Символьные вычисления

```
%Назначили символьные переменные
syms x y;
%%
diff(x^2,2)
```

```
ans
= 2
```

%% Преобразования символьных выражений

%% раскрытие скобок expand

```
syms x y;
f = (x+2)*(x+3) + (y+5);
f = expand(f)
```

```
f =

11 + y + 5*x + x^2
```

%% Функция упрощения simplify

```
syms x y;
f = (x*y^3 + 2*y + x^4*y)/(y);
f = simplify(f)
```

```
f =

2 + x*y^2 + x^4
```

```
%% Функция разложения на множители factor
syms x;
f = (x+4)^4 + x^2 + (x-2)^3;
f = factor(f)
```

```
f = 

248 + 268*x + 91*x^2 + 17*x^3 + x^4
```

```
%% Вычисление значения символьных выражений
syms x;
f = (x^4 + 17*x^3 + 91*x^2 + 268*x) / 248;
f = subs(f, x, 3) %заменяем все x на 3
```

```
f = 

2163/248
```

```
f = double(f)
```

```
f = 
    8.7218
```

```
%% Если у функции несколько переменных, то придется
использовать subs несколько раз
syms x y;
f = (x+4)^4 + x^2 + (y-2)^3;
f = double(subs(subs(f, x, 2), y, 3))
```

```
f =

