح 6 را در متغیری به نام mynum ذخیره می کند و سپس نتیجه را دوباره نمایش می دهد. از آنجا که علامت مساوی عملگر انتساب است و به معنای برابری نیست، عبارت او باید به صورت "mynum مقدار 6 را دریافت می‌کند" خوانده شود (نه "mynum برابر است با 6").

توجه داشته باشید که نام متغیر همیشه باید در سمت چپ و عبارت در سمت راست باشد. اگر اینها معکوس شوند، خطا رخ می دهد.

```
>> 6 = mynum
```

```
6 = mynum

Error: Incorrect use of '=' operator. To assign a value to a variable,
use '='. To compare values for equality, use '=='.

>>
```

قرار دادن نقطه ویرگول (سمی کالون) در انتهای یک عبارت، از نمایش نتیجه در خروجی جلوگیری می کند. مثلا،

```
>> res = 9 – 2;
>>
```

با این کار نتیجه عبارت سمت راست یعنی مقدار 7 به متغیر res اختصاص می یابد. اما نتیجه را نشان نمی دهد. در عوض، یک اعلان دیگر بلافاصله ظاهر می شود. با این وجود، در این نقطه و پس از دستورهایی که از ابتدا تاکنون نوشته ایم، از پنجره فضای کاری هر دو متغیر mynum و res و مقادیر آنها قابل مشاهده هستند.

توجه داشته باشید که در ادامه کتاب، دستوری که بعد از نتیجه ظاهر می شود نشان داده نخواهد شد.

فاصله‌های موجود در یک دستور یا عبارت بر نتیجه تأثیر نمی گذارد، اما خواندن آن را آسان‌تر می کند. دستور زیر که فاقد فاصله است، دقیقاً همان نتیجه عبارت قبلی را به دست می آورد:

```
>> res = 9–2;
```

MATLAB از یک متغیر پیش فرض به نام ans استفاده می کند اگر عبارتی در دستور تایپ شود و به متغیری اختصاص داده نشود. به عنوان مثال، نتیجه عبارت 3 + 6 در متغیر ans ذخیره می شود:

```
>> 6 + 3
ans =
9
```

این متغیر پیش‌فرض، ans، هر زمانی که فقط یک عبارت، نه یک دستور انتساب، در اعلان تایپ شود، دوباره استفاده می‌شود. توجه داشته باشید که به همین دلیل استفاده از ans به عنوان نام یا به عنوان بخشی از یک عبارات ایده خوبی نیست.

یک میانبر برای تایپ مجدد دستورات ضربه زدن به فلش رو به بالا است که به دستور(های) قبلاً تایپ شده برمی گردد. به عنوان مثال، اگر تصمیم دارید به جای استفاده از متغیر پیش‌فرض نتیجه عبارت 3 + 6 را به متغیری با نام result اختصاص دهید، می‌توانید به جای تایپ مجدد کل عبارت، فلش بالا و سپس فلش چپ را بزنید تا دستور را تغییر دهید:

```
>> result =  6 + 3

result  =

9
```

این بسیار مفید است، به خصوص اگر یک عبارت طولانی وارد شود و حاوی یک خطا باشد و برای تصحیح آن می خواهید به عقب برگردید.

1.3.1 مقدار دهی اولیه، افزایش و کاهش

همانطور که قبلا نشان داده شده است، اغلب، مقادیر متغیرها تغییر می‌کند. قرار دادن مقدار اولیه در یک متغیر را مقداردهی اولیه متغیر می نامند. افزودن به متغیر را افزایش می گویند. به عنوان مثال، دستور

```
mynum = mynum + 1
```

مقدار متغیر mynum را 1 افزایش می دهد.

سوال سریع!

چگونه می توان 1 را از مقدار متغیری به نام num کم کرد؟

پاسخ: `num = num – 1;`

1.3.2 نام متغیرها

نام متغیرها نمونه ای از نام های شناسه (مشخص کننده) هستند. نمونه های دیگری از نام های شناسه مانند نام توابع را در فصل های آینده خواهیم دید. قوانین مربوط به نام های شناسه به شرح زیر است.

- نام باید با یک حرف الفبا شروع شود. پس از آن، نام می تواند شامل حروف، اعداد و نویسه زیرخط (به عنوان مثال، value_1) باشد، اما نمی تواند فاصله داشته باشد.
- محدودیتی برای طول نام وجود دارد. تابع توکار namelengthmax می‌گوید که این حداکثر چقدر است (هر کاراکتر اضافی کوتاه می شود).
- MATLAB به حروف بزرگ و کوچک حساس است، به این معنی که بین حروف بزرگ و کوچک تفاوت وجود دارد. بنابراین، متغیرهایی به نامmynum ، MYNUM و Mynum همه متفاوت هستند (اگرچه این امر گیج کننده است و نباید انجام شود).
- اگرچه کاراکترهای خط زیر (underline) در یک نام معتبر هستند، اما استفاده از آنها می تواند در برخی از برنامه هایی که با MATLAB تعامل دارند، مشکل ایجاد کند، بنابراین برخی از برنامه نویسان به جای آن از حروف مختلط استفاده می کنند (به عنوان مثال، partWeights به جای part_weights)
- کلمات خاصی به نام کلمات رزرو شده یا کلمات کلیدی وجود دارد که نمی توان از آنها به عنوان نام متغیر استفاده کرد.
- نام توابع توکار (توضیح داده شده در بخش بعدی) می‌تواند، اما نباید به عنوان نام متغیر استفاده شود. علاوه بر این، نام متغیرها باید همیشه مرتبط و به یادماندنی باشد، به این معنی که آنها باید تا حدودی معنی داشته باشند. به عنوان مثال، اگر متغیر شعاع یک دایره را ذخیره می کند، نامی مانند radius منطقی خواهد بود در حالی که نام x احتمالا این کار را نمی کند.

