Пакет “Mathematica” является наиболее развитой системой символьных и численных вычислений. Программы, написанные с помощью встроенного в пакет языка программирования, коротки и весьма эффективны.

Пакет “Mathematica” состоит из ядра, которое позволяет вычислять огромное число функций с произвольной точностью, выполнять большое количество аналитических вычислений и преобразований, строить графики и т.д. При необходимости использования новых функций, не входящих в ядро можно программировать новые функции или использовать дополнения к пакету (так называемые addons), резко усиливающие эффективность вычислений. Некоторые дополнения входят в состав пакета (например, ряд дополнений по статистике, по работе с векторными полями, представлению на фазовой плоскости решений дифференциальных уравнений и т.д.). Другие – распространяются в Интернете. Наконец, ряд дополнений распространяются на платной основе (например addons Optic, позволяющий моделировать произвольные оптические системы).

В настоящем методическом указании будут даны основы использования данного пакета, необходимые для начала работы.

Команды, входящие в пакет, имеют стандартную структуру:

1. **Команда** начинается с **большой (прописной) буквы** латинского алфавита.
2. **Аргументы** команды стоят в **квадратных скобках.**
3. **Атрибуты** команды (дополнительные указания) стоят в **фигурных скобках**.

Например, команда `Cos[x]` вычисляет величину косинуса аргумента `x`.

Команда `Plot[Sin[x], {x, 0, 10}]` строит график функции `Sin[x]` в диапазоне изменения `x` от 0 до 10.

Выполнение команды происходит после одновременного нажатия клавиш Shift+ Enter (курсор должен находиться в строке этой команды).

ЧИСЛА

В пакете следует различать 4 типа чисел:

1. Целые (Integer) числа (например, `234`).
2. Рациональные (Rational) числа (например, `127/29`).
3. Вещественные (Real) числа (например, `3.14` или `234.`). Обратите внимание на отличие между целым числом `234` и вещественным `234.` – точка у вещественного числа.
4. Комплексные числа (например, 4+7 I). Вещественные и мнимые части комплексных чисел могут быть целыми, рациональными или вещественными.

Математические константы – `Pi` – число π (можно вводить путем последовательного нажатия клавиш Esc P Esc; `I` – мнимая единица (можно вводить путем последовательного нажатия Esc ii Esc); E – основание натуральных логарифмов (можно вводить путем последовательного нажатия клавиш Esc ee Esc); GoldenRatio – золотое отношение.

ВЫЧИСЛЕНИЕ ФУНКЦИЙ

В качестве примера приведем вычисление функции sin(x). Пусть мы хотим вычислить значение sin (x) при x=0.45. Набираем `Sin[0.45]`, Нажимаем Shift+Enter и получаем ответ. На экране после выполнения команды получаем следующее:

```
In[1]:= Sin[0.45]
Out[1]= 0.434966
```

Здесь выражения `In[1]:=` и `Out[1]=` создаются самой программой.

Если есть необходимость одновременно вычислить значения функции при нескольких значениях переменной, то это делается таким образом:

```
In[2]:= Sin[{0.1, 0.2, 0.3}]
Out[2]= {0.0998334,0.198669,0.29552}
```

В этом случае вычисление функции производится для всего списка (List) аргументов

x = `{0.1, 0.2, 0.3}`.

Список функций, входящих в пакет весьма велик. Приведем перечень некоторых из них:

1. Тригонометрические функции:

`Sin[x],Cos[x],Tan[x],ArcSin[x],ArcCos[x],ArcTan[x];`

Здесь аргумент x измеряется в радианах. Если аргумент измеряется в градусах, то вместо x следует записать `x Degree`. Можно также записать `x°`. Для этого надо последовательно нажать x Esc deg Esc.

2. Гиперболические функции:

`Sinh[x], Cosh[x], Tanh[x], ArcSinh[x],ArcCosh[x], ArcTanh[x];`

3. Логарифмические и степенные функции:

`Log[x]` – вычисляет натуральный логарифм; `Log[b, x]` – вычисляет логарифм по основанию b; `Log10[x]` и `Log2[x]` –вычисляет логарифмы по основаниям 10 и 2, соответственно; `x^y` – вычисляет x^y; `Exp[x]` – вычисляет экспоненту; `Sqrt[x]` – квадратный корень.

4. Специальные функции:

`Gamma[x]` – вычисляет гамма функцию:

```
In[3]:= Gamma[3.2]
Out[3]= 2.42397
```

`BesselJ[n, x]` и `BesselK[n, x]` – функции Бесселя $J_n(x)$ и $K_n(x)$, соответственно:

```
In[4]:= BesselK[2,2.1]
Out[4]= 0.217685
```

`SinIntegral[x]` – интегральный синус:

```
In[5]:= SinIntegral[0.7]
Out[5]= 0.681222
```

`Erf[x]` – функция ошибок:

```
In[6]:= Erf[2.3]
Out[6]= 0.998857
```

Описание других специальных функций можно найти в центре документации пакета.

Вычисление функций обладает некоторыми особенностями. Если вы попробуете, например, вычислить `Sin[4]`, то в ответ получите точно такое же выражение. То же самое будет при вычислении `Sin[3/17]`:

```
In[7]:= Sin[3/17]
Out[7]= Sin[3/17]
```

Дело в том, что числа типа Integer и Rational воспринимается как число, имеющее **точное** числовое значение. Подразумевается, что в этом случае должно быть получено **точное** значение функции. Если это сделать невозможно, программа возвращает исходное выражение. Если же это возможно – выдается ответ. Например,

```
In[8]:= ArcSin[1/2]
```

```
Out[8]= π/6
```

Если же аргумент является вещественным числом, то выдается численный ответ. При этом точность ответа зависит от точности задания аргумента. Если точность аргумента меньше 18 десятичных знаков после запятой, то точность ответа будет 10^{-6}:

```
In[9]:= Sin[0.34567867867887899]
Out[9]= 0.338835
```

Если же точность аргумента больше 18 знаков, то точность ответа совпадает с точностью задания аргумента:

```
In[10]:= Sin[0.345678678678878999]
Out[10]= 0.338835287156166419
```

Если мы хотим вычислить значение функции при целом или рациональном значении аргумента, то следует использовать функцию получение численного ответа N[], где в качестве аргумента стоит вычисляемая функция:

```
In[11]:= N[Sin[3/17]]
Out[11]= 0.175556
```

Если значение функции следует получить с большой точностью, то следует использовать команду `N[функция, необходимое число значащих цифр]`:

```
In[12]:= N[Sin[3/17], 25]
Out[12]= 0.1755560760690451928597273
```

В качестве аргумента любой функции может быть комплексное число. В этом случае значение функции также будет комплексным числом:

```
In[13]:= Sin[2.3+0.2 I]
Out[13]= 0.760669-0.134145 I
```

В следующем примере вычисляется значение функции Бесселя первого порядка для значения аргумента 12+I с точностью 20 знаков после запятой:

```
In[14]:= N[BesselJ[1, 12 + I], 20]
Out[14]= -0.34045355876634445954+0.08099594495833409510 I
```

Обратите внимание, что в следующем примере значение этой функции при том же значении аргумента не вычисляется, так как обе его части являются целыми числами:

```
In[15]:= BesselJ[1, 12 + I]
Out[15]= BesselJ[1,12+I]
```

Если же хотя бы одна (вещественная или мнимая) часть аргумента является вещественным числом, то в результате получим ответ:

```
In[16]:= BesselJ[1, 12. + I]
Out[16]= -0.340454+0.0809959 I
```

Если необходимо получить вещественную или мнимую часть значения любой функции или абсолютную величину функции, то следует использовать команды Re[ ], Im[ ] и Abs[ ]:

```
In[17]:= Re[Sin[2.3+0.2 I]]
Out[17]= 0.760669
In[18]:= Im[Sin[2.3+0.2 I]]
Out[18]= -0.134145
In[19]:= Abs[Sin[2.3+0.2 I]]
Out[19]= 0.772407
```