        1301
```

```
%% Символьное дифференцирование в Matlab
f = x^4 + 17*x^3 + 91*x^2 + 268*x + 248;
diff(f)
```

```
ans =

268 + 182*x + 51*x^2 + 4*x^3
```

```
%% Функция от нескольких переменных
syms x y;
f = 2 * x ^ 5 + x * y + y^3;
diff(f, 'x')
diff(f, 'x',2)
```

```
ans = y + 10*x^4

ans = 40*x^3
```

```matlab
%% Функция от нескольких переменных
syms x y;
f = 2 * x ^ 5 + x * y + y^3;

diff(f, 'y')
diff(f, 'y',2)
```

```
ans = 3*y^2 + x

ans = 6*y
```

```matlab
%% Функция от нескольких переменных
clc,syms x y;
f = x^2 * y^2;

diff(f, 'x','y')
diff(diff(f, 'x',2),'y',2)
```

```
ans = 4*x*y

ans = 4
```

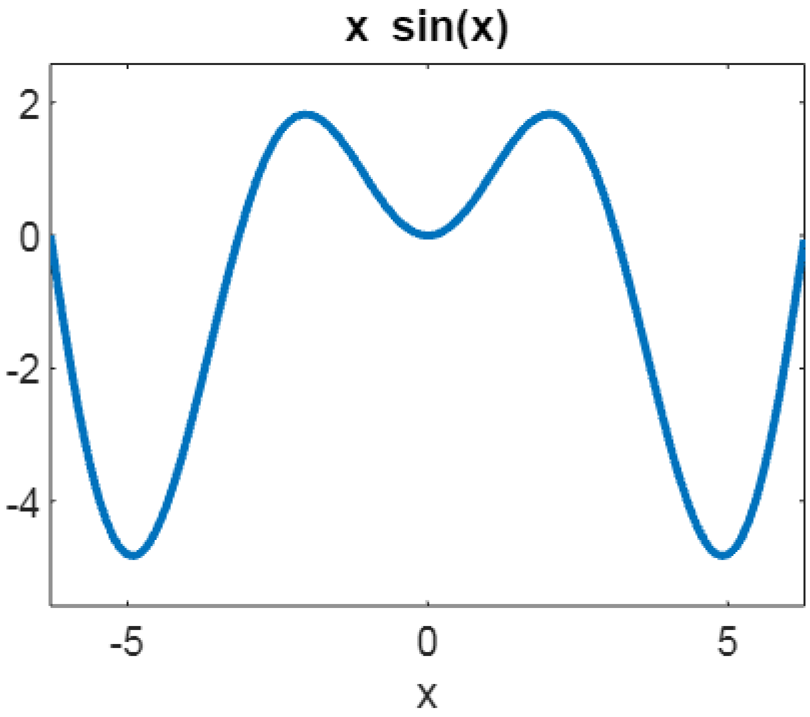
```
%% Символьное интегрирование
% Символьное интегрирование в Matlab выполняется
оператором int
    clear, clc
    syms x
    f = 4*x^3 + 51*x^2 + 182*x + 268;
    int(f)
```

```
    ans = 268*x + 91*x^2 + 17*x^3 + x^4
```

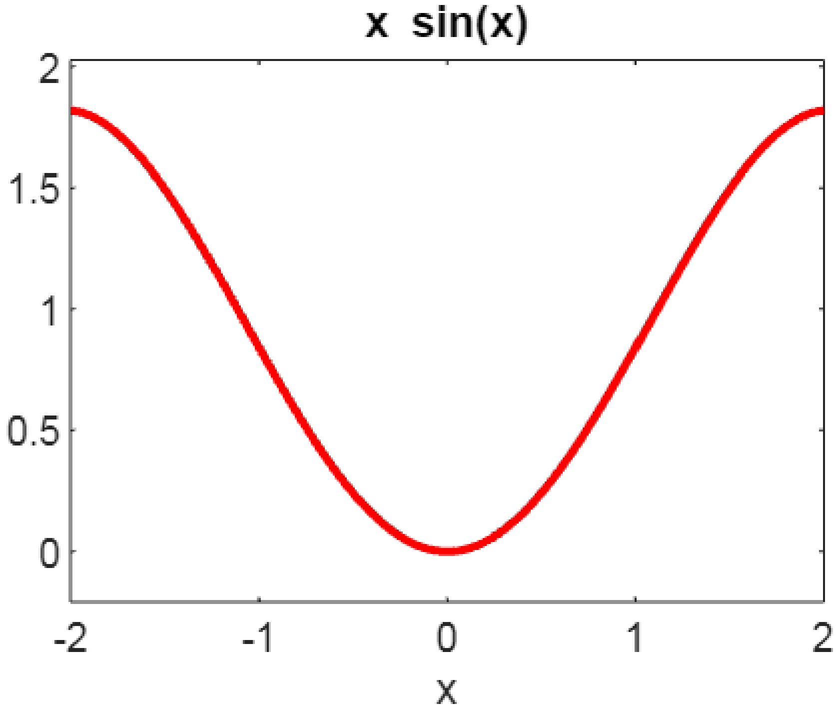
```