دستورات زیر مربوط به متغیرها هستند:

- who متغیرهایی را نشان می دهد که در این پنجره فرمان تعریف شده اند (این دستور فقط نام متغیرها را نشان می دهد)
- whos متغیرهایی را نشان می دهد که در این پنجره فرمان تعریف شده اند (این دستور اطلاعات بیشتری در مورد متغیرها، مشابه آنچه در پنجره Workspace وجود دارد نشان می دهد.)
- clearvars همه متغیرها را پاک می کند تا دیگر وجود نداشته باشند clearvars variablename نام یک متغیر خاص به نام variablename را پاک می‌کند
- clearvars variablename1 variablename2 ... لیستی از متغیرها را پاک می کند (توجه: نام ها را با فاصله جدا کنید، نه با کاما)
- clear شبیه به clearvars است، اما توابع را نیز پاک می‌کند. به عنوان مثال، در ابتدای جلسه MATLAB ، متغیرها می توانند ایجاد شوند و سپس به صورت انتخابی پاک شوند (به یاد داشته باشید که نقطه ویرگول از نمایش خروجی جلوگیری می کند).

```
>> who

>> mynum = 3;

>> mynum + 5;

>> who

Your variables are:

ans mynum

>> clear mynum

>> who

Your variables are:

Ans
```

این تغییرات در پنجره Workspace نیز قابل مشاهده است.

1.3.3 نوع ها (types)

هر متغیر دارای یک نوع مرتبط با آن است. MATLAB از انواع مختلفی پشتیبانی می کند که به آنها کلاس گفته می شود. (در اصل، یک کلاس ترکیبی از یک نوع و عملیاتی است که می‌توان روی مقادیر آن نوع انجام داد، اما برای سادگی، فعلاً از این اصطلاحات به جای یکدیگر استفاده می‌کنیم. مطالب بیشتر در مورد کلاس‌ها در فصل 11 پوشش داده خواهد شد.)

به عنوان مثال، انواعی برای ذخیره انواع مختلف اعداد وجود دارد. برای اعداد شناور یا حقیقی، یا به عبارت دیگر اعداد با رقم اعشار (به عنوان مثال، 5.3)، دو نوع اساسی (پایه) وجود دارند: single و double. نام نوع double کوتاه شده دقت مضاعف (double precision) است. این نوع اعداد بزرگتر از نوع single را ذخیره می کند. MATLAB برای این اعداد از نمایش ممیز شناور استفاده می کند.

انواع اعداد صحیح زیادی در MATLAB وجود دارد، مانند int8، int16، int32 و int64. اعداد موجود در نام ها تعداد بیت های مورد استفاده برای ذخیره مقادیر آن نوع را نشان می دهد. به عنوان مثال، نوع int8 در مجموع از هشت بیت برای ذخیره عدد صحیح و علامت آن استفاده می کند. از آنجایی که یک بیت برای علامت استفاده می شود، لذا از هفت بیت برای ذخیره اعداد 0 یا 1 استفاده می شود. انواع عدد صحیح بدون علامت uint8، uint16، uint32 و uint64 نیز وجود دارند. برای این نوع ها، علامت ذخیره نمی شود، به این معنی که عدد صحیح فقط می تواند مثبت (یا 0) باشد.

هر چه عدد در نام نوع بزرگتر باشد، عدد بزرگتری می تواند در آن ذخیره شود. در بیشتر موارد از نوع int32 زمانی که نیاز به نوع عدد صحیح باشد استفاده خواهیم کرد.

نوع char برای ذخیره کراکتر (نویسه)های single (مانند 'x') یا بردارهای کاراکتر، که دنباله‌ای از کاراکترها هستند (مانند'cat') استفاده می‌شود. هم کاراکترها و هم بردار کاراکترها در بین علامت نقل قول تکی (single quote) قرار می گیرند.

نوع رشته string برای ذخیره رشته ها استفاده می شود (به عنوان مثال، "hello"). رشته ها در علامت نقل قول‌های دوگانه (double quotes) قرار می‌گیرند. استفاده از نقل قول‌های دوگانه در R2017a معرفی شد.

نوع منطقی برای ذخیره مقادیر true/false استفاده می شود.

متغیرهایی که در Command Window ایجاد شده‌اند را می توان در پنجره Workspace مشاهده کرد. در آن پنجره، برای هر متغیر، نام، مقدار و کلاس متغیر (که در اصل نوع آن است) قابل مشاهده است. سایر ویژگی های متغیرها را نیز می توان در پنجره Workspace مشاهده کرد. اینکه کدام ویژگی به طور پیش فرض قابل مشاهده است بستگی به نسخه MATLAB دارد. با این حال، هنگامی که پنجره Workspace انتخاب می شود، کلیک کردن بر روی فلش رو به پایین به کاربر این امکان را می دهد که با تغییر Choose Columns انتخاب کند که کدام ویژگی ها نمایش داده شوند.

به طور پیش فرض، اعداد به عنوان نوع double در MATLAB ذخیره می شوند. از تابع class می توان برای مشاهده نوع هر متغیر استفاده کرد:

```
>> num = 3 + 6;

>> class(num)
```

```
ans =

'double'
```

1.4 عبارت های (عبارات) عددی

عبارات را می توان با استفاده از مقادیر، متغیرهایی که قبلا ایجاد شده اند، عملگرها، توابع توکار و پرانتز ایجاد کرد. عبارت های شامل اعداد، می توانند شامل عملگرهایی مانند ضرب و توابعی مانند توابع مثلثاتی باشند. نمونه ای از چنین عبارتی:

```
>> 2 * sin(1.4)

ans =

1.9709
```

1.4.1 دستور format و ellipsis

پیش فرض در MATLAB نمایش اعدادی است که دارای اعشار با چهار رقم اعشار هستند، همانطور که در مثال قبل نشان داده شده است. (به طور پیش فرض به این معنی است که اگر غیر از این را مشخص نکنید، این همان چیزی است که دریافت می کنید.) از دستور format می توان برای تعیین فرمت خروجی عبارات استفاده کرد.