СОЗДАНИЕ НОВЫХ ФУНКЦИЙ

Стандартная запись создаваемой функции имеет вид: `f[x]=Выражение`

Например, мы хотим ввести функцию $y(x)=\frac{\sin^2(x)}{1+x+x^2}\cdot e^{-3x}$. Для этого набираем следующее выражение: `y[x_]=Sin[x]^2 Exp[-3 x]/(1+x+x^2)`. После выполнения (одновременное нажатие Shift и Enter) получим

```
In[1]:= y[x_]=Sin[x]^2 Exp[-3 x]/(1+x+x^2)
                  -3 x
                 e      Sin[x]^2
Out[1]=  -------------------------
                      2
             1 + x + x
```

Шаблон x_ означает, что вместо x в качестве аргумента может быть использовано любое число или произвольный аргумент. Например:

```
In[2]:= y[x^3]
                  -3 x^3
                 e       Sin[x^3]^2
Out[2]=  ---------------------------
                      3    6
             1 + x^3 + x
```

Теперь мы можем использовать эту функцию для выполнения численных или аналитических расчетов.

Как можно видеть запись выражений в пакете имеет стандартную форму. Заметим, однако, что можно использовать только круглые скобки, так как знаки фигурных и квадратных скобок зарезервированы для создания списков (фигурные) и обозначения функций y[x]. Кроме того, знак умножения (*) можно заменять пробелом.

Вместо знака равенства можно использовать знак отложенного равенства:

```
In[3]:= y[x_]:=Sin[x]^2 Exp[-3 x]/(1+x+x^2)
```

В этом случае правая часть выражения `In[23]` вычисляется не сразу (выход `Out[3]` отсутствует), а только при обращении к этой функции.

Аналогично можно ввести функции нескольких аргументов:

```
In[4]:= f[x_, y_]=Exp[-x y^2]
                 -x y^2
Out[4]= e
```

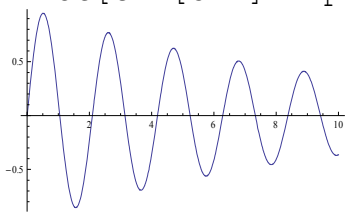
После этого введенную функцию можно использовать обычным образом. Например:

```
In[5]:= f[a+b, x]
                 -(a+b) x^2
Out[5]= e
```

ПОСТРОЕНИЕ ГРАФИКОВ

Для построения графиков функций одного аргумента используется команда `Plot`. В простейшем случае она имеет форму `Plot[выражение функция, {x, x`$_{нач}$`, x`$_{кон}$`}]`. Здесь x- обозначение аргумента, $x_{нач}$ и $x_{кон}$ - начальное и конечное значение аргумента при построении графика. Например, пусть необходимо построить график функции $y(x)=\sin(3x)e^{-0,1x}$ в интервале изменения аргумента x от 0 до 10. Для этого набираем

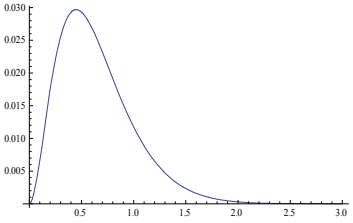
```
In[1]:= Plot[Sin[3 x] Exp[-0.1 x], {x, 0, 10}]
```




```
Out[1]=
```

Если мы ранее определили функцию y(x) (см `In[20]`), то для построения ее графика в итервале изменения аргумента x от 0 до 3 записываем:

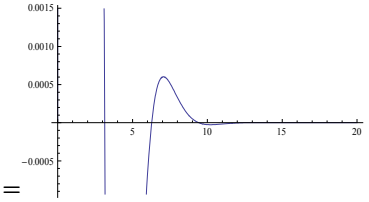
```
In[2]:=Plot[y[x], {x, 0, 3}]
```




```
Out[2]=
```

Если построить график функции $\sin(x)e^{-x}$ в интервале изменения x от 0 до 20, то в результате получим следующий график:

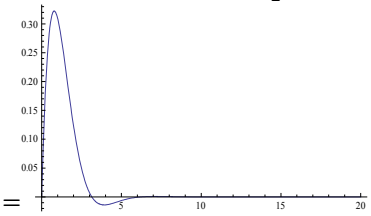
```
In[3]:=Plot[Sin[x]Exp[- x], {x, 0, 20}]
```




```
Out[3]=
```

Легко видеть, что на графике представлена только часть кривой. Это обусловлено тем, что программа пытается по умолчанию выделить наиболее характерные мелкомасштабные изменения функции (в данном случае – осцилляции в области 5 < x < 10). Для того, чтобы получить **весь** график необходимо включить опцию `PlotRange→All` (стрелка набирается путем последовательного нажатия знака минус (-) и знака >). В результате получим:

```
In[4]:=Plot[Sin[x]Exp[- x], {x, 0, 20}, PlotRange→All]
```




```
Out[4]=
```

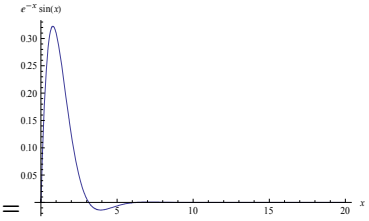
Как видно из этого графика теперь осцилляции в интервале 5 < x < 10 практически не заметны.

У команды `Plot` имеется еще ряд опций. Приведем лишь некоторые из них.

Опция `AxesLabel→{x, y}` ставит по осям обозначения, указанные в фигурных скобках. Например следующая команда выдает график с обозначениями по осям

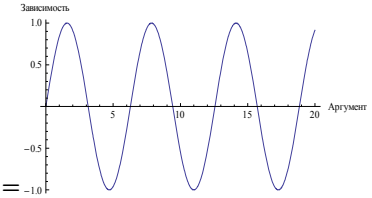
```
In[5]:=Plot[Sin[x]Exp[- x], {x, 0, 20}, PlotRange→All,
AxesLabel→ {x, Sin[x] Exp[-x]}]
```

Out[5]=



Результат выполнения следующей команды приведен на рисунке:

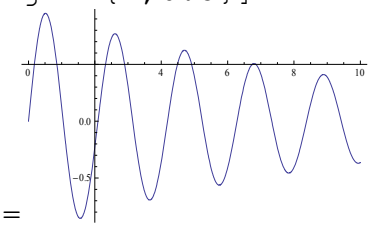
```
In[6]:=Plot[Sin[x], {x, 0, 20}, AxesLabel→{Аргумент, Зави-
симость}]
```



Out[6]=

Опция `AxesOrigin→{x0,y0}` указывает точку пересечения осей x и y.