%% Расчет определенного интеграла

f = 4*x^3 + 51*x^2 + 182*x + 268;
%определенный интеграл на промежутке от 0 до 4
int(f, 0, 4)
```

```
    ans = 3872
```

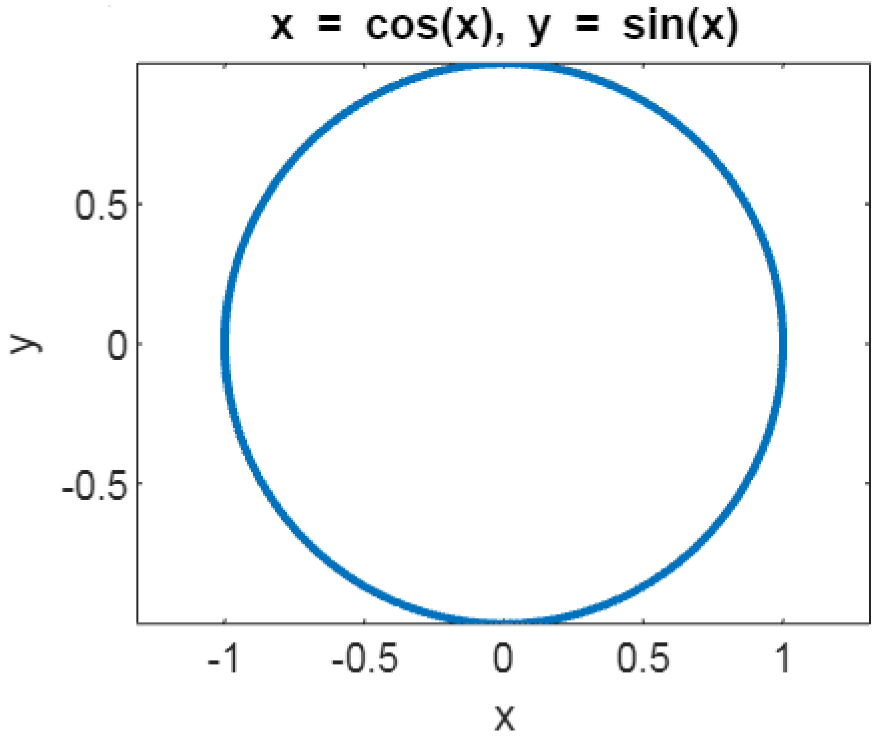
```
%% Построение графика функции
h=ezplot('x*sin(x)')
h.set('LineWidth',2)
```



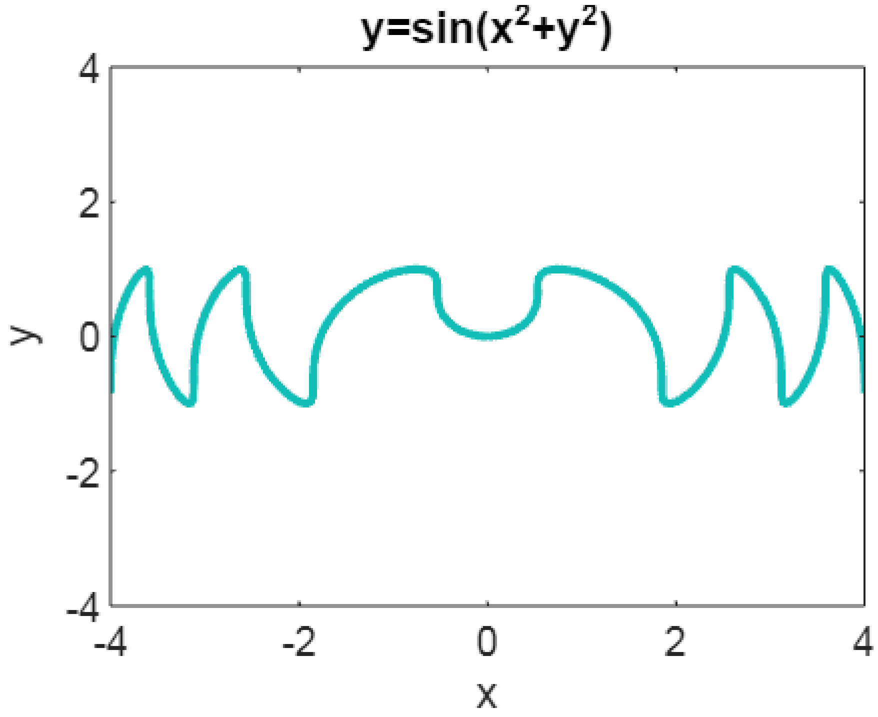
```
%% задание промежутка
h=ezplot('x*sin(x)',[-2,2])
h.set('LineWidth',2)
h.set(Color='r')
```



```
%% Параметрический график
h=ezplot('cos(x)','sin(x)')
h.set('LineWidth',2)
```



```
%% Неявно заданная функция
h=ezplot('y=sin(x^2+y^2)',[-4,4])%
h.set('LineWidth',2)
```



```
%% разложение символьной функции в ряд Тейлора
syms x
T1 = taylor(exp(x))
T2 = taylor(sin(x))
T3 = taylor(cos(x))
```

```
T1 = 1 + x + x^2/2 + x^3/6 + x^4/24 + x^5/120

T2 = x - x^3/6 + x^5/120

T3 = 1 - x^2/2 + x^4/24
```

```
%% изменить выходной порядок символьных полиномов
% ascend – по возрастанию
sympref('PolynomialDisplayStyle','ascend');