گزینه های زیادی وجود دارد، از جمله تبدیل به قالب کوتاه (پیش فرض) short format یا قالب طولانی long format. به عنوان مثال، تغییر قالب به long منجر به 15 رقم اعشار می‌شود و همانطور که در ادامه نشان داده شده است، تا زمانی که قالب به short تبدیل شود، به قوت خود باقی خواهد ماند.

```
>> format long

>> 2 * sin(1.4)

ans =

1.970899459976920

>> format short

>> 2 * sin(1.4)

ans =

1.9709
```

دستور format همچنین می تواند برای کنترل فاصله بین دستور یا عبارت MATLAB و نتیجه استفاده شود. یعنی می‌توان (پیش فرض) را از دست داد یا فشرده کرد.

```
>> format loose
>> 5*33
ans =
165
>> format compact
>> 5*33
ans =
165
>>
```

به خصوص عبارات طولانی را می توان با تایپ سه (یا بیشتر) نقطه، که عملگر ادامه یا ellipsis است، در خط بعدی ادامه داد. برای انجام این کار، بخشی از عبارت را به دنبال سه نقطه ... تایپ کنید، سپس کلید Enter را بزنید و تایپ عبارت را در خط بعدی ادامه دهید.

```
>> 3 + 55 62 + 4 5  . . .

+ 22 - 1

ans =

16
```

1.4.2 عملگرها

به طور کلی دو نوع عملگر وجود دارد: عملگرهای تکی که بر روی یک مقدار یا عملوند عمل می کنند و عملگرهای دودویی (باینری) که بر روی دو مقدار یا دو عملوند عمل می کنند. برای مثال نماد "-" هم عملگر تکی برای منفی کردن است و هم عملگر دودویی برای تفریق است.

در اینجا برخی از عملگرهای رایجی که می توان با عبارات عددی استفاده کرد آورده شده است:

+ addition

_ negation, subtraction

* multiplication

/ division (divided by e.g., 10/5 is 2)

\ division (divided into e.g., 5\10 is 2)

^ exponentiation (e.g., 5^2 is 25)

علاوه بر نمایش اعداد با اعشار، اعداد را می توان با استفاده از نماد علمی یا نمایی نیز نشان داد. رای این کار از e برای (بجای) توان 10 استفاده می‌شود. به عنوان مثال، 2*10^4 را می توان به دو صورت نوشت:

```
>> 2 * 10^4

ans =

20000

>> 2e4

ans =

20000
```

1.4.2.1 قوانین اولویت عملگرها

برخی از عملگرها نسبت به سایرین تقدم دارند. مثلاً در عبارت 3*5+4 ، ضرب بر جمع تقدم دارد، بنابراین ابتدا 5 در 3 ضرب می شود سپس 4 به نتیجه اضافه می‌شود. استفاده از پرانتز می‌تواند اولویت را در یک عبارت تغییر دهد:

```
>> 4 + 5 * 3

ans =

19

>> (4 + 5) * 3

ans =

27
```

در یک سطح تقدم معین، عبارات از چپ به راست ارزیابی می‌شوند (این را قانون انجمنی می‌گویند).

پرانتزهای تو در تو پرانتزهایی در داخل دیگران هستند. ابتدا عبارت در پرانتز داخلی ارزیابی می شود. برای مثال در عبارت 5 – (6 * (4 +2)) ابتدا جمع و سپس ضرب و در نهایت تفریق انجام می شود تا حاصل 31- شود. همچنین می‌توان از پرانتز برای شفاف‌تر کردن یک عبارت استفاده کرد. به عنوان مثال، در عبارت ((1 - (5*3) + 4)) پرانتزها ضروری نیست، بلکه برای نشان دادن ترتیب ارزیابی اجزای عبارت استفاده می شود.

برای عملگرهایی که تاکنون مطرح شده اند، اولویت زیر (از بالاترین به پایین ترین) وجود دارد:

( ) parentheses

^ exponentiation

_ negation

*, /, \ all multiplication and division

+, _ addition and subtraction

تمرین 1.1

در مورد نتایج عبارات زیر فکر کنید و سپس آنها را تایپ کنید تا پاسخ های خود را تأیید کنید:

1\2

-5^2

(-5) ^2

10 - 6/2

5 * 4/2 * 3

1.4.3 توابع توکار (build-in) و کمک (راهنما) (help)

توابع توکار زیادی در MATLAB وجود دارد. از دستور help می توان برای شناسایی توابع MATLAB و همچنین نحوه استفاده از آنها استفاده کرد. به عنوان مثال، تایپ کردن help در فرمان در پنجره فرمان، فهرستی از موضوعات راهنما را نشان می دهد که گروهی از توابع مرتبط هستند. این یک لیست بسیار طولانی است. ابتدایی ترین موضوعات کمکی در ابتدا ظاهر می شوند. همچنین، اگر جعبه ابزاری نصب کرده باشید، آنها نیز لیست خواهند شد.

به عنوان مثال، یکی از موضوعات کمک اولیه به عنوان matlab/elfun ذکر شده است. شامل توابع ریاضی ابتدایی است. یکی دیگر از اولین موضوعات کمکی matlab/ops است که عملگرهایی را که می توان در عبارات استفاده کرد را نشان می دهد.

برای مشاهده لیستی از توابع موجود در یک موضوع کمکی خاص، help و سپس نام موضوع را تایپ کنید. مثلا،

```
>> help elfun
```

لیستی از توابع ریاضی ابتدایی را نشان می دهد. این یک لیست بسیار طولانی است و شامل توابع مثلثاتی (که پیش فرض آن رادیان است، اما توابع معادلی وجود دارد که در عوض از درجه استفاده می کنند)، توابع نمایی، توابع مربوط به فضای مختلط و توابع گردکردن و باقیمانده می باشد.

برای اینکه بفهمید یک تابع خاص چه کاری انجام می دهد و چگونه آن را فراخوانی کنید، help و سپس نام تابع را تایپ کنید. به عنوان مثال، در زیر توضیحی از تابع sin ارائه می شود.

```
>> help sin
```

توجه داشته باشید که با کلیک بر روی *fx* در سمت چپ علامت آماده سازی ">>" در پنجره فرمان نیز می‌توانید توابع موجود در موضوعات راهنما را مرور کنید. انتخاب دکمه help در زیر Resources برای نمایش صفحه Documentation برای MATLAB روش دیگری برای یافتن توابع بر اساس دسته است.

برای فراخوانی یک تابع، نام تابع به همراه آرگومان(هایی) که، توسط نوشتن نام آنها در داخل پرانتز، برای تابع ارسال می شود، باید نوشته شوند. سپس اکثر توابع مقدار(های) را برمی گردانند. به عنوان مثال، برای یافتن مقدار مطلق 4- ، عبارت زیر وارد می شود:

```
>> abs(-4)
```

که فراخوانی تابع abs است. عدد داخل پرانتز، 4- ، آرگومان است. در نتیجه مقدار 4 برگردانده می شود.

سوال سریع!

