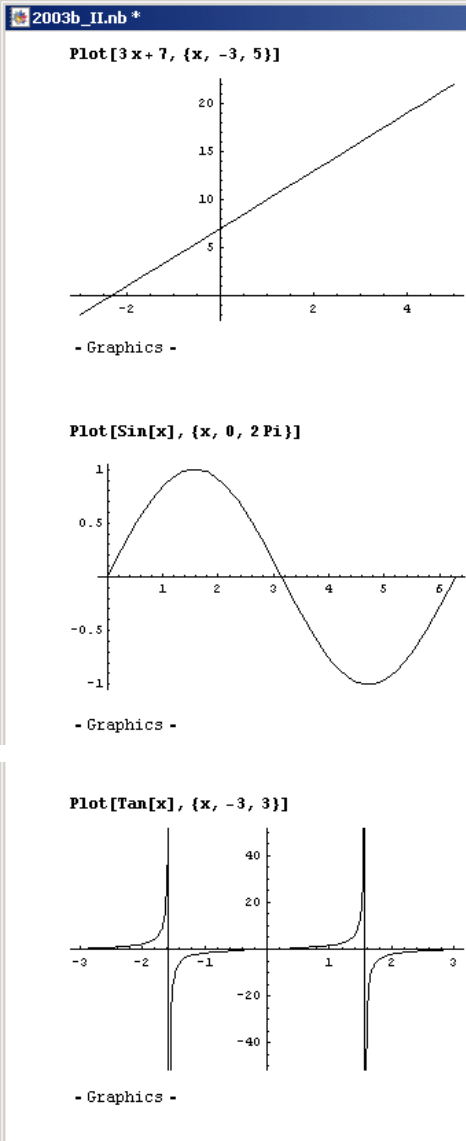
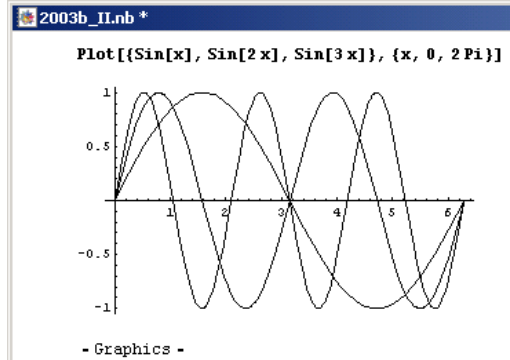


Γραφικές Παραστάσεις με το μαθηματικό πακέτο **Mathematica**

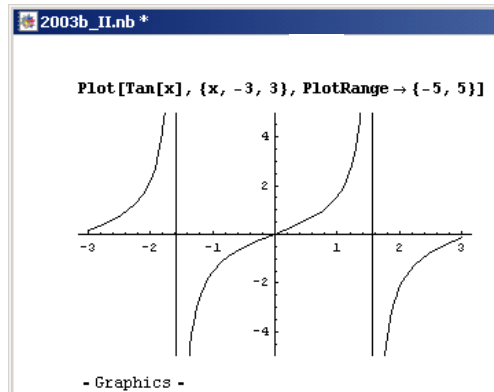
Απλές Γραφικές Παραστάσεις



Πολλαπλές Γραφικές Παραστάσεις



Όρια Γραφήματος

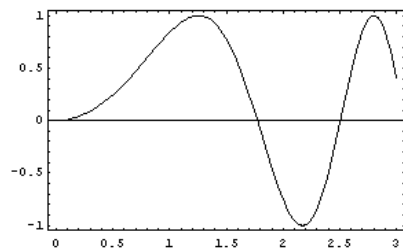


ΔΙΑΜΟΡΦΩΣΗ ΓΡΑΦΙΚΩΝ ΠΑΡΑΣΤΑΣΕΩΝ

Πλαίσια

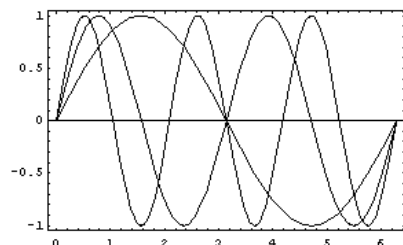
2003b_II.nb *

```
Plot[Sin[x^2], {x, 0, 3}, Frame -> True]
```