```

```
T1 = 1 + x + x^2/2 + x^3/6 + x^4/24 + x^5/120
```

```
%% Восстановить значения по умолчанию
sympref('default');
```

```
T1 = x^5/120 + x^4/24 + x^3/6 + x^2/2 + x + 1
```

```
%% ExpansionPoint - точка разложения
syms x
T = taylor(log(x), x, 'ExpansionPoint', 1)
```

```
T = x - (x - 1)^2/2 + (x - 1)^3/3 - (x - 1)^4/4
                                 + (x - 1)^5/5 – 1
```

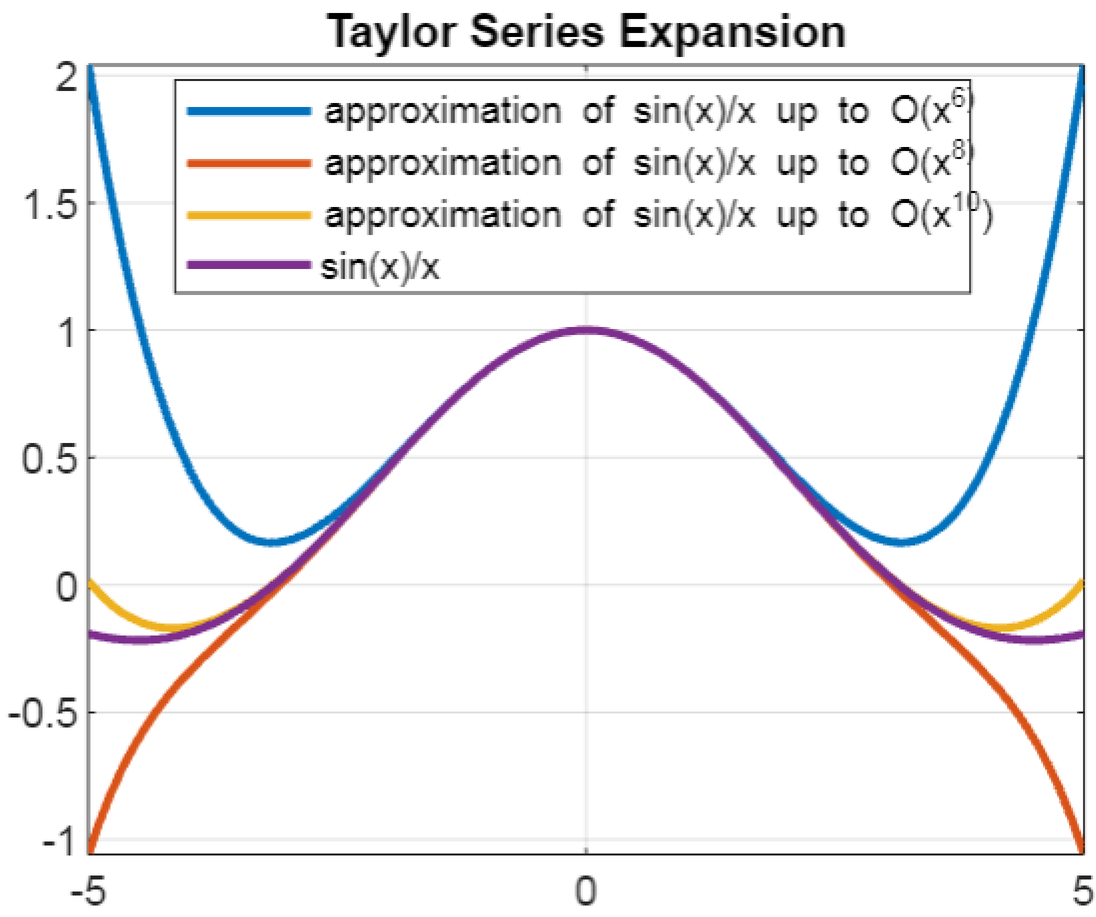
```
%% точка разложения - третий параметр
T = taylor(log(x), x, 1)
```

```
T = x - (x - 1)^2/2 + (x - 1)^3/3 - (x - 1)^4/4
                                 + (x - 1)^5/5 - 1
```

```
%% Порядок разложения
syms x
f = sin(x)/x;
T6 = taylor(f, x)
T8 = taylor(f, x, 'Order', 8)
T10 = taylor(f, x, 'Order', 10)
```

```
T6 = x^4/120 - x^2/6 + 1

T8 = - x^6/5040 + x^4/120 - x^2/6 + 1

T10 = x^8/362880 - x^6/5040 + x^4/120 - x^2/6 + 1
```

```
%% Постройте исходное выражение f
% и его приближения T6, T8, и T10.
