

دوره آموزشی برنامه متمتیکا

دوره سایت فرادرس

جلسه اول

حذف متغیر در برنامه (هر متغیری قبل از این دستور پاک می‌شود).

```
Clear["Global`*"]
```

تنها متغیر a حذف می‌شود.

```
Clear[a]
```

غیرفعال کردن یک دستور:

منوی edit ← Un/comment Section

کلید میانبر : Alt + /

نشان دادن هر متغیری (چه مقدار دهی شده و چه مقدار دهی نشده)

```
Clear["Global`*"]
```

```
Clear["Name`*"]
```

\* همیشه باید در اول هر برنامه از دو کد Remove و Clear استفاده کنیم تا در محاسبات مشکلی پیش نیاید.

\* در متمتیکا حرف اول همه دستورها باید بزرگ شود.

\* روش اجرا دستورها در یک سل : Shift + Enter

اجرای دستورها همه سل‌ها به صورت یکجا: *Evaluation* → *Evaluate Notebook*

دستور برای مقداردهی یک متغیر:

```
Variable_name = Value
```

برای مثال در کد زیر مقدار ۲ را درون متغیری با نام a قرار می‌دهیم:

```
a = 2
```

مشکلی که این نوع مقداردهی دارد این است که بعد از اجرا سلولی که این کد درون آن است، این مقدار در خروجی چاپ می‌شود. برای اینکه هم مقدار موردنظر در متغیر موردنظر ذخیره شود و هم اینکه در خروجی نمایش داده نشود. دو روش وجود دارد. روش اول:

```
a = 2;
```

روش دیگری که موجود است:

```
a := 2
```

تفاوت این دو دستور باهم این است که در روش اول مقدار حساب و ذخیره می‌شوند. ولی در روش دوم مقدار حساب نمی‌شود تا زمانی که از آن در جایی استفاده شود

روش متوقف کردن برنامه در حال اجرا:

Local Quit kernel Evaluation

جواب سوال پرسیده شده: Quit

\* هر دستور در متمتیکا باید در براکت ([]) نوشته شود:

```
Sin[x] , Clear[c]
```

دستورهای پایه ریاضی

برای این بخش دو متغیر از قبل تعریف می‌کنیم:

```
a = 5
```

```
b = 8
```

دستور جمع:

```
a + b      Out : 13
```

دستور تفریق:

```
a - b      Out : -3
```

دستور ضرب:

```
a * b      Out : 40
```

دستور تقسیم (روش اول):

```
1 a / b          Out : 5 / 8
```

دستور تقسیم (روش دوم):

```
1 a/b          Out : 5 / 8
```

نحوه خروجی گرفتن با دقت موردنظر:

```
1 N[expression]          e.g: N[1/3]          Out: 0.333
2 N[expression , accuracy] e.g: N[1/3 , 10]      Out: 0.3333333333
```

برای نوشتن این روش پس از نوشتن صورت تقسیم (در این مثال  $a / b$ ) Ctrl + / را می‌زنیم.  
روش تایپ کردن فرمول‌های خاص:  
*Palettes* → *BasicMathAssistant*  
استفاده از حروف یونانی و اسمبل‌ها در این قسمت نیز موجود است.  
هز گزینه کلید میانبر مخصوص خود را دارد. برای آشنایی با کلید میانبر هر گزینه، روی گزینه کمی نگه دارید.  
\* خطای ۱۰۴۲ : ایجاد اشکال در دستور وارد شده، مثال:

```
1 W = W + 1 , W = W1 + 1
```

روش دیگر برای نوشتن این عبارت: در این حالت کنترلی در تعداد اعشار نداریم.

```
1 1 / 3 // N
```

جلسه دوم : معرفی توابع  
تابع رادیکال:

```
1 Sqrt[4]          Out: 2
2 Sqrt[x**2]       Out:  $\sqrt{x^2}$ 
```

تابع ساده سازی (Simplify):

```
1 Simplify[%, x > 0]          Out: +x
2 Simplify[%, x < 0]          Out: -x
```

در این مثال نماینده آخرین خروجی است.  
تابع نمایی:

```
1 Exp[x]          Out: e**x          e = 2.7
2 Exp[2]          Out: e**2
3 Exp[2] // N      Out: 7.38906
```

تابع جزء صحیح:

```
1 Floor[2.3]       Out: 2
```

تابع قدرمطلق:

```
1 Abs[-2]          Out: 2
2 Abs[-x]          Out: Abs[x]
```

\* در مثال دوم این کد، خروجی از تابع بیرون در نمی‌آید چون x می‌تواند منفی و چیزهای دیگر باشد.  
تابع علامت:

```
1 Sign[{-2, 0, 3}]          Out: {-1, 0, 1}
```

