Основные ошибки в определение функций

In[ ]:=
```
x = 2;
```

In[ ]:=
```
f1[x_]:=x^2 (*верное*)
```

In[ ]:=
```
f2[x] := x^2
```

In[ ]:=
```
f3[x_] = x^2
```

Out[ ]= 4

In[ ]:=
```
f4[x] = x^2
```

Out[ ]= 4

In[ ]:=
```
f1[1]
```

Out[ ]= 1

In[ ]:=
```
f2[1]
```

Out[ ]= f2[1]

In[ ]:=
```
f3[1]
```

Out[ ]= 4

In[ ]:=
```
f4[1]
```

Out[ ]= f4[1]

In[ ]:=
```
f2[2]
```

Out[ ]= 4

Объяснение : f2 - забыта _, т.е. x - не локальная переменная и функция определена только в точке 2 (т.к. x=2);
f3 - забыто :=, т.е. переменная x справа от = не связана с x в аргументе, и функция определена всегда значением 4 $\left(x^2 = 2^2 = 4\right)$;
f4 - комбинация обоих ошибок, функция определена только в точке 2 и значение 4.

Формула Тейлора

In[ ]:=
```
Clear[x]
```
очистить

In[1]:=
```
f[x_] := Cos[x]
```
косинус

Разложение по формуле Тейлора функции `f(x)` в точке `x = 0` до члена с x^3 включительно :

In[7]:=
```
g = Series[f[x], {x, 0, 3}]
```
разложить в ряд

Out[7]= $1 - \frac{x^2}{2} + 0[x]^4$

Отбрасываем O :

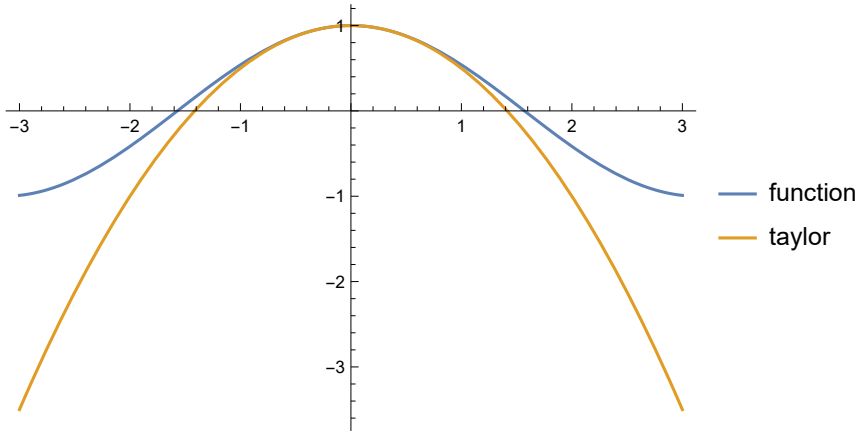
In[•]:=
```
g1 = Normal@g
```
нормальное

Out[•]=
$$1 - \frac{x^2}{2}$$

Визуализация :

In[•]:=
```
Plot[{f[x], g1}, {x, -3, 3}, PlotLegends → {"function", "taylor"}]
```
график функции легенды графика

Out[•]=



Видно, что функция и ее разложение мало отличимо в некоторой довольно большой окрестности нуля.

Определим функцию, вычисляющее разложение до x^n, n – целое :

In[•]:=
```
taylor[n_Integer] := Series[f[x], {x, 0, n}] // Normal
```
целое число разложить в ряд нормаль

In[•]:=
```
taylor[1.]
```

Out[•]=
```
taylor[1.]
```

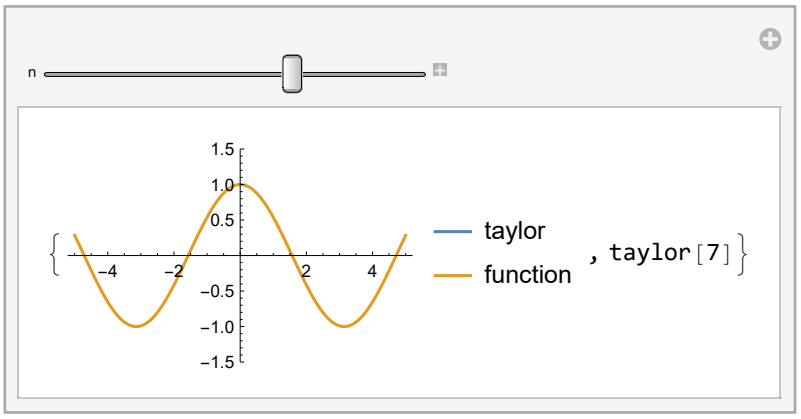
In[•]:=
```
taylor[5]
```

Out[•]=
$$1 - \frac{x^2}{2} + \frac{x^4}{24}$$

Создадим динамический модуль визуализации в зависимости от n : (n меняется от 1 до 10 с шагом 1)