اگر از نام یک تابع، به عنوان مثال abs، را به عنوان نام متغیر استفاده کنید، چه اتفاقی می افتد؟

پاسخ: این در MATLAB مجاز است، اما تا زمانی که متغیر پاک نشود (با استفاده از clear یا clearvars) نمی‌توان از abs به عنوان تابع توکار استفاده کرد. به عنوان مثال دنباله زیر را بررسی کنید:

```
>> clearvars
>> abs(-6)

ans =

6

>> abs = 11

abs =

11

>> abs(-6)

>> Index exceeds matrix dimensions.
```

```
>> who
Your variables are:
abs ans
>> clearvars abs
>> who
Your variables are:
ans
>> abs(-6)
ans =
6
```

همه عملگرها دارای یک شکل تابعی هستند. برای نمونه، 5 + 2 را می توان به کمک تابع plus به شکل زیر نوشت:

```
>> plus(2,5)

Ans =    7
```

MATLAB یک میانبر مفید دارد که به آن tab ویژگی تکمیل می گویند. اگر کاراکترهای ابتدایی را در نام یک تابع تایپ کنید و کلید تب را بزنید، لیستی از توابع که با کاراکترهای تایپ شده شروع می شوند ظاهر می شود. خطاهای حروف بزرگ به طور خودکار رفع می شوند. همچنین اگر نام تابعی اشتباه تایپ شده باشد، MATLAB نام صحیحی را پیشنهاد می کند.

```
>> abso(-4)
Undefined function or variable 'abso'.

Did you mean:
>> abs(-4)
```

4.4.1 ثابت

از متغیرها برای ذخیره مقادیری استفاده می‌شود که ممکن است تغییر کنند یا مقادیر آن‌ها از قبل مشخص نیست. اکثر زبان ها همچنین ظرفیت ذخیره سازی ثابت ها را دارند، که مقادیری هستند که از قبل شناخته شده اند و احتمالاً نمی توانند تغییر کنند. مثالی از یک مقدار ثابت pi یا π می‌باشد که 3.14159 گرد شده تا شش رقم با معنای آن می باشد. در MATLAB، توابعی وجود دارند که برخی از این مقادیر ثابت را برمی گرداند که برخی از آنها عبارتند از:

```
>> pi
ans =
   3.1416

>> i
ans =
  0.0000 + 1.0000i

>> j
ans =
  0.0000 + 1.0000i

>> inf
ans =
  Inf

>> nan
ans =
  NaN
```

NaN برای اعلام "غیر عدد" مانند 0/0 ظاهر می شود.

```
>> 0/0
ans =
  NaN
```

5.4.1 اعداد تصادفی (Random numbers)

هنگامی که یک برنامه برای کار با داده ها نوشته می شود و داده ها هنوز در دسترس نیستند، اغلب مفید است که ابتدا برنامه را با مقداردهی اولیه متغیرهای داده به *اعداد تصادفی* آزمایش کنید. اعداد تصادفی در شبیه سازی نیز مفید هستند. چندین تابع داخلی در MATLAB وجود دارد که اعداد تصادفی را تولید می کنند که در این قسمت تعدادی از آنها توضیح داده خواهد شد.

مولدهای یا توابع اعداد تصادفی واقعاً تصادفی نیستند. اساساً روش کار به این صورت است که فرآیند با یک عدد شروع می شود که به آن دانه (seed) می‌گویند. اغلب، دانه اولیه یا یک مقدار از پیش تعیین شده است یا از ساعت داخلی رایانه به دست می آید. سپس، بر اساس این دانه، یک فرآیند "عدد تصادفی" بعدی را تعیین می کند. با استفاده از آن عدد به عنوان دانه دفعه بعد، یک عدد تصادفی دیگر تولید می شود و غیره. اینها در واقع شبه تصادفی نامیده می شوند - آنها واقعاً تصادفی نیستند زیرا فرآیندی وجود دارد که هر بار مقدار بعدی را تعیین می کند.
تابع rand می تواند برای تولید اعداد واقعی تصادفی با توزیع یکنواخت استفاده شود. با فراخوانی آن، یک عدد واقعی تصادفی در بازه باز (0,1) ایجاد می شود، که به این معنی است که نقاط انتهایی محدوده شامل نمی شود. هیچ آرگومانی برای تابع rand در ساده ترین شکل آن وجود ندارد. در اینجا دو مثال از فراخوانی تابع rand آورده شده است:

```
>> rand
ans =
0.8147

>> rand
ans =
0.9058
```

هر بار که MATLAB شروع می شود، seed برای تابع rand همیشه یکسان خواهد بود، مگر اینکه seed اولیه تغییر کند. تابع rng دانه اولیه را تنظیم می کند. چندین روش وجود دارد که می توان rng را فراخوانی کرد:

```
>> rng('shuffle')
>> rang(an intger number)
>> rng('default')
```

با"shuffle" ، تابع rng از تاریخ و زمان جاری استفاده می کند که از تابع ساعت داخلی بازگردانده می شود تا دانه را تنظیم کند، بنابراین seed همیشه متفاوت خواهد بود.
یک عدد صحیح نیز می تواند منتقل شود تا دانه باشد.
گزینه "default" دانه را به مقدار پیش فرضی که هنگام راه اندازی MATLAB استفاده می شود برمی گرداند. تابع rng را نیز می توان بدون آرگومان فراخوانی کرد که وضعیت فعلی مولد اعداد تصادفی را برمی گرداند : (توجه شود که نوشته های پس از علامت ":" فقط برای توضیح آنچه سمت چپ اش نوشته شده اند می باشند و خود نوشته های سمت راست این علامت در MATAB اجرا نمی شوند.)

```
>> state_rng = rng;    % gets state

>> randone = rand
randone =
0.1270

>> rng(state_rng);    % restores the state

>> randtwo = rand    % same as randone
randtwo =
0.1270
```

مولد اعداد تصادفی با شروع MATLAB مقداردهی اولیه می شود، که چیزی را تولید می کند که جریان سراسری (global stream) اعداد تصادفی نامیده می شود. همه توابع تصادفی مقادیر خود را از این جریان دریافت می کنند.

همانطور که rand یک عدد prdrd را در بازه باز (0,1) برمی گرداند، ضرب نتیجه در یک عدد صحیح N، یک عدد واقعی تصادفی را در بازه باز (0,N) برمی‌گرداند. به عنوان مثال، ضرب rand در 10 یک حقیقی را در بازه باز (0,10) برمی گرداند.