تابع فاکتوریل (روش اول):

```
1 x = 5
2 Facorial[x]          Out: 120
```

تابع فاکتوریل (روش دوم):

```
1 x!          Out: 120
```

تابع لگاریتم:

```
Log[x] Out: Log[x]
```

زمانی که برای الگوریتم پایه تعریف نکنیم، نرم افزار به صورت خودکار پایه را عدد نپر (E یا  $\ln$ ) قرار می‌دهد.

```
Log[E] Out: 1
```

تعریف پایه الگوریتم:

```
Log[a , x] a is The base of the logarithm
Log[10 , 1000] Out: 3
```

توابع مثلثاتی:

```
Sin[x] e,g: Sin[Pi/3] Out: sqrt[3] / 2
Cos[x]
Tan[x]
Cot[x]
ArcSin[x]
ArcTan[x]
```

به صورت دیفالت مقدار محاسبه این توابع به صورت رادیان است، اگر بخواهیم نرم افزار خاص درجه موردنظر را حساب کند، این گونه عمل می‌کنیم:

```
Sin[30 Degree] Out: sqrt[3] / 2
Sin[30 Degree] // N Out: 0.5
```

توابع هیپربولیک

```
Sinh[x]
Cosh[x]
Coth[x]
```

تابع تولید اعداد تصادفی  
برای اعداد حقیقی:

```
RandomReal[{-2, 2}, 5] Out: {1.31496, 0.0146972, -1.486, -1.93584, 0.638742}
RandomReal[{-2, 2}, {2, 5}] Out: {{1.86401, -1.47587, 0.0531725, 1.73636, 0.243321},
{1.50354, 0.924516, 1.60522, 0.870081, -1.22427}}
```

در مثال اول پنج عدد تصادفی در بازه ۲، -۲ انتخاب می‌شوند.  
در مثال دوم دو دسته پنج تایی عدد تصادفی در بازه ۲، -۲ انتخاب می‌شوند.  
برای اعداد صحیح:

```
RandomInteger[{-2, 2}, 5] Out: {-2, 0, 1, 2, -2}
```

پنج عدد صحیح تصادفی در بازه ۲، -۲ انتخاب می‌شوند.  
تبدیل عدد به عامل های اول:

```
FactorInteger[10] Out: {{2, 1}, {5, 1}} 10 = 2^1 * 5^1
```

به توان رساندن:

```
Superscript[2, 3] Out: Superscript[2,3]
Superscript @@ {2, 3} Out: Superscript[2,3]
Superscript @@@ {{2, 3}, {6, 5}} Out: {Superscript[2,3], Superscript[6,5]}
Superscript @@ {{2, 3}, {4, 5}} Out: Superscript[{2, 3},{4, 5}]
```

```
CenterDot[x, y] Out: x\[CenterDot]y
CenterDot @@ (Superscript @@@ (FactorInteger[15])) Out: Superscript[3,1]\[CenterDot]
Superscript[5,1]
```

کوچکترین مضرب مشترک

```
LCM[5, 6]
```

بزرگترین تقسیم الیه مشترک

```
1 GCD[15, 9]
```

باقی مانده

```
1 Mod[15, 2]
```

خارج قسمت

```
1 Quotient[15, 3]
```

ترکیب

```
1 Binomial[15, 3]
```

دلتا دیراک

```
1 DiracDelta[x]
2 DiracDelta[x] // TraditionalForm Out: \[Delta](x)
```

دلتا کراینیکر

```
1 DiracDelta[m, n] // TraditionalForm
```

اعداد مختلط:

```
1 z = x + I y Out: x + I y
2 Re[z] Out: -Im[y] + Re[x]
```

برای تعریف کردن قسمت موهومی و واقعی اعداد مختلط:

```
1 Refine[Re[z], Element[{x, y}, Reals]] Out: x // Real Part
2 Refine[Im[z], Element[{x, y}, Reals]] Out: y // Imaginary Part
```

مزدوج گیری (مختلط):

```
1 Conjugate[z] Out: Conjugate[x] - I Conjugate[y]
```

ریفاین کردن

```
1 Refine[Conjugate[z], Element[{x, y}, Reals]] Out: x - I y
```

تعریف تابع دلخواه:

```
1 F[x_] = x^2 + 1 Out: 1 + x^2
2 F[3] Out: 10
3 Map[F, {2, 3}] Out: {5, 10}
4 F /@ {2, 3} Out: {5, 10}
```

برای تعریف توابع چند متغیره:

```
1 G[x_, y_] = x + y + 2 Out: 2 + x + y
2 G[{2, 3}, {5, 6}] Out: {9, 11}
```

جلسه سوم : محاسبات جبری و مثلثاتی، سری ها