In[ ]:=
```
Manipulate[
 варьировать
  {Plot[Evaluate[{taylor[n], f[x]}], {x, -5, 5}, PlotLegends → {"taylor", "function"},
   гр··· вычислить                                легенды графика
    PlotRange → {-1.5, 1.5}], taylor[n]}, {n, 1, 10, 1}]
    отображаемый диапазон графика
```

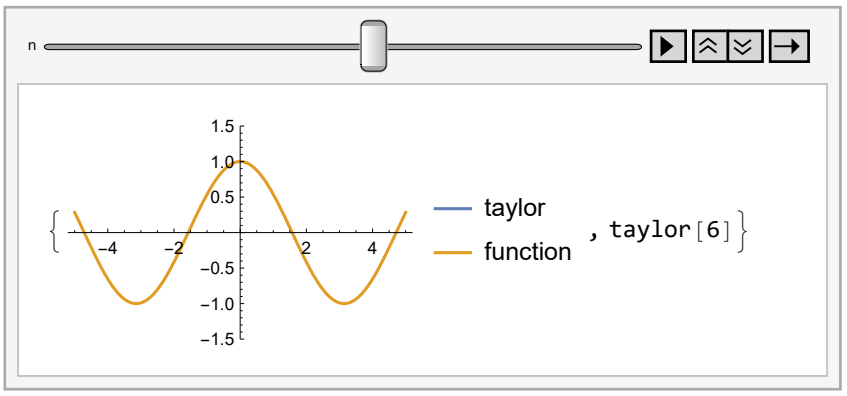
Out[ ]=



Автоматическая анимация :

In[ ]:=
```
Animate[
 анимировать
  {Plot[Evaluate[{taylor[n], f[x]}], {x, -5, 5}, PlotLegends → {"taylor", "function"},
   гр··· вычислить                                легенды графика
    PlotRange → {-1.5, 1.5}], taylor[n]}, {n, 1, 10, 1}]
    отображаемый диапазон графика
```

Out[ ]=

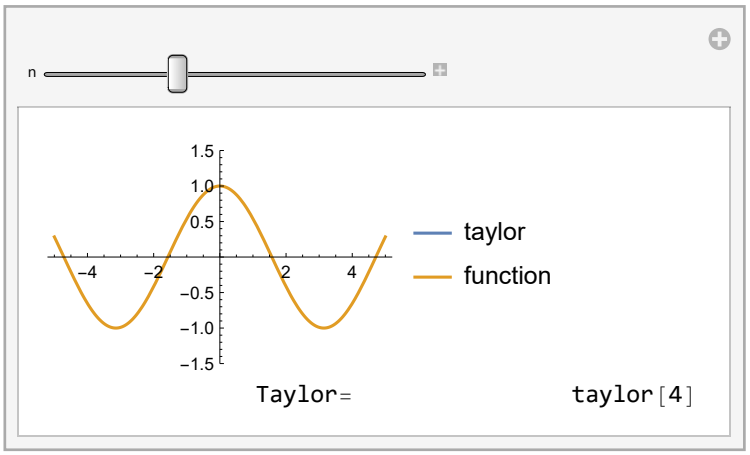


Можно экспортировать в видео/gif через Export[]

Разметка динамического окна через Grid:

In[ ]:=
```
Manipulate[Grid[
 {{Plot[Evaluate[{taylor[n], f[x]}], {x, -5, 5}, PlotLegends → {"taylor", "function"},
   PlotRange → {-1.5, 1.5}]}, {"Taylor=", taylor[n]}}], {n, 1, 10, 1}]
```
варьировать | таблица | гр··· | вычислить | легенды графика | отображаемый диапазон графика

Out[ ]=



Суммы и ряды

Вычисление суммы ряда:

In[ ]:=
```
Sum[1/2^k, {k, 1, ∞}]
```
сумма

Out[ ]=
```
1
```

Бесконечность : Infinity или Esc + i + n + f + Esc

```
:inf

Infinity
```
бесконечность

In[ ]:=
```
Sum[1/2^k, {k, 1, 10}]
```
сумма

Out[ ]=
$$\frac{1023}{1024}$$

Проверка на сходимость ряда :