برای تولید یک عدد واقعی تصادفی در محدوده کم به زیاد، ابتدا متغیرهایی مثلا مانند low و high را ایجاد کنید. سپس، از عبارت زیر استفاده کنید:

```
rand *(high _ low) + low
```

برای نمونه، نوشتن دنباله دستورات زیر

```
>> low = 3;
>> high = 5;
>> rand *(high _ low) + low
```

یک عدد حقیقی تصادفی در بازه (0,5) تولید خواهد کرد.

تابع randn برای تولید اعداد حقیقی تصادفی با توزیع نرمال بکار گرفته می شود.

1.5.4.1 تولید اعداد تصادفی صحیح

از آنجایی که تابع rand یک عدد واقعی را برمی گرداند، می توان آن را گرد کرد تا یک عدد صحیح تصادفی تولید شود. مثلا،

```
>> round(rand*10)
```

ابتدا یک عدد صحیح تصادفی در محدوده 0 تا 10 ، که شامل (rand*10) نیست، ایجاد می‌کند. یک روش بهتر استفاده از تابع randi است که در ساده ترین شکل randi(imax) یک عدد صحیح تصادفی در محدوده 1 تا imax را برمی گرداند. برای مثال randi(4) یک عدد صحیح تصادفی را در محدوده 1 تا 4 برمی‌گرداند. به این ساختار دستوری، یک محدوده نیز می‌تواند ارسال شود، برای مثال

```
randi([imin, imax])
```

یک عدد صحیح تصادفی در محدوده ای از اعداد که شامل اعداد از imin به imax می باشد برمی‌گرداند:

```
>> randi([3, 6])
ans =
4
```

تمرین 2.1

یک عدد تصادفی n ایجاد کنید که

- عددی حقیقی در بازه (0,1) باشد
- عددی حقیقی در بازه (0,100) باشد
- عددی حقیقی در بازه (20,35) باشد

- عددی صحیح در بازه ای بسته از 1 تا 100 باشد
- عددی صحیح در بازه ای بسته از 20 تا 35 باشد

5.1 کاراکترها و رشته ها

یک کاراکتر در MATLAB با استفاده از نقل قول‌های تکی نمایش داده می شود (به عنوان مثال، 'a' یا 'x'). نقل قول ها برای نشان دادن یک کاراکتر ضروری هستند. بدون آنها، یک حرف به عنوان یک نام متغیر تفسیر می شود. کاراکترها با استفاده از چیزی که رمزگذاری کاراکتر (character encoding) نامیده می شود به ترتیب قرار می گیرند. در رمزگذاری کاراکتر، تمام کاراکترهای مجموعه کاراکترهای رایانه در یک دنباله قرار می‌گیرند و مقادیر صحیح معادل به آنها داده می‌شود. مجموعه کاراکترها شامل تمام حروف الفبا، اعداد و علائم نگارشی است. اساساً تمام کلیدهای صفحه کلید کاراکتر هستند. کاراکترهای ویژه مانند کلید Enter نیز گنجانده شده است. بنابراین، 'x'، '!'و '3' همه کاراکتر هستند. با نقل قول، "3" یک کاراکتر است، نه یک عدد.

به تفاوت قالب بندی زمانی که یک عدد در مقابل یک کاراکتر نمایش داده می شود توجه کنید:

```
>> var = 3
var =
3

>> var = '3'
var =
'3'
```

MATLAB همچنین آرایه‌های کاراکتری را که دنباله‌ای از کاراکترها در نقل قول‌های تکی هستند و رشته‌هایی که دنباله‌ای از کاراکترها در نقل قول‌های دوتایی هستند، را مدیریت می‌کند.

```
>> myword = 'hello'
myword =
'hello'

>> yourword = "ciao"
yourword =
"ciao"
```

رایج ترین رمزگذاری کاراکترها کد استاندارد آمریکایی برای تبادل اطلاعات یا است. استاندارد ASCII دارای 128 کاراکتر است که دارای مقادیر صحیح معادل از 0 تا 127 است. 32 عدد (مقادیر صحیح 0 تا 31) کاراکترهای غیرچاپی هستند. حروف الفبا به ترتیب هستند، به این معنی که «الف» قبل از «ب» آمده، سپس «ج» و غیره. MATLAB در واقع می تواند از یک دنباله رمزگذاری بسیار بزرگتر استفاده کند که همان 128 کاراکتر اولیه ASCII را دارد. اطلاعات بیشتر در مورد رمزگذاری کاراکترها و تبدیل کاراکترها به مقادیر عددی آنها در بخش 7.1پوشش داده خواهد شد.

6.1 عبارات رابطه ای

عباراتی که از نظر مفهومی درست یا نادرست هستند، عبارت های رابطه ای نامیده می شوند. آنها همچنین گاهی اوقات عبارات بولی یا عبارات منطقی نامیده می شوند. این عبارات می توانند هم از عملگرهای رابطه ای استفاده کنند که دو عبارت از انواع سازگار را به هم مرتبط می کنند و هم از عملگرهای منطقی که بر روی عملوندهای منطقی عمل می کنند.

عملگرهای رابطه ای در MATLAB عبارتند از:

عملگر	معنی و عملکرد آن
>	بزرگ تر از
<	کوچک تر از
>=	بزرگ تر یا مساوی
<=	کوچک تر یا مساوی
==	برابری
~=	نابرابری

همه این مفاهیم باید آشنا باشند، اگرچه عملگرهای واقعی مورد استفاده ممکن است با آنهایی که در سایر زبان های برنامه نویسی یا در کلاس های ریاضی استفاده می شوند متفاوت باشند. به طور خاص، توجه به این نکته مهم است که عملگر برابری دو علامت مساوی متوالی است، نه یک علامت مساوی (زیرا علامت مساوی منفرد قبلاً به عنوان عملگر انتساب استفاده می شود). برای عملوندهای عددی، استفاده از این عملگرها ساده است. به عنوان مثال، 3 < 5 به معنای "3 کمتر از 5" است، که از نظر مفهومی، یک عبارت درست است. در MATLAB، مانند بسیاری از زبان های برنامه نویسی، "true" با مقدار منطقی 1 و "false" با مقدار منطقی 0 نمایش داده می شود. بنابراین، عبارت 3 < 5 در واقع در Command Window مقدار 1 (منطقی) را در MATLAB نمایش می دهد. نمایش نتیجه عباراتی مانند این در پنجره فرمان، مقادیر عبارات را نشان می دهد.

```
>> 3 < 5
ans =
logical
1

>>  2 > 9
ans =
logical
0

>>  class(ans)
ans =
'logical'
```

نوع نتیجه یک نوع منطقی است، نه از نوع ممیز شناور مضاعف. MATLAB همچنین دارای توابع توکار true و false است.

```
>>  true
ans =
logical
1
```

به عبارت دیگر، true هم ارز است با logical(1) و false هم ارز است با logical(0) .

