

دوره آموزشی برنامه متمتیکا
دوره سایت فرادرس
جلسه اول
حذف متغیر در برنامه (هر متغیری قبل از این دستور پاک می‌شود).

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

تنها متغیر a حذف می‌شود.

```
1 Clear[a]
```

غیرفعال کردن یک دستور:
منوی Un/comment Section ← edit
کلید میانبر : Alt + /
نشان دادن هر متغیری (چه مقدار دهنده و چه مقدار نشده)

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

- * همیشه باید در اول هر برنامه از دو کد Clear و Remove استفاده کنیم تا در محاسبات مشکلی پیش نیاید.
- * در متمتیکا حرف اول همه دستورها باید بزرگ شود.
- * روش اجرا دستورهای در یک سل : Shift + Enter
- اجرای دستورهای همه سل‌ها به صورت یکجا: Evaluation → EvaluateNotebook
- دستور برای مقداردهنده یک متغیر:

```
1 Variable_name = Value
```

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

```
1 a = 2
```

مشکلی که این نوع مقداردهنده دارد این است که بعد از اجرا سلولی که این کد درون آن است، این مقدار درخروجی 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:

```
1 a = 2;
```

روش دیگری که موجود است:

```
1 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:

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

دستورهای پایه ریاضی
برای این بخش دو متغیر از قبل تعریف می‌کنیم:

```
1 a = 5  
2 b = 8
```

دستور جمع:

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

دستور تفریق:

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

دستور ضرب:

```
1 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: 0.333333333333
```

برای نوشتن این روش پس از نوشتن صورت تقسیم (در این مثال a / b) را $\text{Ctrl} + \text{Shift} + =$ میزنیم.
روش تایپ کردن فرمولهای خاص:

Palettes → BasicMathAssistant

استفاده از حروف یونانی و اسمايلها 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، مثلا:

```
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: معرف توابع
تابع رادیکال:

```
1 Sqrt[4]           Out: 2
2 Sqrt[x**2]       Out: √(x^2)
```

: (Simplify) تابع ساده سازی

```
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 نمی آید 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:

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

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

```
1 Log[E]
```

```
Out: 1
```

تغیریف پایه الگوریتم:

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

توابع مثلثاتی:

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

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

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

توابع هیپربولیک

```
1 Sinh[x]  
2 Cosh[x]  
3 Coth[x]
```

تابع تولید اعداد تصادفی
برای اعداد حقیقی:

```
1 RandomReal[{-2, 2}, 5] Out: {1.31496, 0.0146972, -1.486, -1.93584, 0.638742}  
2 RandomReal[{-2, 2}, {2, 5}] Out: {{1.86401, -1.47587, 0.0531725, 1.73636, 0.243321},  
3 {1.50354, 0.924516, 1.60522, 0.870081, -1.22427}}
```

در مثال اول پنج عدد تصادفی در بازه $[-2, 2]$ -انتخاب می‌شوند.
در مثال دوم دو دسته پنج تایی عدد تصادفی در بازه $[-2, 2]$ -انتخاب می‌شوند.
برای اعداد صحیح:

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

پنج عدد صحیح تصادفی در بازه $[-2, 2]$ -انتخاب می‌شوند.
تبديل عدد به عامل های اول:

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

به توان رساندن:

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

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

کوچکترین مضرب مشترک

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

بزرگترین تقسیم الیه مشترک

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

باقي مانده

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

خارج قسمت

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

ترکیب

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

دلtar دیراک

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

دلتا کراینیکر

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

اعداد مختلف:

```
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 مختلف:

```
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، سری‌ها بسط یک عبارت

```
1 (x + 1)^3 Out: (1 + x)^3
2 Expand[(x + 1)^3] Out: 1 + 3 x + 3 x^2 + x^3
3 (x + 1)^3 // Expand Out: 1 + 3 x + 3 x^2 + x^3
4 Expand[(a + b)^2*(x + 1)^2] Out: a^2 + 2 a b + b^2 + 2 a^2 x + 4 a b x + 2 b^2 x
5 + a^2 x^2 + 2 a b x^2 + b^2 x^2
```

همانطور که مشاهده می‌شود در مثال آخر کدهای بالا هر دو جمله بسط داده می‌شوند. گاهی نیاز است که تنها یکی از این جملات بسط داده شود. برای این کار:

```
1 Expand[(a + b)^2*(x + 1)^2, x] Out: (a + b)^2 + 2 (a + b)^2 x + (a + b)^2 x^2
```

همانطور که مشاهده می‌شوند فقط جمله‌ای که در آن x وجود دارد، بسط داده خواهد شد.

فاکتورگیری