In[ ]:=
```
SumConvergence[1/2^k, k]
```
сходимость ряда

Out[ ]=
```
True
```

In[ ]:=
```
SumConvergence[1/n, n]
```
сходимость ряда

Out[ ]=
```
False
```

In[•]:= **SumConvergence**$\left[\frac{1}{n^a}, n\right]$
сходимость ряда

Out[•]= Re[a] > 1

В пакете по умолчанию переменные комплексные, но можно упростить полученное выражение в предположении о вещественности переменной a :

In[•]:= **Refine[%, Assumptions → a ∈ Reals]**
уточнить предположения множест

Out[•]= a > 1

Значок принадлежит : Element[a, Reals] ил Esc + e + l + e + m + Esc

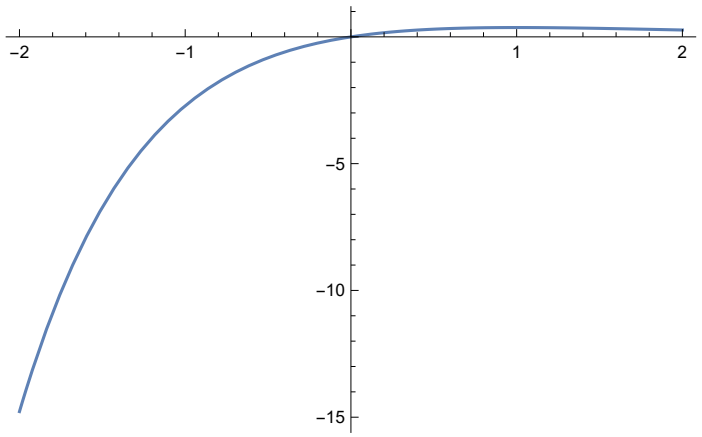
:**elem**

Исследование функций

In[•]:= **f[x_] := x Exp[-x]**
показатели

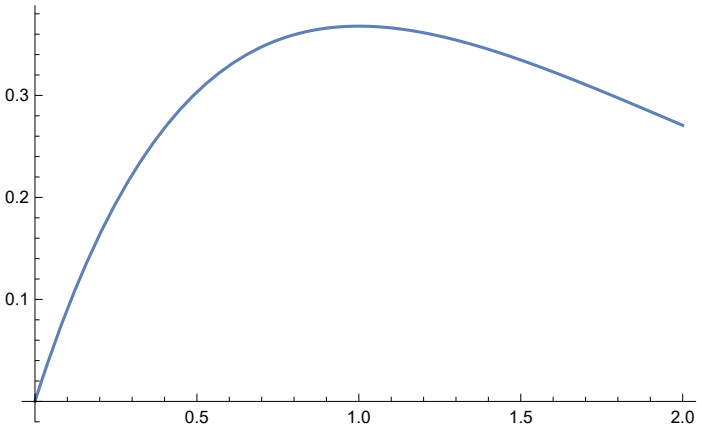
In[•]:= **Plot[f[x], {x, -2, 2}]**
график функции

Out[•]=



In[•]:= **Plot[f[x], {x, 0, 2}]**
график функции

Out[•]=



Локальные экстремумы

Решение уравнения f' = 0

```
In[ ]:= Solve[f'[x] == 0, x]
```
решить уравнения

```
Out[ ]= {{x → 1}}
```

Приближенные методы поиска экстремумов - определить примерное значение по графику и применить функции FindMaximum/FindMinimum для уточнения

Например, по графику считаем что максимум при x = 0.8

```
In[ ]:= FindMaximum[f[x], {x, 0.8}]
```
найти максимум

```
Out[ ]= {0.367879, {x → 1.}}
```

и получаем что максимум в x = 1, а значение функции в этой точке 0.3678 ...

```
In[ ]:= FindMinimum[f[x], {x, 0.8}]
```
найти минимум

General: Overflow occurred in computation.

FindMinimum: The function value Overflow[] is not a real number at {x} = {−6.91145 × 10^15}.

```
Out[ ]= {-1.433740734531272 × 10^272 873 025 596 888, {x → -6.28313 × 10^14}}
```

Минимума нет!