- Graphics -

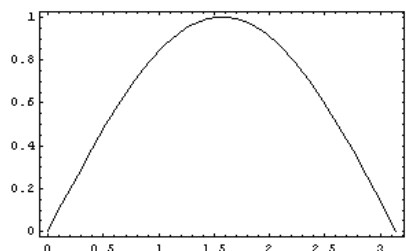
```
Plot[{Sin[x], Sin[2 x], Sin[3 x]}, {x, 0, 2 Pi}, Frame -> True]
```



- Graphics -

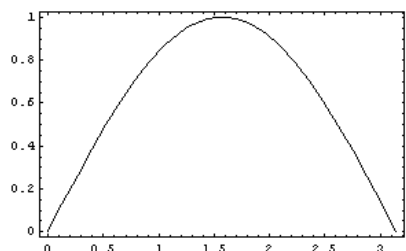
2003b_II.nb *

```
Plot[Sin[x], {x, 0, Pi}, Frame -> True]
```



- Graphics -

```
Show[%]
```

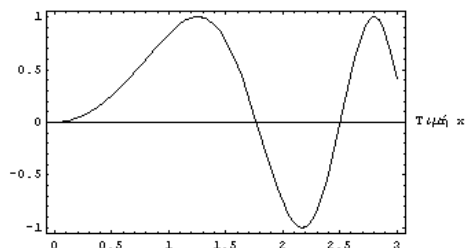


- Graphics -

Άξονες

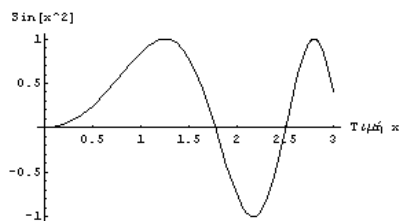
2003b_II.nb *

```
Plot[Sin[x^2], {x, 0, 3}, AxesLabel -> {"Τιμή x", "Sin[x^2]"}, Frame -> True]
```



- Graphics -

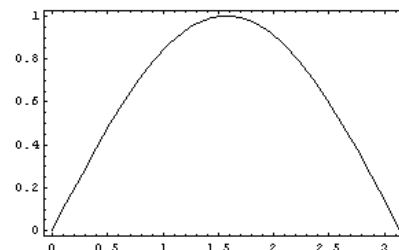
```
Plot[Sin[x^2], {x, 0, 3}, AxesLabel -> {"Τιμή x", "Sin[x^2]"}]
```



- Graphics -

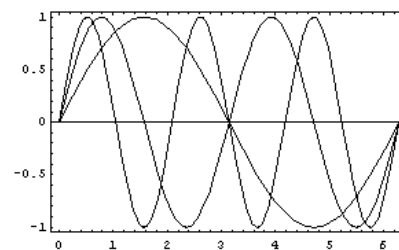
2003b_II.nb *

```
Show[Out[11], AxesLabel -> {"Τιμή x"}]
```



- Graphics -

```
Show[Out[6], Frame -> True]
```



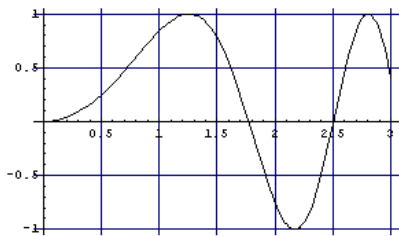
- Graphics -

ΔΙΑΜΟΡΦΩΣΗ ΓΡΑΦΙΚΩΝ ΠΑΡΑΣΤΑΣΕΩΝ

Διαμόρφωση Γραφικών

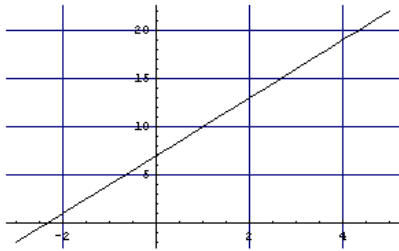
2003b_II.nb *

```
Plot[Sin[x^2], {x, 0, 3}, GridLines -> Automatic]
```