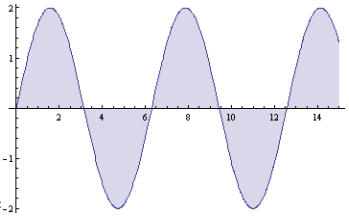
```
In[7]:=Plot[Sin[3 x] Exp[-0.1 x], {x, 0, 10},
AxesOrigin→{2,0.5}]
```



Out[7]=

Опция `Filling` позволяет закрашивать график под кривой. `Filling->Axis` заштриховывает область между графиком и осью:

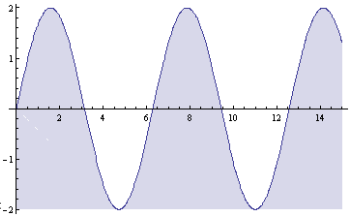
```
In[8]:=Plot[2Sin[x],{x,0,15}, Filling→Axis]
```



Out[8]=

`Filling->Bottom` закрашивает область между кривой и нижним краем рисунка:

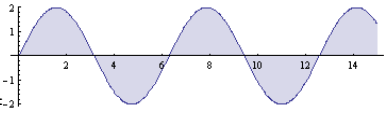
```
In[9]:=Plot[2Sin[x],{x,0,15},Filling→Bottom]
```



Out[9]=

Опция `AspectRatio -> Automatic` устанавливает одинаковый масштаб по осям x и y:

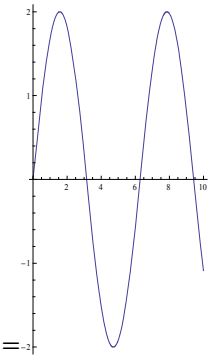
```
In[10]:=Plot[2Sin[x],{x,0,15},Filling→Axis,
AspectRatio→Automatic]
```



Out[10]=

Опция `AspectRatio -> k` делает отношение высоты графика к его ширине (не масштаб по осям) равным k:

```
In[11]:=Plot[2Sin[x],{x,0,10}, AspectRatio→2]
```

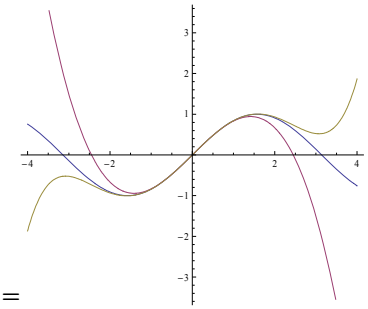


```
Out[11]=
```

Ряд других опций позволяет менять цвет и вид кривых (пунктир, штрих-пунктир) кривых, наносить сетку, выбирать шрифт надписей и т.д.

Если на одном графике необходимо построить зависимости от x нескольких функций, то из этих функций создают список, а затем к этому списку применяют функцию `Plot:`

```
In[12]:=Plot[{Sin[x], x-x^3/6, x-x^3/6+x^5/120}, {x, -4, 4},
AspectRatio→Automatic]
```

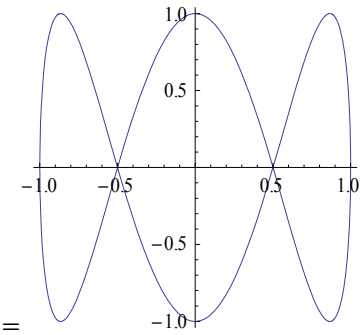


```
Out[12]=
```

На данном рисунке построены графики трех функций: $\sin(x)$, $x-x^3/6, x-x^3/6+x^5/120$.

Функция `ParametricPlot[{`f_x`,`f_y`},{`t`,`t_{min}`,`t_{max}`}]` строит параметрический график, в котором координаты x и y определяются, соответственно, функциями f_x и f_y, зависящими от параметра t. Значение t изменяется от t_{min} до t_{max}. Например, график построенный ниже дает пример фигуры Лиссажу при отношении частот 3:1.

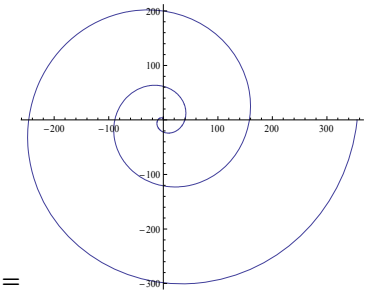
```
In[13]:=ParametricPlot[{Sin[t],Cos[3t]},{t,0,2Pi}]
```




```
Out[13]=
```

Функция `PolarPlot[f[θ],{θ, θmin, θmax}]` строит в полярных координатах кривую `r = f[θ]`. Например, на следующем графике построена спираль

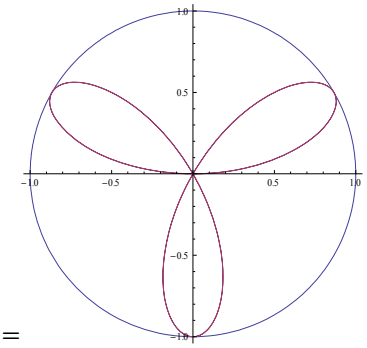
```
In[14]:=PolarPlot[t^2, {t, 0, 6Pi}]
```



```
Out[14]=
```

Следующая команда строит два графика – окружность и трилистник:

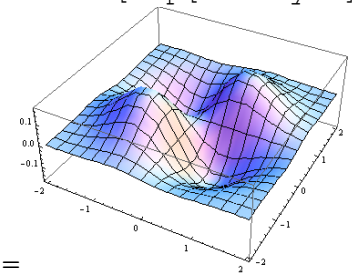
```
In[15]:=PolarPlot[{1, Sin[3 t]}, {t, 0, 2Pi}]
```



```
Out[15]=
```

Для представления графиков функций двух переменных $z = f(x, y)$ можно использовать несколько способов. Команда `Plot3D[f[x, y], {x, xmin, xmax}, {y, ymin, ymax}]` строит трехмерный график f как функцию x и y.При этом x и y меняются в интервале $x_{min} < x < x_{max}$ и $y_{min} < y < y_{max}$, соответственно.

```
In[16]:=Plot3D[Exp[-x^2-y^2]Sin[x y], {x, -2, 2}, {y, -2, 2}]
```

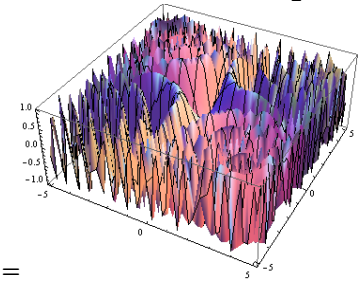


```
Out[16]=
```

Среди многочисленных опций этой команды отметим следующие:

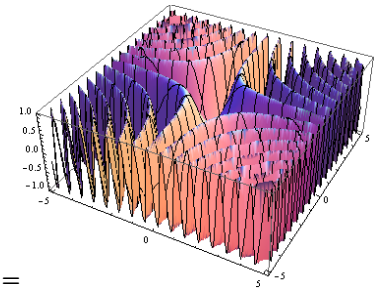
`PlotPoints->n`, которая строит график используя по n точек в каждом направлении x и y. Данную опцию следует использовать для сильно изрезанной поверхности, когда число точек по умолчанию (n=20) недостаточно для правильного отображении поверхности. Сравните результаты выполнения следующих команд:

```
In[17]:=Plot3D[Sin[2x y], {x, -5, 5}, {y, -5, 5}]
```



```
Out[17]=
```

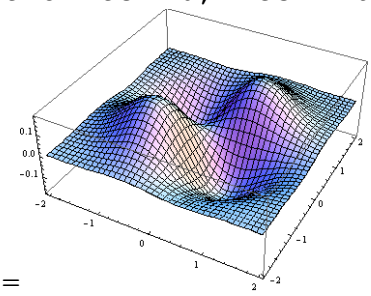
```
In[18]:=Plot3D[Sin[2x  y],  {x,  -5,  5},  {y,  -5,  5},
PlotPoints->50]
```



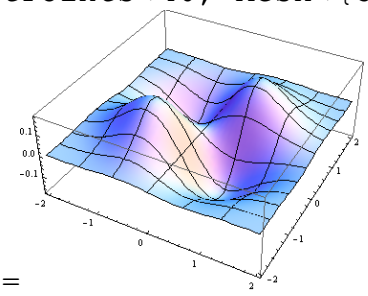
```
Out[18]=
```

По умолчанию число выводимых делений сетки независимо от опции `PlotPoints` составляет 15 в каждом направлении. Если такого числа недостаточно, то можно использовать опцию `Mesh->Full`, которая формирует сетку, число делений которой определяется числом точек в опции `PlotPoints`, или опцию `Mesh->{k, m}`, которая указывает число линий в направлении x и y:

```
In[19]:=Plot3D[Exp[-x^2-y^2]Sin[x  y],  {x,  -2,  2},  {y,  -2,
2}, PlotPoints→40, Mesh→Full]
```