Наибольшее и наименьшее значение функции на отрезке

Найдем наибольшее значение функции f (x) на отрезке [-1, 2] :

```
In[ ]:= Maximize[{f[x], -1 ≤ x ≤ 2}, x]
```
максимизировать

```
Out[ ]= {1/ℯ, {x → 1}}
```

Найдем наименьшее значение функции f (x) на отрезке [-1, 2] :

```
In[ ]:= Minimize[{f[x], -1 ≤ x ≤ 2}, x]
```
минимизировать

```
Out[ ]= {-ℯ, {x → -1}}
```

Ответ вида {значение, {точка в которой достигается}}

Также есть функции, ищущие это приближенно:

```
In[ ]:= NMaximize[{f[x], -1 ≤ x ≤ 2}, x]
```
численная максимизация

```
Out[ ]= {0.367879, {x → 1.}}
```

```
In[ ]:= NMinimize[{f[x], -1 ≤ x ≤ 2}, x]
```
численная минимизация

```
Out[ ]= {-2.71828, {x → -1.}}
```

Можно искать на R:

```
In[ ]:= Maximize[f[x], x]
        максимизировать
```

$$\text{Out[ ]}= \left\{\frac{1}{e}, \{x \to 1\}\right\}$$

```
In[ ]:= Minimize[f[x], x]
        минимизировать

        ••• Minimize: The minimum is not attained at any point satisfying the given constraints.
```

$$\text{Out[ ]}= \{-\infty, \{x \to -\infty\}\}$$

Наименьшего значения нет:

Правила и подстановки

Предыдущие функции выдавали ответы вида x -> a. Это называется правилом.

Например, правило x -> 1 читается так: заменить x на 1

Правила можно применять к выражениям с помощью оператора /.

```
In[ ]:= 2 x + 3 /. x → 1
```

```
Out[ ]= 5
```

Читается так : заменить x на 1 в выражении 2 x + 3

Многие функции выдают правила в качестве результата:

```
In[ ]:= s = Solve[2 x + 3 == 0, x]
         решить уравнения
```

$$\text{Out[ ]}= \left\{\left\{x \to -\frac{3}{2}\right\}\right\}$$

Здесь список из списка, для получения элемента нужно использовать оператор Part[] или краткая форма [[ ]]

```
s[[1]]
```

Индексация начинается с 1!

Результат Solve можно применить к любому выражению

```
In[ ]:= x^2 + x /. s[[1]]
```

$$\text{Out[ ]}= \frac{3}{4}$$

Также можно применять списки правил

```
In[ ]:= x + y /. {x → 1, y → -1}
```

```
Out[ ]= 0
```

и применять последовательно несколько правил

```
In[ ]:= x + y /. x → 1 /. y → -1
```

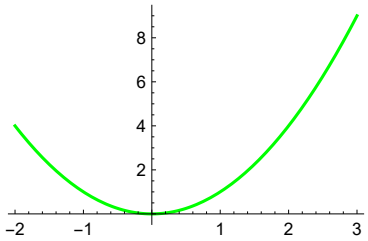
```
Out[ ]= 0
```

Визуализация функций

Визуализация явно заданной функции : (f(x) = x^2)

In[ ]:=
```
h1 = Plot[x^2, {x, -2, 3}, PlotStyle → Green]
```
график функции стиль графика зелёный

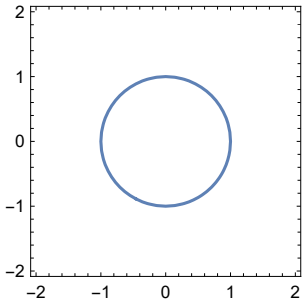
Out[ ]=



Визуализация неявно заданной функции ($x^2 + y^2 = 1$)

In[ ]:=
```
h2 = ContourPlot[x^2 + y^2 == 1, {x, -2, 2}, {y, -2, 2}]
```
контурный график

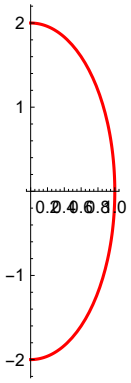
Out[ ]=



Визуализация параметрически заданной функции ((x(t) = sin(t), y(t) = 2 cos(t))

In[ ]:=
```
h3 = ParametricPlot[{Sin[t], 2 Cos[t]}, {t, 0, π}, PlotStyle → Red]
```
график параметри··· синус косинус стиль графика красн

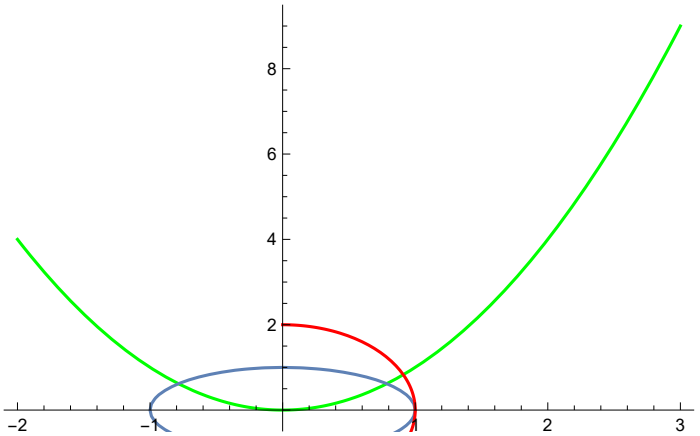
Out[ ]=



Можно объединять на одной координатной плоскости :