اگرچه این مقادیر منطقی هستند، اما عملیات ریاضی را می توان روی 1 یا 0 حاصل انجام داد.

```
>> logresult  =  5 < 7
logresult =
```

```
1

>> logresult + 3
ans =
4
```

مقایسه کاراکترها (به عنوان مثال، ’a’ < ’c’) نیز امکان پذیر است. کاراکترها با استفاده از مقادیر معادل ASCII آنها در رمزگذاری کاراکتر مقایسه می شوند. بنابراین، 'a' < 'c' یک عبارت درست است زیرا کاراکتر 'a' قبل از کاراکتر 'c' آمده است.

```
>> 'a' < 'c'
ans =
1
```

عملگرهای منطقی عبارتند از:

عملگر	معنای آن
||	یا
&&	و
~	نه (نقیض)

همه عملگرهای منطقی روی عملوندهای بولی یا منطقی عمل می کنند. عملگر ~ یک عملگر یگانی (unary) است، سایر عملگرهای جدول بالا دوتایی هستند. عملگر ~ یک عبارت منطقی درست یا نادرست را می گیرد و مقدار مخالف را می دهد. برای مثال، ~(3<5) نادرست است همانطور که (3<5) درست است. عملگر || دارای دو عبارت منطقی به عنوان عملوند است. نتیجه درست است اگر یکی یا هر دو عملوند درست باشد، و فقط اگر هر دو عملوند نادرست باشند، نادرست است. عملگر && همچنین بر روی دو عملوند منطقی عمل می کند. نتیجه یک and فقط در صورتی درست است که هر دو عملوند درست باشند. اگر یکی یا هر دو نادرست باشد، نتیجه نادرست است. عملگرهای || و && نشان داده شده در اینجا برای اسکالرها یا مقادیر منفرد استفاده می شوند. سایر عملگرهای || و && در فصل 2 توضیح داده خواهند شد.
عملگرهای || و && در MATLAB نمونه هایی از عملگرهایی هستند که به عنوان عملگرهای اتصال کوتاه شناخته می شوند. این بدان معناست که اگر بتوان نتیجه عبارت را بر اساس قسمت اول تعیین کرد، قسمت دوم حتی ارزیابی نخواهد شد. به عنوان مثال، در عبارت:

```
2 < 4 || 'a' == 'c'
```

قسمت اول یعنی، 2 < 4، درست است، بنابراین کل عبارت درست است. در این حالت، بخش دوم یعنی 'a' == 'c' ارزیابی نمی شود.
علاوه بر این عملگرهای منطقی، MATLAB یک تابع xor (exclusive or) نیز دارد که یک یای مانع جمع است. اگر یکی (و تنها یکی) از آرگومان ها درست باشد، مقدار منطقی true را برمی گرداند. به عنوان مثال، در موارد زیر فقط آرگومان اول درست است، بنابراین نتیجه درست است:

```
>> xor(3 < 5, 'a' > 'c')
ans =
1
```

در زیر مثال، هر دو آرگومان درست هستند، بنابراین نتیجه نادرست است:

```
>> xor(3 < 5, 'a' < 'c')
ans =
0
```

با فرض اینکه متغییرهای x و y مقادیر منطقی true و false را گرفته باشند، جدول درستی (به جدول 1.1 مراجعه کنید) نشان می دهد که چگونه عملگرهای منطقی برای همه ترکیب ها کار می‌کنند. توجه داشته باشید که عملگرهای منطقی دارای خاصیت جابجایی هستند (به عنوان مثال، y || x همان x || y است). همانند عملگرهای عددی، دانستن قوانین تقدم عملگر نیز مهم است. جدول 2.1 قوانین عملگرهایی را نشان می دهد که تاکنون به ترتیب اولویت پوشش داده شده اند.

Table 1.1 Truth Table for Logical Operators

x	*y*	~*x*	*x* | *y*	*x* && *y*	xor(*x*,*y*)
True	True	False	True	True	False
True	False	False	True	False	True
False	False	True	False	False	False

Table 1.2 Operator Precedence Rules

Operators	Precedence
Parentheses: ( )	Highest
Power ^	
Unary: negation (-), not (~)	
Multiplication, division *, /, \	
Addition, subtraction +, -	
Relational <, <=, >, >=, ==, ~=	
And &&	
Or ||	Lowest

پرسش سریع:

فرض کنید یک متغیر x وجود دارد که مقداردهی اولیه شده است. ارزش عبارت 3 < x < 5 چه خواهد بود اگر x مقدار 4 را داشته باشد؟
اگر x مقدار 7 را داشته باشد چطور؟
پاسخ: مقدار این عبارت همیشه بدون توجه به مقدار متغیر x از نظر منطقی درست یا 1 خواهد بود. عبارات از چپ به راست ارزیابی می شوند. بنابراین، ابتدا عبارت 3<x ارزیابی خواهد شد. تنها دو احتمال وجود دارد: یا این درست باشد یا نادرست، به این معنی که عبارت دارای یکی از مقادیر 1 یا 0 است. پس از این قسمت بعدی عبارت ارزیابی می گردد، که یا 1<5 و یا 0<5 خواهد بود. هر دوی این دو عبارت درست هستند. بنابراین، اینکه مقدار x چه باشد تاثیری در نتیجه عبارت صورت پرسش بالا ندارد و نتیجه آن همیشه درست است. این یک خطای منطقی است و هیچ پیامی یا پیشنهادی از سوی MATLAB دریافت نخواهد کرد. اگر هدف ما بکارگیری عبارتی بود و باشد که نتیجه آن تنها وقتیx در بازه یاد شده باشد، انگاه می‌توانیم عبارت زیر را بکار بگیریم: 3<x && x<5