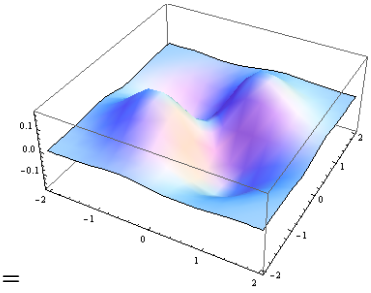
```
Out[19]=
In[20]:=Plot3D[Exp[-x^2-y^2]Sin[x  y],  {x,  -2,  2},  {y,  -2,
2}, PlotPoints→40, Mesh→{5, 10}]
```



```
Out[20]=
```

Если вывод сетки не нужен, следует использовать опцию `Mesh->False:`

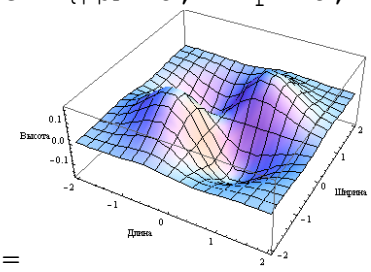
```
In[21]:=Plot3D[Exp[-x^2-y^2]Sin[x  y],  {x,  -2,  2},  {y,  -2,
2}, Mesh→False]
```



```
Out[21]=
```

Для вывода информации об обозначении осей следует использовать опцию `AxesLabel:`

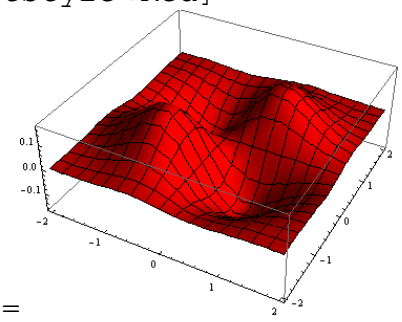
```
In[22]:=Plot3D[Exp[-x^2-y^2]Sin[x y], {x,-2, 2}, {y, -2, 2},
AxesLabel→{Длина, Ширина, Высота}]
```



Out[22]=

Опция PlotStyle позволяет изменять цвет выводимой поверхности:

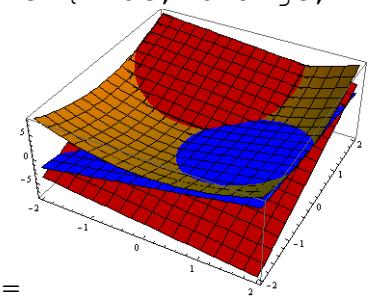
```
In[23]:=Plot3D[Exp[-x^2-y^2]Sin[x y], {x, -2, 2}, {y, -2,
2}, PlotStyle→Red]
```



Out[23]=

Как и в случае построения графиков функции одной переменной можно строить на одном рисунке одновременно несколько функций. При этом каждый из них можно закрашивать своим цветом

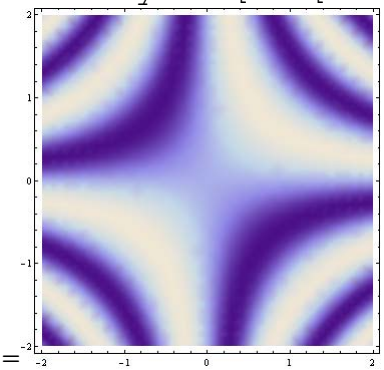
```
In[24]:=Plot3D[{2x-y, x^2+y^2,-x+3 y},{x,-2, 2},{y, -2, 2},
PlotStyle→{Blue, Orange, Red}]
```



Out[24]=

Команда DensityPlot позволяет выводить информацию о зависимости f(x,y) путем изменения плотности изображения:

```
In[25]:=DensityPlot[Sin[3 x y], {x, -2, 2}, {y, -2, 2}]
```

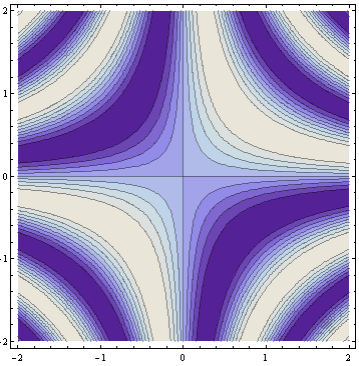


Out[25]=

Команда ContourPlot позволяет построить линии постоянных значений функции:

```
In[26]:=ContourPlot[Sin[3 x y],{x,-2, 2},{y,-2,2}]
```

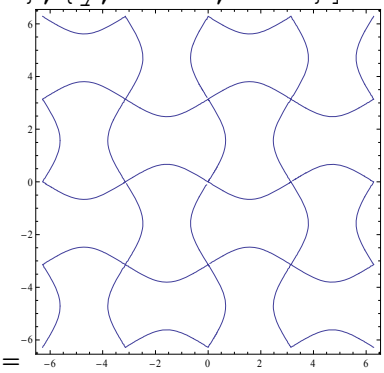
Out[26]=



Команда `ContourPlot` позволяет также на плоскости (x, y) строить линии, на которых равны значения двух функций f(x,y)=g(x,y):

```
In[27]:=ContourPlot[Sin[x]Sin[y]==Cos[x]^2     -Cos[y]^2,{x,-
2Pi,2 Pi},{y,-2 Pi,2 Pi}]
```

Out[27]=



Обратите внимание, что условие равенства двух функций обозначается знаком ==, а не =.

РЕШЕНИЕ УРАВНЕНИЙ

С помощью команды Solve[уравнения, переменные] аналитически решается уравнение или система уравнений для указанных переменных. Например, система уравнений $\begin{cases} ax^2 + y^2 = 1 \\ x + y = 1 \end{cases}$ решается с помощью команды

```
In[1]:= Solve[{a x^2 +y^2== 1, x+y==1}, {x, y}]
```

Out[1]= $\left\{\{y \to 1, x \to 0\}, \left\{y \to \frac{-1+a}{1+a}, x \to \frac{2}{1+a}\right\}\right\}$

Из результата видно, что существует два решения: $x = 0, y = 1$ и $x = \frac{2}{1+a}, y = \frac{-1+a}{1+a}$.

Обратим внимание, что знак равенства в уравнениях имеет вид == (два знака равенства).

Следующая команда

```
In[2]:= Solve[5 x^4+x^3-6==0, x]
```

Out[2]=

$\left\{\{x \to 1\}, \left\{x \to -\frac{2}{5} + \frac{2^{1/3}}{5} - \frac{3\,2^{2/3}}{5}\right\}, \left\{x \to -\frac{2}{5} + \frac{3\left(1 - i\sqrt{3}\right)}{5\,2^{1/3}} - \frac{1 + i\sqrt{3}}{5\,2^{2/3}}\right\}, \left\{x \to -\frac{2}{5} - \frac{1 - i\sqrt{3}}{5\,2^{2/3}} + \frac{3\left(1 + i\sqrt{3}\right)}{5\,2^{1/3}}\right\}\right\}$ находит

все корни уравнения 4-ой степени $5x^4 + x^3 - 6 = 0$.