- Graphics -

```
plot1 = Plot[3 x + 7, {x, -3, 5}, GridLines -> Automatic]
```

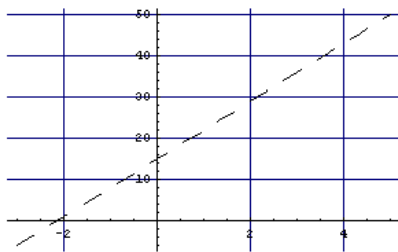


- Graphics -

2003b_II.nb *

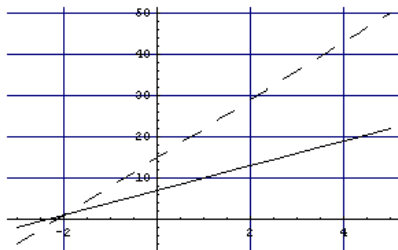
```
g[x_] := 7 x + 15
```

```
plot2 = Plot[g[x], {x, -3, 5}, PlotStyle -> Dashing[{0.05}], GridLines -> Automatic]
```



- Graphics -

```
Show[plot1, plot2]
```

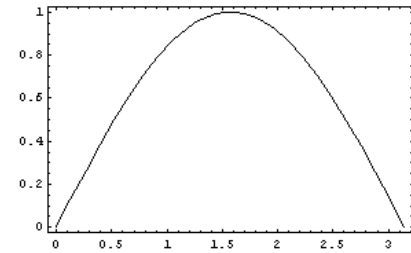


- Graphics -

Διαχείριση προηγούμενων γραφημάτων

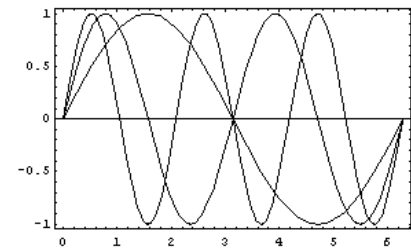
2003b_II.nb *

```
Show[Out[11], AxesLabel -> {"Τιμή x"}]
```



- Graphics -

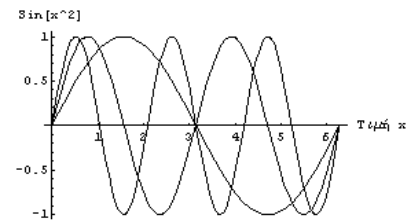
```
Show[Out[6], Frame -> True]
```



- Graphics -

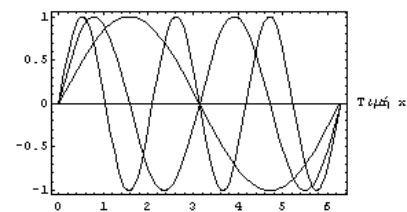
2003b_II.nb *

```
Show[Out[6], AxesLabel -> {"Τιμή x", "Sin[x^2]"}]
```



- Graphics -

```
Show[Out[6], AxesLabel -> {"Τιμή x", "Sin[x^2]"}, Frame -> True]
```



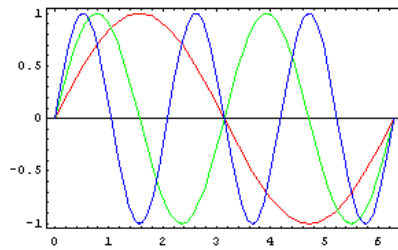
- Graphics -

ΔΙΑΜΟΡΦΩΣΗ ΓΡΑΦΙΚΩΝ ΠΑΡΑΣΤΑΣΕΩΝ

Διαμόρφωση Χρωμάτων

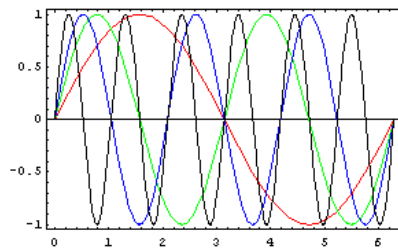
2003b_Π.nb *

```
Plot[{Sin[x], Sin[2 x], Sin[3 x]}, {x, 0, 2 Pi}, PlotStyle -> {{RGBColor[1, 0, 0]}, {RGBColor[0, 1, 0]}, {RGBColor[0, 0, 1]}},  
Frame -> True]
```



- Graphics -

```
Plot[{Sin[x], Sin[2 x], Sin[3 x], Sin[6 x]}, {x, 0, 2 Pi},  
PlotStyle -> {{RGBColor[1, 0, 0]}, {RGBColor[0, 1, 0]}, {RGBColor[0, 0, 1]}, {RGBColor[0, 0, 0]}}, Frame -> True]
```