% Определите -
% как точность приближения зависит
% от порядка разложения.

h=fplot([T6 T8 T10 f])
h.set('LineWidth',2)
xlim([-5 5])
grid on
legend('approximation of sin(x)/x up to O(x^6)',...
    'approximation of sin(x)/x up to O(x^8)',...
    'approximation of sin(x)/x up to O(x^{10})',...
    'sin(x)/x','Location','Best')
title('Taylor Series Expansion')
```



```
%% Для некоторых выражений относительный порядок
% разложения обеспечивает более точные приближения.

Ta = taylor(1/(exp(x)) - exp(x) + 2*x, x, 'Order', 5)
Tr = taylor(1/(exp(x)) - exp(x) + 2*x, x, 'Order', 5,
'OrderMode', 'relative')
```

```
Ta = -x^3/3


Tr =- x^7/2520 - x^5/60 - x^3/3
```

```
%% Ряд Маклорена для функций нескольких переменных
syms x y z
f = sin(x) + cos(y) + exp(z);
T = taylor(f, [x, y, z])
```

```
T = x^5/120 - x^3/6 + x + y^4/24 - y^2/2 +
z^5/120 + z^4/24 + z^3/6 + z^2/2 + z + 2
```

```
%% Точка разложения
syms x y
f = y*exp(x - 1) - x*log(y);
T = taylor(f, [x, y], [1, 1], 'Order', 3)
```

```
T =

x + (x - 1)^2/2 + (y - 1)^2/2
```

```matlab
%% Вычисление предела limit(f)
syms x h
f = sin(x)/x;
limit(f,x,0)
```

```
ans = 1
```

```matlab
%%
f = (sin(x+h)-sin(x))/h;
limit(f,h,0)
```

```
ans = cos(x)
```

```matlab
%%
f = (cos(x+h)-cos(x))/h;
limit(f,h,0)
```

```
ans = -sin(x)
```

```
%% Правые и левые пределы
syms x
f = 1/x;
limit(f,x,0,'right')
limit(f,x,0,'left')
```

```
ans = Inf

ans = -Inf
```

```
%% Поскольку предел слева не равняется пределу
справа,
% двухсторонний предел не существует.
% В этом случае, limit возвращает NaN (не номер).
limit(f,x,0)
%%
```

```
ans = NaN
```

```
%% symvar - найти символьные переменные
clc,clear
syms x y;

z=x*y;
f=x+y+z;

symvar(f)
```

```
ans = [x, y]
```

```
%% sym - создание символьных переменных,
выражений, функций, матриц

%% переменная
x = sym('x')

%% вектор
a = sym('a',[1 4])
```

```
a = [a1, a2, a3, a4]
```

```
%% переменные с помощью строки формата
% Этот синтаксис не создает символьные переменные
x_1..., x_4
% в рабочем пространстве MATLAB.
% Доступ к элементам а использование стандартных
методов индексирования.