In[ ]:=
```
Show[h1, h2, h3]
```
показать

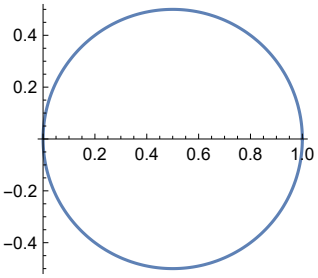
Out[ ]=



Визуализация функции заданной в полярной системе координат (r(ϕ) = cos (ϕ))

In[ ]:=
```
PolarPlot[Cos[ϕ], {ϕ, 0, π}]
```
полярная ⋯ косинус

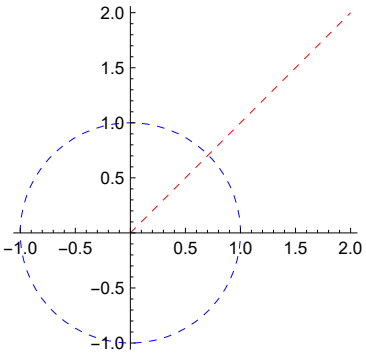
Out[ ]=



Двумерная графика - набор графических примитивов

In[ ]:=
```
hp = Graphics[{Dashed, Red, Line[{{0, 0}, {2, 2}}], Blue, Circle[{0, 0}, 1]}, Axes → True]
```
графика штрих⋯ кр⋯ (ломаная) линия синий круг оси истина

Out[ ]=

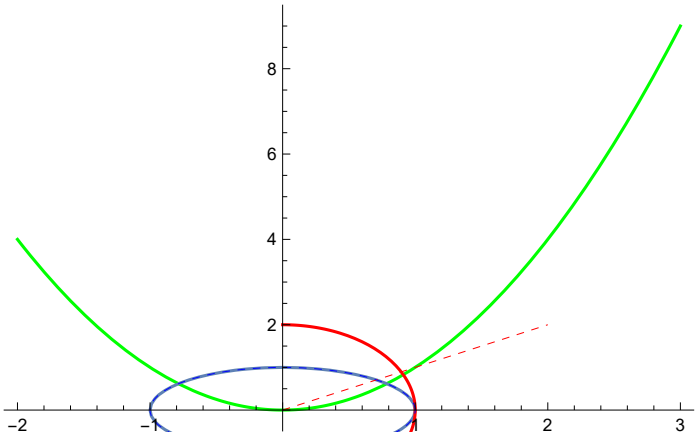


Команды в списке исполняются последовательно, и стили действуют на все что дальше в списке!

Можно объединять с графиками функций:

In[ ]:=
```
Show[h1, h2, h3, hp]
```
показать

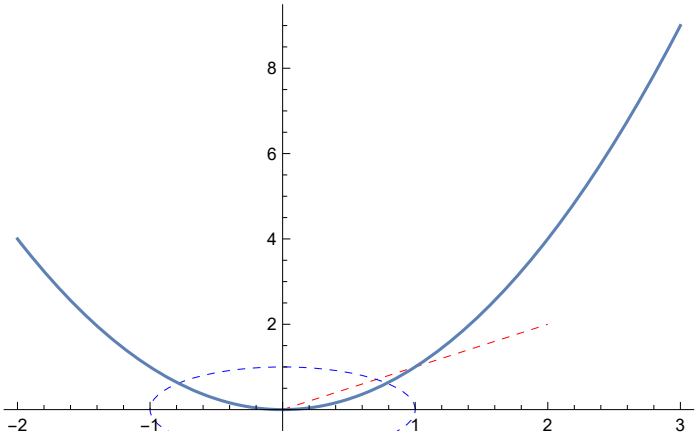
Out[ ]=



Также можно рисовать графические примитивы сразу на Plot - с помощью опций Prolog и Epilog : первая под графиком, вторая - поверх графика:

In[ ]:=
```
Plot[x^2, {x, -2, 3},
 Prolog → {Dashed, Red, Line[{{0, 0}, {2, 2}}], Blue, Circle[{0, 0}, 1]}]
```
график функции
пролог штрих··· кр··· (ломаная) линия синий круг

Out[ ]=



In[ ]:=
```
Plot[x^2, {x, -2, 3},
 Epilog → {Dashed, Red, Line[{{0, 0}, {2, 2}}], Blue, Circle[{0, 0}, 1]}]
```
график функции
эпилог штрих··· кр··· (ломаная) линия синий круг

Out[ ]=