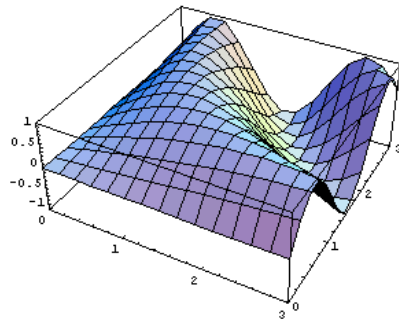


- Graphics -

Γραφικές Παραστάσεις 3 διαστάσεων

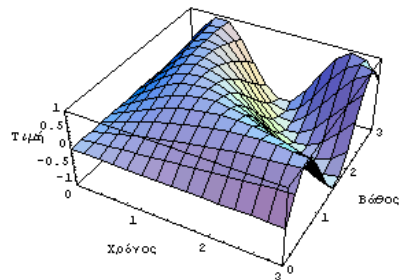
2003b_IL.nb *

```
Plot3D[Sin[x y], {x, 0, 3}, {y, 0, 3}]
```



- SurfaceGraphics -

```
Show[%, AxesLabel -> {"Χρόνος", "Βάθος", "Τιμή"}]
```

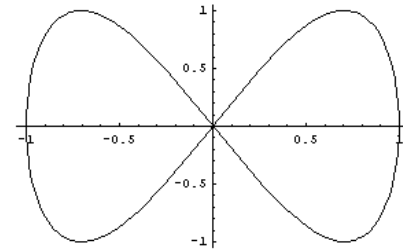


- SurfaceGraphics -

Παμετρικές Γραφικές Παραστάσεις

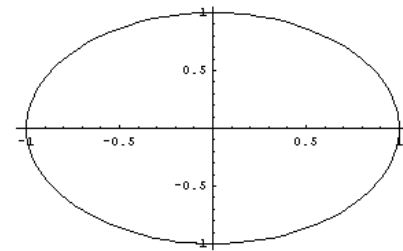
2003b_IL.nb *

```
ParametricPlot[{Sin[t], Sin[2 t]}, {t, 0, 2 Pi}]
```



- Graphics -

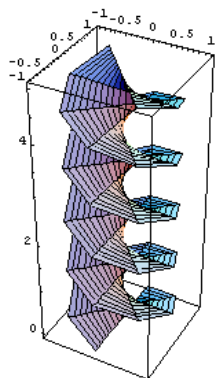
```
ParametricPlot[{Sin[t], Cos[t]}, {t, 0, 2 Pi}]
```



- Graphics -

2003b_IL.nb *

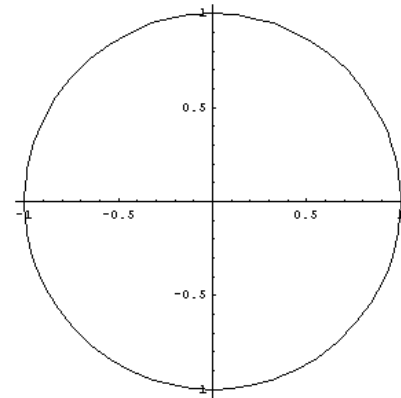
```
ParametricPlot3D[{u Sin[t], u Cos[t], t/3}, {t, 0, 15}, {u, -1, 1}]
```



- Graphics3D -

2003b_IL.nb *

```
ParametricPlot[{Sin[t], Cos[t]}, {t, 0, 2 Pi}, AspectRatio -> Automatic]
```



- Graphics -

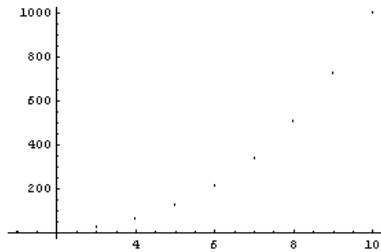
ΓΡΑΦΙΚΕΣ ΠΑΡΑΣΤΑΣΕΙΣ ΑΡΙΘΜΗΤΙΚΩΝ ΔΕΔΟΜΕΝΩΝ

2003b_II.nb *

```
t = Table[i^3, {i, 10}]
```

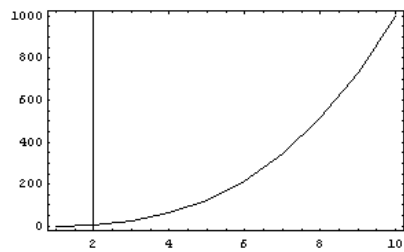
```
{1, 8, 27, 64, 125, 216, 343, 512, 729, 1000}
```

```
ListPlot[t]
```