```
1 Factor[x^4 + x^2 + x] Out: x (1 + x + x^3)
2 Factor[x^4 + x^2] Out: x^2 (1 + x^2)
```

به مثال زیر توجه کنید:

```

1 y = x^4 + x^2 + x
2 x = a + b
3 Factor[y]      Out: (a + b) (1 + a + a^3 + b + 3 a^2 b + 3 a b^2 + b^3)

```

همانطور که مشاهده می شود در این کد، از $(a+b)$ فاکتور گرفته می شود.
ساده کردن یک عبارت کسری

```

1 Cancel[(x^2 - 1)/(x - 1)]      Out: 1 + x

```

تجزیه کسر به کسرهای جزئی

```

1 Apart[1/((1 + x)*(5 + x))]   Out: 1/(4 (1 + x)) - 1/(4 (5 + x))
2 Expand[(1 + x)*(5 + x)]       Out: 5 + 6 x + x^2
3 Apart[1/%]                     Out: 1/(4 (1 + x)) - 1/(4 (5 + x))

```

بسط عبارات مثلثاتی و تبدیل آنها به عبارت نهایی

```

1 Cancel[Sin[2*x]/Sin[x]]        Out: Csc[x] Sin[2 x]
2 Cancel[Sin[2*x]/Sin[x], Trig -> True]  Out: 2 Cos[x]

```

ساده سازی عبارت های مثلثاتی

```

1 TrigFactor[Cos[x + y] + Sin[x]*Sin[y]]      Out: Cos[x] Cos[y]

```

بسط عبارت های مثلثاتی

```

1 TrigExpand[Cos[x + y]]           Out: Cos[x] Cos[y] - Sin[x] Sin[y]
2 TrigExpand[Sin[2*x]]             Out: 2 Cos[x] Sin[x]

```

تبدیل عبارت های مثلثاتی درجه های بالاتر به عبارت خطی

```

1 TrigReduce[2*(Cos[x])^2]         Out: 1 + Cos[2 x]
2 TrigReduce[(Cos[x])^3]           Out: 1/4 (3 Cos[x] + Cos[3 x])

```

تبدیل عبارت های مثلثاتی به نمایی و بر عکس

```

1 TrigToExp[Cos[x]]               Out: E^(-I x)/2 + E^(I x)/2
2 ExpToTrig[Exp[Ix]]              Out: Cosh[Ix] + Sinh[Ix]

```

ساده سازی عبارت ها

نکته: در نرم افزار متمتیکا در دو حالت زیر ضرب تعریف می شود:

(۱) بین دو عبارت علامت ضرب "*" قرار گیرد.

(۲) بین دو عبارت اسپیس گذاشته شود.

```

1 Simplify[(x - 1) (x + 1) (x^2 + 1) + 1]      Out: x^4
2 Simplify[(Sin[x])^2 + (Cos[x])^2]                Out: 1
3 FullSimplify[Cosh[x] - Sinh[x]]                  Out: E^-x

```

نکته ۱: Simplify از FullSimplify قوی تر است. کارایی هر دو یکی است.

نکته ۲: گاهی باید برای ساده سازی یک عبارت، ابتدا باید آن Expand شود یا در کسرها از Cancel استفاده شود.
سری (مجموعه ها)

```

1 Sum[Sin[i x], {i, 1, 5}]      Out: Sin[x] + Sin[2 x] + Sin[3 x] + Sin[4 x] + Sin[5 x]
2 Sum[i, {i, 1, n}]              Out: 1/2 n (1 + n)
3 Sum[i^2, {i, 1, n}]            Out: 1/6 n (1 + n) (1 + 2 n)
4 Sum[Sin[i x], {i, 1, 10, 2}]   Out: Sin[x] + Sin[3 x] + Sin[5 x] + Sin[7 x] + Sin[9 x]

```

روش دیگر: با استفاده از کلید میان ESC + Sum + ESC می توان این نماد را رسم کرد.

$$\sum_{i=1}^n i$$

$$\sum_{i=1}^5 i^2$$

خروجی کدهای تصویر بالا:

```
1 Out1: 1/2 n (1 + n)
2 Out2: 55
```

حاصل ضرب:

```
1 Product[Sin[i x], {i, 1, 10}]      Out: Sin[x] Sin[2 x] Sin[3 x] Sin[4 x] Sin[5 x]
2                                Sin[6 x] Sin[7 x] Sin[8 x] Sin[9 x] Sin[10 x]
3
4 Product[Sin[i x], {i, 1, 10, 2}]  Out: Sin[x] Sin[3 x] Sin[5 x] Sin[7 x] Sin[9 x]
```

روش دیگر: با استفاده از کلید میان ESC + Prod + ESC می‌توان این نماد را رسم کرد.

خروجی کد تصویر بالا:

```
1 Out: 120
```

جلسه چهارم : حد، مشتق و انتگرال
بخش اول - محاسبه حد

```
1 Limit[f[x], x -> x0]
2 Limit[f[x], x -> x0] // TraditionalForm
```

خروجی کد بالا به شکل زیر می‌باشد:

مثالی دیگر:

```
1 Limit[Sin[x]/x, x -> 0]      Out: 1
```

نکته: اگر یک عبارت حد چپ و راست برابر نداشته باشد، حد ندارد و در نرم‌افزار پیامی طبق، این مفهوم می‌آید.