Для решения системы линейных уравнений также используется команда Solve. Например,

система 3-х линейных уравнений $\begin{cases} 3x_1 + 5x_2 - 4x_3 = 2 \\ 2x_1 + x_2 - 6x_3 = 5 \\ 7x_1 + x_2 + 4x_3 = 7 \end{cases}$ решается с помощью команды

```
In[3]:= Solve[{3 x1 + 5x2-4 x3==2, 2 x1+x2- 6 x3==5, 7 x1+x2 +4
x3==7}, {x1, x2, x3}]
Out[3]= {{x1->141/100,x2->-(17/20),x3->-(101/200)}}
```

Команда NSolve численно решает систему произвольных алгебраических уравнений. Ее использование аналогично использованию команды Solve. Решение системы уравнения

$\begin{cases} x^5 + 2y^3 = 30 \\ -3x^2 + y^4 = 5 \end{cases}$ осуществляется таким образом:

```
In[4]:= NSolve[{x^5+2y^3==30,y^4-3x^2==5}, { x, y}]
Out[4]= {{x→2.17172,y→-2.09188},
         {x→-1.54718-0.954777 i,y→1.86356 +0.355288 i},
         {x→-1.54718+0.954777 i,y→1.86356 -0.355288 i},
         {x→0.639297 -2.00077 i,y→0.926483 +1.49718 i},
         {x→0.639297 +2.00077 i,y→0.926483 -1.49718 i},
         {x→0.492353 -1.91039 i,y→-1.39346+0.911647 i},
         {x→0.492353 +1.91039 i,y→-1.39346-0.911647 i},
         {x→-1.41583-1.22564 i,y→-0.468649+1.81228 i},
         {x→-1.41583+1.22564 i,y→-0.468649-1.81228 i},
         {x→1.99672 +0.214426 i,y→-0.0769252+2.02962 i},
         {x→1.99672 -0.214426 i,y→-0.0769252-2.02962 i},
         {x→-1.78018+1.12845 i,y→0.420271 +1.95881 i},
```

```
{x→-1.78018-1.12845 i,y→0.420271 -1.95881 i},
{x→0.568826 +1.75796 i,y→-0.801577+1.40509 i},
{x→0.568826 -1.75796 i,y→-0.801577-1.40509 i},
{x→0.740346 -1.8425 i,y→1.52084 -0.820516 i},
{x→0.740346 +1.8425 i,y→1.52084 +0.820516 i},
{x→-1.64564-1.3305 i,y→-1.91156-0.50556 i},
{x→-1.64564+1.3305 i,y→-1.91156+0.50556 i},
{x→1.73085,y→1.9339}}
```

Всего имеется 20 различных решений.

Эта же команда численно решает тригонометрические и иррациональные уравнения.

Система уравнений $\begin{cases}\sqrt{x}+y^{1/3}=7\\ \sqrt{x-y}=3\end{cases}$ имеет единственное решение:

```
In[5]:= NSolve[{Sqrt[x]+y^(1/3)==7,Sqrt[x-y]==3},{x,y}]
Out[5]= {{x->21.7509,y->12.7509}}
```

Тригонометрическое уравнение $\sin^2 x+3\cos x=0$ решается таким образом.

```
In[6]:= NSolve[Sin[x]^2+3 Cos[x]==0, x]
During evaluation of In[6]:= Solve::ifun: Inverse functions
are being used by Solve, so some solutions may not be found;
use Reduce for complete solution information. >>
Out[6]= {{x->1.8784},{x->0.-1.86416 I},{x->0.+1.86416
I},{x->1.8784}}
```

Однако при этом выдается сообщение о том, что вследствие использования обратной функции некоторые решения могут быть пропущены. В этом случае рекомендуется использовать команду `Reduce`.

Команда `Reduce` является весьма полезной при решении различных уравнений и систем, но так как ее использование требует значительных затрат машинного времени ее следует использовать только когда вышеприведенные команды не дают результата.

Применение команды `Reduce` к вышеприведенному тригонометрическому уравнению позволяет аналитически решить его:

```
In[7]:= Reduce[Sin[x]^2+3 Cos[x]==0, x]
```

```
Out[7]= C[1] ∈ Integers && (x == -2 i ArcTanh[Sqrt[1/3 (-2 + Sqrt[13])]] + 2 π C[1] || x == 2 i ArcTanh[Sqrt[1/3 (-2 + Sqrt[13])]] + 2 π C[1] ||
        x == -2 ArcTan[Sqrt[1/3 (2 + Sqrt[13])]] + 2 π C[1] || x == 2 ArcTan[Sqrt[1/3 (2 + Sqrt[13])]] + 2 π C[1])
```

Таким образом, существует четыре различных решения – два комплексных

$x=-2i\,\mathrm{Arcth}\sqrt{\frac{1}{3}\left(-2+\sqrt{13}\right)}+2\pi C$, $x=2i\,\mathrm{Arctgh}\sqrt{\frac{1}{3}\left(-2+\sqrt{13}\right)}+2\pi C$ и два вещественных

$x=-2\,\mathrm{Arctg}\sqrt{\frac{1}{3}\left(2+\sqrt{13}\right)}+2\pi C$, $x=2\,\mathrm{Arctg}\sqrt{\frac{1}{3}\left(2+\sqrt{13}\right)}+2\pi C$, где C – целое число.

Команда `Reduce` позволяет решать также неравенства и системы неравенств, а также системы, состоящие из уравнений и неравенств. Кроме того, эта команда позволяет выделить

класс решений (например, найти только целые решения, либо вещественные). Так для того, чтобы найти целые положительные решения уравнения $x^2-7y^2=1$ необходимо записать

```
In[8]:= Reduce[{x^2-7y^2==1, x>0,y>0},{x,y},Integers]
```

Out[8]=

$$C[1]\in \text{Integers}\ \&\&\ C[1]\geq 1\ \&\&\ x=\frac{1}{2}\left(\left(8-3\sqrt{7}\right)^{C[1]}+\left(8+3\sqrt{7}\right)^{C[1]}\right)\ \&\&\ y=-\frac{\left(8-3\sqrt{7}\right)^{C[1]}-\left(8+3\sqrt{7}\right)^{C[1]}}{2\sqrt{7}}$$

В обычной записи ответ таков $x=\frac{1}{2}\left(\left(8-3\sqrt{7}\right)^k+\left(8+3\sqrt{7}\right)\right)^k$, $y=-\frac{\left(\left(8-3\sqrt{7}\right)^k-\left(8+3\sqrt{7}\right)\right)^k}{2\sqrt{7}}$, где $k\geq 1$– целое число. Так при *k=1 x= 8, y=3* , при *k=2 x=127, y=48* и т.д.

Команда FindRoot[f, {x, x_0}] находит численное значение одного из корней уравнения f(x)=0, причем ищется корень ближайший к точке x=x_0. Например, для того, чтобы найти корень уравнения $\sin^7 x+\cos x=0$, ближайший к точке x=30, необходимо выполнить команду

```
In[9]:= FindRoot[Sin[x]^7+Cos[x], {x, 30}]
Out[9]= {x->30.3131}
```

Эта же команда, записанная в форме `FindRoot[f == g, {x. x0}]` находит корень уравнения f(x)=g(x) вблизи точки x=x_0.

Команда Roots[f ==g, x] находит все корни уравнения f(x) = g(x) в том случае, когда обе функции являются многочленами. Если коэффициенты многочленов рациональные числа, то, если возможно ответ выдается в аналитическом виде. Так, в последующем примере находятся все три корня уравнения $3x^3-2x+3=x^2$:

```
In[10]:= Roots[3 x^3 - 2 x + 3==x^2, x]
```

Out[10]=

$$x==\frac{1}{9}\left(1-19\left(\frac{2}{673-9\sqrt{5253}}\right)^{1/3}-\left(\frac{1}{2}\left(673-9\sqrt{5253}\right)\right)^{1/3}\right)\ ||$$

$$x==\frac{1}{9}+\frac{1}{18}\left(1+i\sqrt{3}\right)\left(\frac{1}{2}\left(673-9\sqrt{5253}\right)\right)^{1/3}+\frac{19\left(1-i\sqrt{3}\right)}{9\,2^{2/3}\left(673-9\sqrt{5253}\right)^{1/3}}\ ||$$

$$x==\frac{1}{9}+\frac{1}{18}\left(1-i\sqrt{3}\right)\left(\frac{1}{2}\left(673-9\sqrt{5253}\right)\right)^{1/3}+\frac{19\left(1+i\sqrt{3}\right)}{9\,2^{2/3}\left(673-9\sqrt{5253}\right)^{1/3}}$$

Если хотя бы один из коэффициентов имеет тип Real, выдаются численные значения корней:

```
In[11]:= Roots[3 x^3 - 2. x + 3==x^2, x]
Out[11]= x==-1.09973||x==0.716531-0.629204 I||
x==0.716531+0.629204 I
```

ВЫПОЛНЕНИЕ ПРЕОБРАЗОВАНИЙ

В пакете существует множество команд для выполнения преобразований. Ниже будут рассмотрены лишь некоторые из них.