تمرین 3.1

به این فکر کنید که با عبارات زیر چه چیزی تولید می شود و سپس آنها را در پنجره دستورات تایپ کنید تا پاسخ های خود را تأیید کنید:

```
3 == 5 + 2
'b' < 'a' + 1
10 > 5 + 2
(10 > 5) + 2
'c' == 'd' - 1 && 2 < 4
'c' == 'd' - 1 jj 2 > 4
xor('c' == 'd' - 1, 2 > 4)
```

```
xor('c' == 'd' - 1, 2 < 4)
10 > 5 > 2
```

توجه: در استفاده از عملگرهای برابری و نابرابری با اعداد دقت کنید. گاهی اوقات، خطاهای گردکردن ظاهر می شود، به این معنی که اعداد به مقدار درست خود نزدیک هستند اما دقیقاً همانی که باید نیستند. به عنوان مثال، cos(pi/2) باید 0 باشد. با این حال، به دلیل یک خطای گردکردن، عدد بسیار کوچکی است اما دقیقا 0 نیست.

```
>> cos(pi/2)
ans =
6.1232e-17

>> cos(pi/2) == 0
ans =
0
```

7.1 محدوده نوع و تبدیل نوع

محدوده یک نوع که نشان دهنده کوچکترین و بزرگترین اعداد قابل ذخیره در نوع است قابل محاسبه است. به عنوان مثال، نوع uint8 تعداد 2^8 یا 256 عدد صحیح را ذخیره می کند که از 0 تا 255 متغیر است. محدوده مقادیری که می توان در int8 ذخیره کرد، از 128- تا 127+ است. محدوده را می توان برای هر نوع با ارسال نام نوع به عنوان یک رشته یا بردار کاراکتر (که قاعدتا در علامت نقل قول تکی است) به توابع intmin و intmax پیدا کرد. مثلا،

```
>> intmin('int8')
ans =
-128

>> intmax('int8')
ans =
127
```

توابع زیادی وجود دارند که مقادیر را از یک نوع به نوع دیگر تبدیل می کنند. نام این توابع با نام انواع یکسان است. این نام ها می توانند به عنوان توابعی برای تبدیل یک مقدار به آن نوع استفاده شوند. این را تبدیل یک مقدار به نوع دیگر می نامند. به عنوان مثال، برای تبدیل یک مقدار از نوع double، که پیش فرض است، به نوع int32، از تابع int32 استفاده می شود. برای نمونه می دانیم با نوشتن و اجرای یک عبارت انتساب به شکل زیر

```
>> val = 6 + 3;
```

مقدار عدد 9 حاصل از جمع به متغیر val ، که به طور پیش فرض از نوع اعشاری مضاعف (double precision) است، انتساب داده می شود. پس از آن دستور زیر

```
>> val = int32(val);
```

بدون تغییر مقدار val نوع آن را به int32 تغییر می دهد.
مثال دیگری با بکارگیری دو متغییر را در زیر می بینیم:

```
>> num = 6 + 3;
>> numi = int32(num);
>> whos
```

```
Name  Size  Bytes  Class Attributes
num   1x1   8      double
numi  1x1   4      int32
```

توجه داشته باشید که whos نوع (کلاس) متغیرها و همچنین تعداد بایت های استفاده شده برای ذخیره مقدار یک متغیر را نشان می دهد. یک بایت معادل هشت بیت است، بنابراین نوع int32 از 4 بایت استفاده می کند.

یکی از دلایل استفاده از نوع عدد صحیح برای یک متغیر، صرفه جویی در فضای حافظه است.

پرسش سریع!

اگر برای یک نوع خاص از محدوده آن فراتر بروید چه اتفاقی می افتد؟ به عنوان مثال، بزرگترین عدد صحیحی که می توان در int8 ذخیره کرد 127 است، بنابراین اگر بخواهیم تبدیل نوع عدد صحیحی بزرگتر از 127 را به کمک int8 انجام دهیم چه اتفاقی می افتد؟

```
>> int8 (200)
```

پاسخ: مقدار در این محدوده بزرگترین خواهد بود، در این مورد بزرگترین عدد ممکن 127 است.
اگر در عوض از یک عدد منفی که کوچکتر از کمترین مقدار در محدوده int8 است استفاده کنیم، مقدار آن 128- خواهد بود. این نمونه ای از چیزی است که محاسبات اشباع نامیده می شود.

```
>> int8(200)
ans =
127

>> int8(_130)
ans =
-128
```

تمرین 4.1

-- محدوده اعداد صحیح قابل ذخیره در انواع int16 و uint16 را محاسبه کنید. برای تایید نتایج خود از intmin و intmax استفاده کنید.
-- یک عبارت انتساب را وارد کنید و نوع متغیر را در پنجره Workspace مشاهده کنید. سپس نوع آن را تغییر دهید و دوباره مشاهده کنید. آن را با استفاده از whos نیز مشاهده کنید.

همچنین یک تابع تبدیل (cast) وجود دارد که می تواند یک متغیر را به یک نوع خاص تبدیل کند. این گزینه ای برای تبدیل یک متغیر به نوعی مشابه نوع معلوم دیگر با استفاده از "like" است

```
. >> a = uint16(43);
>> b = 11;
>> whos
Name  Size  Bytes  Class Attributes
A     1x1   2      uint16
b     1x1   8      double
```

```
>> b = cast(b,'like',a);
>> whos
Name  Size  Bytes  Class Attributes
A     1x1   2      uint16
b     1x1   2      uint16
```

از توابع عددی نیز می توان برای تبدیل یک کاراکتر به مقدار عددی معادل آن استفاده کرد (به عنوان مثال، استفاده از double یک کاراکتر را به یک مقدار دوگانه تبدیل می کند و int32 آن را با استفاده از 32 بیت به یک مقدار صحیح تبدیل می کند). به عنوان مثال، برای تبدیل کاراکتر "a" به معادل عددی آن، می توان از عبارت زیر استفاده کرد:

```
>> numequiv = double('a')
numequiv =
97
```

اجرای این دستور مقدار از نوع اعشاری مضاعف شده 97 را در متغیر numequiv ذخیره می‌کند، که نشان می‌دهد کاراکتر "a" نود و هشتمین کاراکتر در رمزگذاری کاراکتر است (زیرا اعداد معادل از 0 شروع می‌شوند). مهم نیست که کدام نوع عددی برای تبدیل "a" استفاده می شود. برای نمونه،

```
>> numequiv = int32('a')
```