a = sym('x_%d',[1 4])
```

```
a = [x_1, x_2, x_3, x_4]
```

```
a(1)
a(2:3)
```

```
ans = x_1

ans = [x_2, x_3]
```

```
%% символьная матрица
A = sym('A',[3 4])
```

```
A =
[A1_1, A1_2, A1_3, A1_4]
[A2_1, A2_2, A2_3, A2_4]
[A3_1, A3_2, A3_3, A3_4]
```

```
%% матрица с помощью строки формата
B = sym('x_%d_%d',4)
```

```
B =

[x_1_1, x_1_2, x_1_3, x_1_4]

[x_2_1, x_2_2, x_2_3, x_2_4]

[x_3_1, x_3_2, x_3_3, x_3_4]

[x_4_1, x_4_2, x_4_3, x_4_4]
```

```
%% многомерный массив
A = sym('a',[2 2 2])
```

```
A(:,:,1) =
[a1_1_1, a1_2_1]
[a2_1_1, a2_2_1]


A(:,:,2) =
[a1_1_2, a1_2_2]
[a2_1_2, a2_2_2]
```

```
%% символьные числа
% MATLAB сначала преобразует выражение в число с плавающей запятой,
% которое теряет точность.
% sym не может всегда восстанавливать эту потерянную точность.

%% символьные числа
inaccurate1 = sym(1/1234567)
accurate1 = 1/sym(1234567)
```

```
inaccurate1 =

7650239286923505/9444732965739290427392


accurate1 =

1/1234567
```

```
%%
accurate2 = sqrt(sym(1234567))
inaccurate2 = sym(sqrt(1234567))
%%
inaccurate3 = sym(exp(pi))
accurate3 = exp(sym(pi))
```

```
%% Большие символьные числа
inaccurateNum = sym(11111111111111111111)
accurateNum = sym('11111111111111111111')
```

```
inaccurateNum =

11111111111111110656


accurateNum =

11111111111111111111
```

```
%% используйте кавычки, чтобы создать
символьные комплексные числа
sym('1234567 + 1i')
```

```
ans =

1234567 + 1i
```

```
%% символьные выражения из Указателей на функции
pascal(3)
```

```
ans =
     1     1     1
     1     2     3
     1     3     6
```

```
h_matrix = @(x)(x*pascal(3));
sym_matrix = sym(h_matrix)
```

```
sym_matrix =

[x,   x,   x]
[x, 2*x, 3*x]
[x, 3*x, 6*x]
```

```
%% символьные выражения из Указателей на функции
h_expr = @(x)(sin(x) + cos(x));
sym_expr = sym(h_expr)
```

```
%% Установите предположения при создании переменных
x = sym('x','real');
y = sym('y','positive');
z = sym('z','rational');
t = sym('t',{'positive','integer'});
assumptions(x)
assumptions(y)
assumptions(z)
assumptions(t)
```

```
ans = in(x, 'real')

ans = 0 < y

ans = in(z, 'rational')

ans = [in(t, 'integer'), 1 <= t]
```

```
%% Создайте символьную матрицу и установите
предположения на каждом элементе той матрицы.
A = sym('A%d%d',[2 2],'positive')
assumptions(A)
%%
```

```
A =
[A11, A12]
[A21, A22]

ans =
[0 < A11, 0 < A12, 0 < A21, 0 < A22]
```

```
%% Решим уравнение A(1,1)^2-1 относительно A(1,1)
solve(A(1,1)^2-1, A(1,1))
```

```
ans = 1
```

```
%% Очистим assumptions
assume(A,'clear');
assumptions(A)
```

```
ans =

Empty sym: 1-by-0
```

```
%% Решим уравнение A(1,1)^2-1 относительно A(1,1)
solve(A(1,1)^2-1, A(1,1))
```

```
ans =

-1
 1
```