Команда Simplify[выражение] выполняет последовательность алгебраических и других возможных преобразований выражения, возвращая простейшую найденную форму:

```
In[1]:= Simplify[(x^2-1)(x^2+1)(1-x)(1+x)]
Out[1]= -1+x^2+x^4-x^6
```

Другой пример упрощения выражения $\frac{1}{3(1+x)}-\frac{-1+2x}{6(1-x+x^2)}+\frac{2}{3(1+\frac{1}{3}(-1+2x)^2)}$:

```
In[2]:= Simplify[1/(3 (1+x))-(-1+2 x)/(6 (1-x+x^2))+2/(3
(1+1/3 (-1+2 x)^2))]
Out[2]= 1/(1+x^3)
```

Если попытаться упростить такое простое выражение, как $\sqrt{x^2}$, то будет возвращено исходное выражение:

```
In[3]:= Simplify[Sqrt[x^2]]
Out[3]= Sqrt[x^2]
```

Это обусловлено тем, что результат выполнения такого преобразования зависит от знака x. Чтобы это учесть, необходимо указать предположения о знаке x. Эти предположения указываются в той же команде:

```
In[4]:= Simplify[Sqrt[x^2], x<0]
Out[4]= -x
```

Более мощная команда FullSimplify

ОПЕРАЦИИ С ВЕКТОРАМИ И МАТРИЦАМИ

n – мерный вектор в пакете создается путем формирования списка длиной n.

Для нахождения скалярного произведения векторов применяется команда `Dot` или (.). Например, следующие команды формируют два 4 – мерных вектора **a** и **b** и находят их скалярное произведение:

```
In[1]:= a={2, 3, 4, 7}
b={3, 5, 8, 2}
a.b
Out[1]= {2,3,4,7}
Out[2]= {3,5,8,2}
Out[3]= 67
```

Длина (норма) вектора находится с помощью команды `Norm`:

```
In[4]:= Norm[a],
Out[4]= Sqrt[78]
```

Для нахождения угла между двумя векторами используется команда `VectorAngle`:

```
In[5]:= VectorAngle[a, b]
Out[5]= ArcCos[67/(6 Sqrt[221])]
```

Этот же угол можно найти по другому:

```
In[6]:= ArcCos[a.b/(Norm[a]Norm[b])]
Out[6]= ArcCos[67/(6 Sqrt[221])]
```

Если необходимо найти численное значение угла, необходимо сразу после получения результата применить команду `N[%]`. Знак процента означает, что в качестве аргумента команды используется результат вычислений в предыдущей строке:

```
In[7]:= N[%]
Out[7]= 0.720992
```

Векторное произведение двух трехмерных векторов находится командой `Cross`. Например, в следующем примере находится векторное произведение векторов, имеющих координаты (1, -2, 3) и (3, -5, 7):

```
In[8]:= Cross[{1, -2, 3},{3, -5, 7}]
Out[8]= {1,2,1}
```

Для представления векторного произведения в более привычном виде вместо команды Cross можно набрать EscCrossEsc между сомножителями:

```
In[9]:= {1, -2, 3}x{3, -5, 7}
Out[9]= {1,2,1}
```

Команда `Projection`[a, b] дает проекцию вектора **a** на направление вектора **b**. Например, в следующем примере находится проекция вектора с координатами (5,6,7) на направление орта (1,0,0)

```
In[10]:= Projection[{5,6,7},{1,0,0}]
```

```
Out[10]= {5,0,0}
```

Ниже находится проекция орта (1,0,0) на направление вектора с координатами (5,6,7):

```
In[11]:= Projection[{1,0,0},{5,6,7}]
Out[11]= {5/22,3/11,7/22}
```

Матрицы определяются с помощью списка следующим образом:

{{элементы 1-ой строки, разделенные запятой}, {элементы 2-ой строки, разделенные запятой}, …{элементы последней строки, разделенные запятой}}. Например, в следующем примере определяется матрица $g = \begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 8 \end{pmatrix}$

```
In[12]:= g={{1, 2, 3}, {4,5,6}, {7, 8, 8}}
Out[12]= {{1,2,3},{4,5,6},{7,8,8}}
```

Для просмотра матрицы в стандартной форме используется команда `MatrixForm`

```
In[13]:= MatrixForm[g]
Out[13]//MatrixForm=
```

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

Операция умножения матриц выполняется с помощью команды `Dot[]`. Например

```
In[14]:= Dot[g, g]
Out[14]= {{30,36,48},{66,81,108},{116,142,190}}
```

умножает матрицу g на самое себя.

Вместо этой команды можно использовать операцию умножения

```
In[15]:= g.g
Out[15]= {{30,36,48},{66,81,108},{116,142,190}}
```

Отметим, что в данном случае нельзя использовать пробел вместо операции умножения. При использовании пробела будут перемножаться соответствующие элементы матриц:

```
In[16]:= g g
Out[16]= {{1,4,9},{16,25,36},{49,64,121}}
```

Определитель матрицы вычисляется с помощью операции `Det[]`:

```
In[17]:= Det[g]
Out[17]= -6
```

Сформируем, например, две квадратные матрицы размером 4*4.

```
In[18]:= z={{a11, a12, a13, a14},{a21,a22, a23, a24}, {a31,
a32, a33, a34},{a41, a42, a43, a44}}
Out[18]=
{{a11,a12,a13,a14},{a21,a22,a23,a24},{a31,a32,a33,a34},{a41,
a42,a43,a44}}
In[19]:= h={{b11,b12,b13,b14},{b21,b22,b23,b24},{b31, b32,
b33, b34},{b41, b42, b43, b44}}
Out[19]= {{b11,b12,b13,b14},{b21,b22,b23,b24},
{b31,b32,b33,b34},{b41, b42,b43,b44}}
```

Проверим что определитель произведения двух матриц равен произведению определителей.

```
In[20]:= Det[h.z]-Det[h] Det[z]
Out[20]= -((-(a14 b11+a24 b12+a34 b13+a44 b14) (a13 b21+a23
b22+a33   (не приводим длинное выражение)   +b11 b22 b33 b44)
In[21]:= Simplify[%]
Out[21]= 0
```

Для того, чтобы исключить промежуточный результат следует поставить знак (`;`) в конце строки `In[20]:=` . В этом случае результат будет существенно короче.

```
In[22]:= Det[h.z]-Det[h] Det[z];
In[23]:= Simplify[%]
Out[23]= 0
```

Обратная матрица вычисляется с помощью команды Inverse[ ]:

```
In[24]:= Inverse[g]
Out[24]= {{-(7/6),-(1/3),1/2},{1/3,5/3,-1},{1/2,-1,1/2}}
```

Если перемножить матрицу g и обратную к ней получим единичную матрицу:

```
In[25]:= g.Inverse[g]
Out[25]= {{1,0,0},{0,1,0},{0,0,1}}
In[26]:= MatrixForm[%]
```

$$\text{Out[26]//MatrixForm=} \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

Собственные значения матрицы находятся с помощью команды Eigenvalues[ ]:Например, для того, чтобы найти собственные значения введенной ранее матрицы g с точностью 10 знаков, необходимо выполнить следующие команды:

```
In[27]:= Eigenvalues[g];
In[28]:= N[%, 10]
Out[28]= {17.32630760,-0.7738201326,0.4475125360}
```

Собственные вектора матрицы находятся с помощью команды `Eigenvectors`[ ]:

```
In[29]:= Eigenvectors[g];
In[30]:= N[%]
Out[30]= {{0.29983,0.682059,1.},{-0.963758,-
0.334138,1.},{1.57302,-2.39338,1.}}
```

Так как элементы матрицы g целые числа в этом случае делается попытка аналитически вычислить величину собственных векторов. Для того, чтобы не выводить получающееся громоздкое выражение в конце первой строки ставится команда `;`. После этого командой N[%] вычисляются численные значения собственных векторов.