این مقدار عدد صحیح 97 را نیز در متغیر numequiv ذخیره می کند. تنها تفاوت بین اینها نوع متغیر حاصل خواهد بود (در مورد اول اعشاری مضاعف double، در حالت دوم int3)

تابع char برعکس عمل می کند. هر عددی به کاراکتر معادل اش تبدیل می کند:

```
>> char(97)
ans =
'a'
```

توجه داشته باشید که در نسخه های قبلیMATLAB ، نقل قول ها هنگام نمایش کاراکتر نشان داده نمی شدند.
از آنجایی که حروف الفبا به ترتیب هستند، کاراکتر 'b' معادل 98، کاراکتر 'c' معادل 99 است. عملیات ریاضی را می توان روی کارکتر ها بکار گرفت. به عنوان مثال، برای دریافت کاراکتر بعدی در رمزگذاری کاراکتر، 1 را می توان به عدد صحیح یا کاراکتر اضافه کرد:

```
>> numequiv = double('a');
>> char(numequiv + 1)
ans =
'b'

>> 'a' + 2
ans =
```

99

128 کاراکتر اول معادل 128 کاراکتر در ASCII استاندارد است. MATLAB از یک رمزگذاری استفاده می‌کند که دارای 65535 کاراکتر است. کاراکترهای از 128 تا 65535 به تنظیمات محلی رایانه شما بستگی دارد که زبان رابط شما را تعیین می کند. به عنوان مثال، "en_US" محل زبان انگلیسی در ایالات متحده است.

برای جابجایی کاراکترهای یک رشته به سمت بیشتر (یعنی بالا) در رمزگذاری کاراکتر، می توان یک مقدار صحیح به یک رشته اضافه کرد. به عنوان مثال، عبارت زیر به میزان یک مکان (مقدار) تغییر می کند:

```
>> char("abcd" + 1)
ans =
"bcde"
```

تمرین 5.1

-- معادل عددی کاراکتر x را پیدا کنید.
-- کاراکتر معادل 107 را پیدا کنید.

8.1 توابع عددی توکار

ما در بخش 3.4.1 دیدیم که از دستور help می توان برای مشاهده موضوعات راهنما مانند elfun و همچنین توابع ذخیره شده در هر موضوع help استفاده کرد.
MATLAB بسیاری از توابع مثلثاتی داخلی برای سینوس، کسینوس، مماس و غیره را دارد. به عنوان مثال، sin تابع سینوس بر حسب رادیان است. تابع معکوس یا آرکسین در رادیان asin ، تابع سینوس هذلولی بر حسب رادیان sinh است و تابع سینوس هذلولی معکوس asinh است. همچنین توابعی وجود دارند که به جای رادیان از درجه استفاده می کنند: مثلا sind و. asind . تغییرات مشابهی برای سایر توابع مثلثاتی وجود دارد.
fix, نیز دارای توابع گردکردن و باقیمانده است که بسیار مفید هستند. برخی از این موارد عبارتند از elfun helpعلاوه بر توابع مثلثاتی، بحث floor, ceil, round, mod, rem, sign.

هر دو تابع rem و mod باقیمانده را از یک تقسیم برمی گرداند. برای مثال 5 دو بار با باقیمانده 3 به 13 وجود دارد، بنابراین نتیجه این عبارت 3 است:

```
>> rem(13,5)
ans =
3
```

پرسش سریع!

اگر به اشتباه ترتیب آرگومان ها را معکوس کنید و موارد زیر را تایپ کنید چه اتفاقی می افتد:

```
rem (5,13)
```

پاسخ: تابع rem نمونه ای از تابعی است که دو آرگومان به آن ارسال شده است. در برخی موارد، ترتیب ارسال آرگومان ها مهم نیست، اما برای تابع rem ترتیب اهمیت دارد. تابع rem آرگومان دوم را به آرگومان اول تقسیم می کند. در این حالت، آرگومان دوم، یعنی 13، صفر بار در 5 وجود دارد با باقیمانده 5. بنابراین 5 به عنوان نتیجه برمی‌گردد.

تابع دیگر در مبحث elfun help تابع علامت است که اگر آرگومان مثبت باشد 1، اگر 0 باشد 0 و اگر منفی باشد 1- را برمی گرداند. مثلا،

```
>> sign(-5)
ans =
_1

>> sign(3)
ans =
1
```

تمرین 6.1

از تابع help استفاده کنید تا متوجه شوید که عملکردهای fix, floor, ceil, round چه کاری انجام می دهند. با انتقال مقادیر مختلف به توابع، از جمله برخی منفی، برخی مثبت، و برخی با کسری کمتر از 0.5 و برخی بزرگتر، آنها را آزمایش کنید. هنگام آزمایش توابع بسیار مهم است که با آزمایش انواع مختلف آرگومان ها به طور کامل آزمایش کنید!

تابع round انتخابی دارد که عدد آرگومان را تا تعداد اعداد مشخص شده گرد می کند.

```
>> round(pi,3)
ans =
3.1420
```

MATLAB دارای عملگر توان ^ و همچنین تابع sqrt برای محاسبه ریشه ها و nthroot برای یافتن ریشه n ام یک عدد است. برای مثال عبارت زیر ریشه سوم 64 را پیدا می کند.

```
>> nthroot(64,3)
ans =
4
```

می دانیم وقتی $x = b^y$ ، یعنی y لگاریتم x در پایه b است، به عبارت دیگر، $y = \log_a x$. MATLAB دارای توابع توکار زیر برای محاسبه لگاریتم است:

log(x) لگاریتم طبیعی x را نتیجه می دهد،

$log2(x)$ لگاریتم در پایه 2 عدد x را محاسبه و بازپس می فرستد،

$log10(x)$ لگاریتم در پایه 10 عدد x را محاسبه و بازپس می فرستد.

پرسش سریع!

تابع توکاری برای عد نپر e در MATLAB وجود ندارد، بنابراین چگونه می توان آنرا در MATLAB بدست آورد؟

پاسخ: تابع نمایی exp را بکار بگیرید،

```
>> exp(1)
ans =
2.7183
```

در R2015b توابع deg2rad و rad2deg برای تبدیل بین درجه و رادیان ارایه گردید، برای مثال،

```
>> deg2rad(180)
ans =
```

3.1416