```
1 Limit[1/x, x -> 0]          Out: Indeterminate
```

همچنین می‌توانیم حد چپ و راست را به صورت جداگانه برای یک عبارت حساب کنیم. به مثال زیر توجه کنید.
حد چپ: از سمت مقادیر کمتر از صفر به سمت صفر

```
1 Limit[1/x, x -> 0, Direction -> 1]      Out: -Infinity
```

حد چپ: از سمت مقادیر بیشتر از صفر به سمت صفر

```
1 Limit[1/x, x -> 0, Direction -> -1]      Out: Infinity
```

بخش دوم - مشتق
برای انجام عمل مشتق در برنامه می‌توان به دو صورت زیر اقدام کرد:

```
1 Derivative[n][f][x + 1]
2 D[f[x], {x, n}]
```

که خروجی کد بالا به شکل زیر در ترمینال نمایش داده خواهد شد:

```
Out[•]= f^(n) [1 + x]
```

```
Out[•]= f^(n) [x]
```

نکته: در کد دوم پارامتر $f(x)$ تابع مورد نظر برآش مشتق گرفتن، پارامتر x متغیری که می خواهیم طبق آن مشتق گیری انجام دهیم و پارامتر n تعداد مشتق است.
به مثال زیر توجه کنید:

```
1 D[(Sin[x])^10, x] Out: 10 Cos[x] Sin[x]^9
```

در این مثال تابع $\sin(x)$ به عنوان تابع ما و x به عنوان پارامتر مشتق گیری قرار داده شده است. توجه شود که پارامتر n در این مثال مشخص نشده و به پیشفرض ۱ در نظر گرفته می شود.
مشتق جزئی:

```
1 D[f[x, y], x, y]
2 D[f[x, y], {x, 2}, {y, 3}]
```

خروجی آن در ترمینال به شکل زیر است:

```
Out[•]= f^(1,1) [x, y]
```

```
Out[•]= f^(2,3) [x, y]
```

مثال کاربردی:

```
1 D[x^3 + y^2, {x, 2}, {y, 1}] Out: 0
```

بخش سوم - انتگرال گیری

```
1 Integrate[f[x], x]
```

که خروجی آن در ترمینال به صورت زیر نمایش داده می شود:

```
Out[•]= \int f[x] dx
```

یک مثال از انتگرال نامعین:

```
1 Integrate[Sin[x], x] Out: -Cos[x]
```

انتگرال دوگانه نامعین:

```
1 Integrate[f[x, y], x, y]
```

که خروجی آن در ترمینال به صورت زیر نمایش داده می شود:

```
Out[•]= \int \int f[x, y] dy dx
```

یک مثال از انتگرال دوگانه نامعین:

```
1 Integrate[x*y + 1, x, y] Out: x y + (x^2 y^2)/4
```

انتگرال معین:

```
1 Integrate[1/(x^3 + 1), {x, 0, 1}] Out: 1/18 (2 Sqrt[3] \[Pi] + Log[64])
```

نکته: جواب انتگرال گیری‌های معین است.

یک مثال از انتگرال دوگانه معین:

```
1 Integrate[x*y + 1, {x, 0, t}, {y, -x, x}] Out: t^2
```

انتگرال‌های عددی:

```
1 NIntegrate[Sin[Sin[x]], {x, 0, 4}] Out: 1.45747
```

اما چرا از این روش استفاده می‌شود. باید از روش قبل انتگرال بالا را حساب کنیم:

```
1 Integrate[Sin[Sin[x]], {x, 0, 4}]
```

خروجی به شکل زیر خواهد بود:

$$Out[\bullet]= \int_0^4 \text{Sin}[\text{Sin}[x]] \, dx$$

دلیل آن بر این است که این انتگرال به صورت بسته با توابع ابتدایی بیان شدنی نیست. همین است که Integrate در معمولاً جواب نمادین برنمی‌گرداند. Mathematica یک مثال دیگر:

```
1 a = 2 ;
2 NIntegrate[a*Sin[Sin[x]], {x, 0, 4}] Out: 2.91494
```