Можно пойти несколько другим путем – сначала применить команду N[g], превращая элементы матрицы g в вещественные числа, а затем вычислить собственные векторы:

```
In[31]:= Eigenvectors[N[g]]
Out[31]= {{-0.240434,-0.546943,-0.801901},{-0.674686,-
0.233916,0.700058},{0.518535,-0.78896,0.329643}}
```

На первый взгляд, получаются различные результаты. Однако, это не так. Собственные векторы определяются с точностью до постоянного множителя. Легко видеть, что умножая первый вектор в `Out[32]=` на -0.801901 мы получим первый вектор в `Out[34]=`, и так далее.

ВЫЧИСЛЕНИЕ ПРЕДЕЛОВ

Для вычисления предела $\lim_{x\to x_0} f(x)$ используется команда Limit[f[x], x->x0]. Например, для того, чтобы найти предел $\lim_{x\to 0}\frac{(1-\cos(3x))^2}{x^4}$ необходимо выполнить следующую команду:

```
In[2]:= Limit[(1-Cos[3 x])^2/x^4, x->0]
Out[2]= 81/4
```

Предел $\lim_{z\to\infty}\left(1+\frac{x}{z}\right)^z$ находится следующей командой

```
In[3]:= Limit[(1+x/z)^z, z->Infinity]
Out[3]= e^x
```

Если функция в точке, где ищется предел, разрывная, но существуют пределы слева или справа, то эти односторонние пределы находятся с помощью опции `Assumptions`. Например, найдем право- и левосторонний предел функции $\frac{|x-5|}{x-5}$ при $x\to 5$. Левосторонний предел находится в предположении, что $x<5$:

```
In[14]:= Limit[Abs[x-5]/(x-5), x->5, Assumptions->x<5]
Out[14]= -1
```

а правосторонний предел – в предположении $x>5$:

```
In[22]:= Limit[Abs[x-5]/(x-5), x->5, Assumptions->x>5]
Out[22]= 1
```

СУММИРОВАНИЕ, ДИФФЕРЕНЦИРОВАНИЕ, ИНТЕГРИРОВАНИЕ

Вычисление **сумм** вида $\sum_{i_{\min}}^{i_{\max}} f_i$ выполняется с командой `Sum[f[i], {i, i`$_{min}$`, i`$_{max}$`}]`. Например, пусть необходимо найти $\sum_{k=1}^{\infty}\frac{1}{k^4}$. Для этого нужно записать

```
In[1]:= Sum[1/k^4, {k, 1, Infinity}]
Out[1]= π^4/90
```

Сумма кубов первых 100 чисел получается при выполнении следующей команды

```
In[2]:= Sum[k^3, {k, 1, 100}]
Out[2]= 25502500
```

Дифференцирование функций осуществляется с помощью команды

`D[функция, переменная дифференцирования]`

При выполнении данной команды всегда считается, что остальные величины, входящие в функцию, не зависят от переменной , по которой берется производная.

Например, взятие производной от функции $x^2e^{-3x^3}$ по x осуществляется так:

```
In[3]:= D[x^2 Exp[-3 x^3], x]
Out[3]= 2 e^(-3 x^3) x - 9 e^(-3 x^3) x^4
```

Чтобы упростить полученное выражение можно применить команду упрощения:

```
In[4]:= Simplify[%]
Out[4]= e^(-3 x^3) (2 x - 9 x^4)
```

Для того, чтобы найти производные старших порядков, необходимо записать команду D в форме D[функция, {переменная, порядок производной}. Например, чтобы найти производную 5-го порядка от той же функции (сразу упростим) надо написать:

```
In[5]:=Simplify[D[x^2 Exp[-3 x^3], {x, 5}]]
Out[5]= -9 e^(-3 x^3) (40 - 3240 x^3 + 17820 x^6 - 21870 x^9 + 6561 x^12)
```

Команда нахождения полной производной – `Dt[функция, переменная дифференцирования]` отличается от команды `D[]` тем, что в ней все переменные, от которых зависит функция, считаются зависящими от переменной, по которой производится дифференцирование:

```
In[6]:= Dt[a x^2, x]
Out[6]= 2 a x+x^2 Dt[a,x]
```

Действительно, $\frac{d(a(x)x^2}{dx} = 2ax + \frac{da}{dx}x^2$.

Нахождение неопределенного интеграла $\int f(x)dx$ осуществляется с помощью команды `Integrate[f[x], x]`. Так, чтобы найти $\int xe^{ax}\sin(bx)dx$, надо выполнить команду

```
In[7]:= Integrate[x Exp[a x]Sin[b x], x]
                e^(a x) (-b (-2 a + a^2 x + b^2 x) Cos[b x] + (-a^2 + b^2 + a^3 x + a b^2 x) Sin[b x])
Out[7]= ------------------------------------------------------------------------------------
                                          (a^2 + b^2)^2
```

Т.е. $\int xe^{ax}\sin(bx)dx = -be^{ax}\frac{(b^2x+a^2x-2a)\cos bx+(ab^2x+a^3x+b^2-a^2)\sin bx}{\left(a^2+b^2\right)^2}$. Нетрудно представить, сколько времени заняло бы вычисление такого интеграла вручную.

Определенный интеграл $\int_a^b f(x)dx$ находится командой `Integrate[f[x], {x, a, b}]`. В качестве пределов интегрирования может стоять и бесконечность. Например, $\int_0^\infty x^4e^{-x}dx$ находится следующим образом:

```
In[8]:= Integrate[x^4Exp[-x], {x, 0, Infinity}]
Out[8]= 24
```

Обратим внимание, что если записать команду взятия $\int_1^\infty x^4e^{-x}dx$ в виде

```
In[9]:= Integrate[x^4Exp[-x], {x, 1, Infinity}]
```

```
Out[9]= 65/e,
```

то ответ будет точным: $\int_1^{\infty} x^4 e^{-x} dx = 65/e$. Если же записать эту команду в виде

```
In[10]:= Integrate[x^4 Exp[-x], {x, 1., Infinity}]
Out[10]= 23.9122
```

то ответ будет приближенным. Это связано с тем, что, как уже обсуждалось, в первом случае нижний предел целое число (1), а во втором приближенное (1.). При этом программа сначала вычисляет неопределенный интеграл, а потом подставляет пределы интегрирования. Если неопределенный интеграл не берется, то исходный интеграл возвращается в неизменном виде. Так $\int_0^1 \frac{x}{x+3} e^{x^2} \sin(x^4) dx$ не вычисляется:

```
In[11]:= Integrate[x/(x+3) Exp[x^2]Sin[x^4], {x, 0, 1}]
Out[11]= Integrate[E^x^2 x Sin[x^4]/(3 + x), {x, 0, 1}]
```

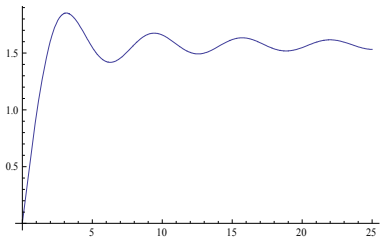
Большинство определенных интегралов не берутся в аналитическом виде. Для нахождения численного значения интеграла существует команда `NIntegrate[f[x],{x,a,b}]`. Тогда вышеприведенный интеграл сразу вычисляется:

```
In[12]:= NIntegrate[x/(x+3) Exp[x^2]Sin[x^4], {x, 0, 1}]
Out[12]= 0.0856139
```

Рассмотрим более сложную задачу. Пусть необходимо построить график зависимости от $t \in (0, 25)$ следующего важного в теории связи интеграла $f(t) = \int_0^t \frac{\sin(x)}{x} dx$. Для этого необходимо записать

```
In[13]:= Plot[NIntegrate[Sin[x]/x, {x,0,t}],{t, 0, 25},
PlotRange→All]
```

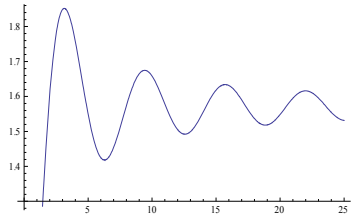
Out[13]=



Атрибут PlotRange->All поставлен для того, чтобы вывести всю кривую. Без него на графике будет представлена лишь наиболее «важная» по мнению программы, часть графика.

```
In[14]:= Plot[NIntegrate[Sin[x]/x, {x, 0, t}], {t, 0, 25}]
```

Out[14]=



РАЗЛОЖЕНИЕ ФУНКЦИЙ В СТЕПЕННЫЕ РЯДЫ

Для разложения функции f(x) в степенной ряд используется команда `Series[f,{x,x0,n}]`, где x_0 – точка, вблизи которой производится разложение, n – показатель старшей степени $(x-x_0)^n$, до которой проводится разложение. Например, разложение в ряд Тейлора функции $e^{-bx}\cos(ax)$ вблизи точки $x_0 = 0$ с точностью до x^6 выполняется так:

```
In[1]:= Series[Cos[a x]Exp[-b x], {x, 0, 6}]
```

$$\text{Out[1]}= 1 - b\,x + \frac{1}{2}\left(-a^2+b^2\right)x^2 + \left(\frac{a^2 b}{2} - \frac{b^3}{6}\right)x^3 + \frac{1}{24}\left(a^4 - 6\,a^2 b^2 + b^4\right)x^4 + \frac{1}{120}\left(-5\,a^4 b + 10\,a^2 b^3 - b^5\right)x^5 + \frac{1}{720}\left(-a^6 + 15\,a^4 b^2 - 15\,a^2 b^4 + b^6\right)x^6 + O[x]^7$$

Команда `Normal[ ]` создает обычное выражение из разложения, удаляя остаточный член:

```
In[2]:= Normal[%]
```

$$\text{Out[2]}= 1 - b\,x + \frac{1}{2}\left(-a^2+b^2\right)x^2 + \left(\frac{a^2 b}{2} - \frac{b^3}{6}\right)x^3 + \frac{1}{24}\left(a^4 - 6\,a^2 b^2 + b^4\right)x^4 + \frac{1}{120}\left(-5\,a^4 b + 10\,a^2 b^3 - b^5\right)x^5 + \frac{1}{720}\left(-a^6 + 15\,a^4 b^2 - 15\,a^2 b^4 + b^6\right)x^6$$

Рассмотрим более сложный пример. Необходимо построить в интервале $x \in [0,1]$ графики функции $e^{-2x}\cos(3x)$ и ее разложения в степенной ряд с точностью до x^6. Программа выглядит так

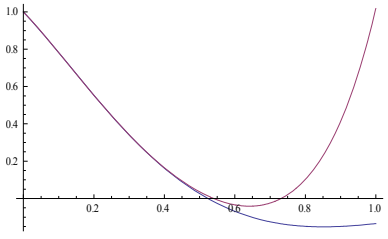
```
In[3]:= f[x_]=Exp[-2x] Cos[3 x]
        Normal[Series[f[x],{x,0,6}]]
        Plot[{f[x], %},{x,0,1}]
```

В результате получим

```
Out[3]= E^(-2 x) Cos[3 x]
```

$$\text{Out[4]}= 1 - 2\,x - \frac{5\,x^2}{2} + \frac{23\,x^3}{3} - \frac{119\,x^4}{24} - \frac{61\,x^5}{60} + \frac{407\,x^6}{144}$$

Out[5]=



Очевидно, что вплоть до $x \approx 0.5$ разложение в ряд прекрасно аппроксимирует функцию.

Разложение в степенной ряд функции $\cos(x)/x^2$ будет содержать не только положительные, но и отрицательные степени x:

```
In[6]:= Series[Cos[x]/x^2, {x, 0, 9}]
```

$$\text{Out[6]}= \frac{1}{x^2} - \frac{1}{2} + \frac{x^2}{24} - \frac{x^4}{720} + \frac{x^6}{40\,320} - \frac{x^8}{3\,628\,800} + O[x]^{10}$$

Отметим, что с полученное выражение можно использовать для дальнейших вычислений, причем результат вычислений будет также степенным рядом с тем же порядком точности. Например, можно посчитать разложение величины $\frac{1}{\left(1+\cos(x)/x^2\right)^2}-2\frac{\cos(x)}{x^2}$, используя предыдущий результат:

```
In[7]:= 1/(1+%)^2 -2%
```

$$\text{Out[7]}= -\frac{2}{x^2}+1-\frac{x^2}{12}+\frac{361\,x^4}{360}-\frac{20\,161\,x^6}{20\,160}+\frac{1\,209\,601\,x^8}{1\,814\,400}+O[x]^{10}$$

Ряды могут содержать также дробные степени:

```
In[8]:=Series[Sin[x]^(1/3),{x,0,7}]
```

$$\text{Out[8]}= x^{1/3}-\frac{x^{7/3}}{18}-\frac{x^{13/3}}{3240}-\frac{53\,x^{19/3}}{1\,224\,720}+O[x]^{22/3}$$

и даже логарифмы:

```
In[9]:= Series[x^(Sqrt[x]),{x,0,4}]
```

$$\text{Out[9]}= 1+\text{Log}[x]\sqrt{x}+\frac{1}{2}\text{Log}[x]^2\,x+\frac{1}{6}\text{Log}[x]^3\,x^{3/2}+\frac{1}{24}\text{Log}[x]^4\,x^2+\frac{1}{120}\text{Log}[x]^5\,x^{5/2}+\frac{1}{720}\text{Log}[x]^6\,x^3+\frac{\text{Log}[x]^7\,x^{7/2}}{5040}+\frac{\text{Log}[x]^8\,x^4}{40\,320}+O[x]^{9/2}$$

Разложение в степенной ряд можно использовать для оценки не берущихся определенных интегралов, зависящих от параметра. Например, $\int_0^a \sin(a\cos(x))dx$ не берется в явном виде и является функцией параметра a. Тогда, разлагая в ряд по этому параметру, получим:

```
In[10]:= Integrate[Sin[a Cos[x]],{x,0,a}]
```

$$\text{Out[10]}= \int_0^a \text{Sin[a Cos[x]]}\,dx$$

```
In[11]:= Series[%, {a, 0, 10}]
```

$$\text{Out[11]}= a^2-\frac{a^4}{3}+\frac{a^6}{10}-\frac{23\,a^8}{630}+\frac{7\,a^{10}}{648}+O[a]^{11}$$

Теперь можно использовать это выражение для оценки интеграла при различных a:

```
In[12]:= h[a_]=Normal[%]
```

$$\text{Out[12]}= a^2-\frac{a^4}{3}+\frac{a^6}{10}-\frac{23\,a^8}{630}+\frac{7\,a^{10}}{648}$$

```
In[13]:= h[0.7]
Out[13]:= 0.419932.
```

Точное значение интеграла при a = 0.7, вычисленное с помощью команды `NIntegrate` равняется 0.419896.