- Graphics -

```
ListPlot[t, PlotJoined → True, Frame → True]
```



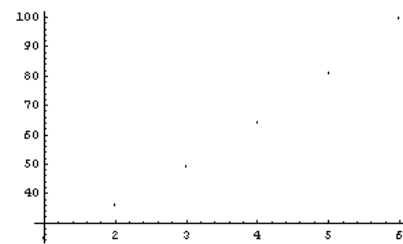
- Graphics -

2003b_II.nb *

```
t = Table[i^2, {i, 5, 10}]
```

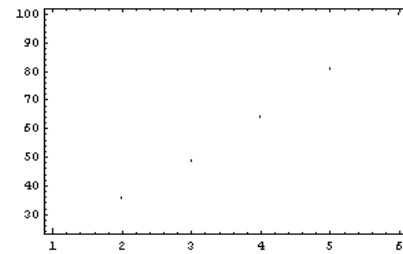
```
{25, 36, 49, 64, 81, 100}
```

```
ListPlot[t]
```



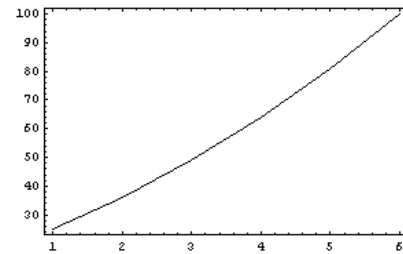
- Graphics -

```
ListPlot[t, Frame → True]
```



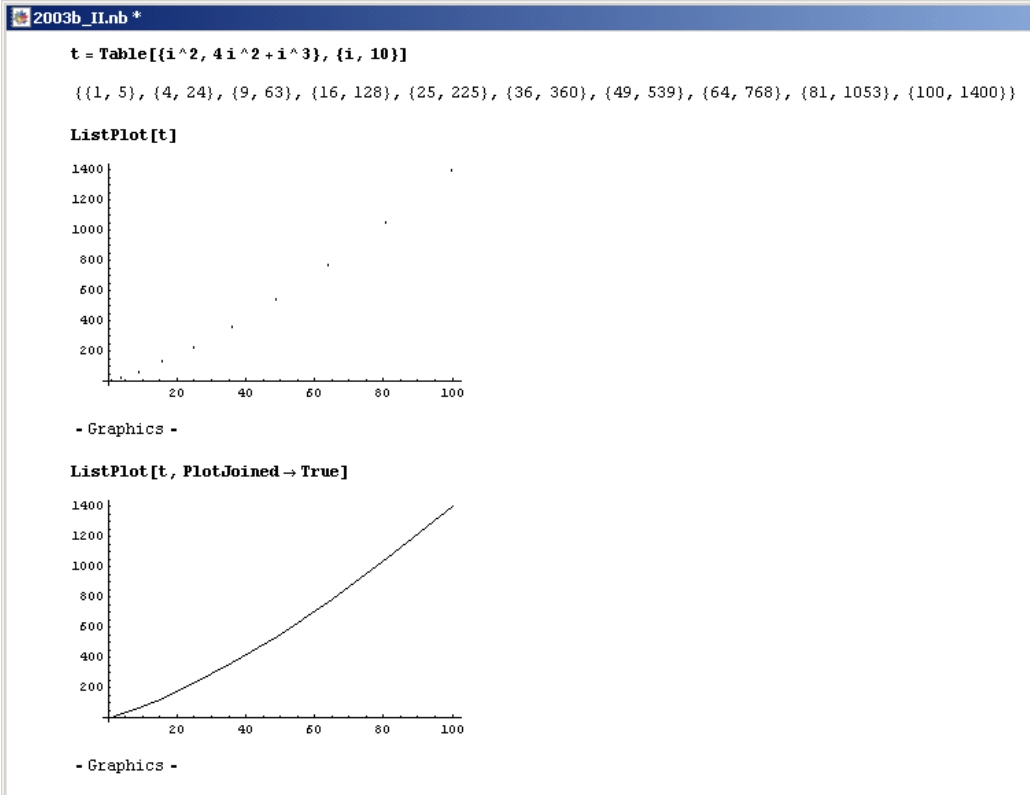
- Graphics -

```
ListPlot[t, Frame → True, PlotJoined → True]
```

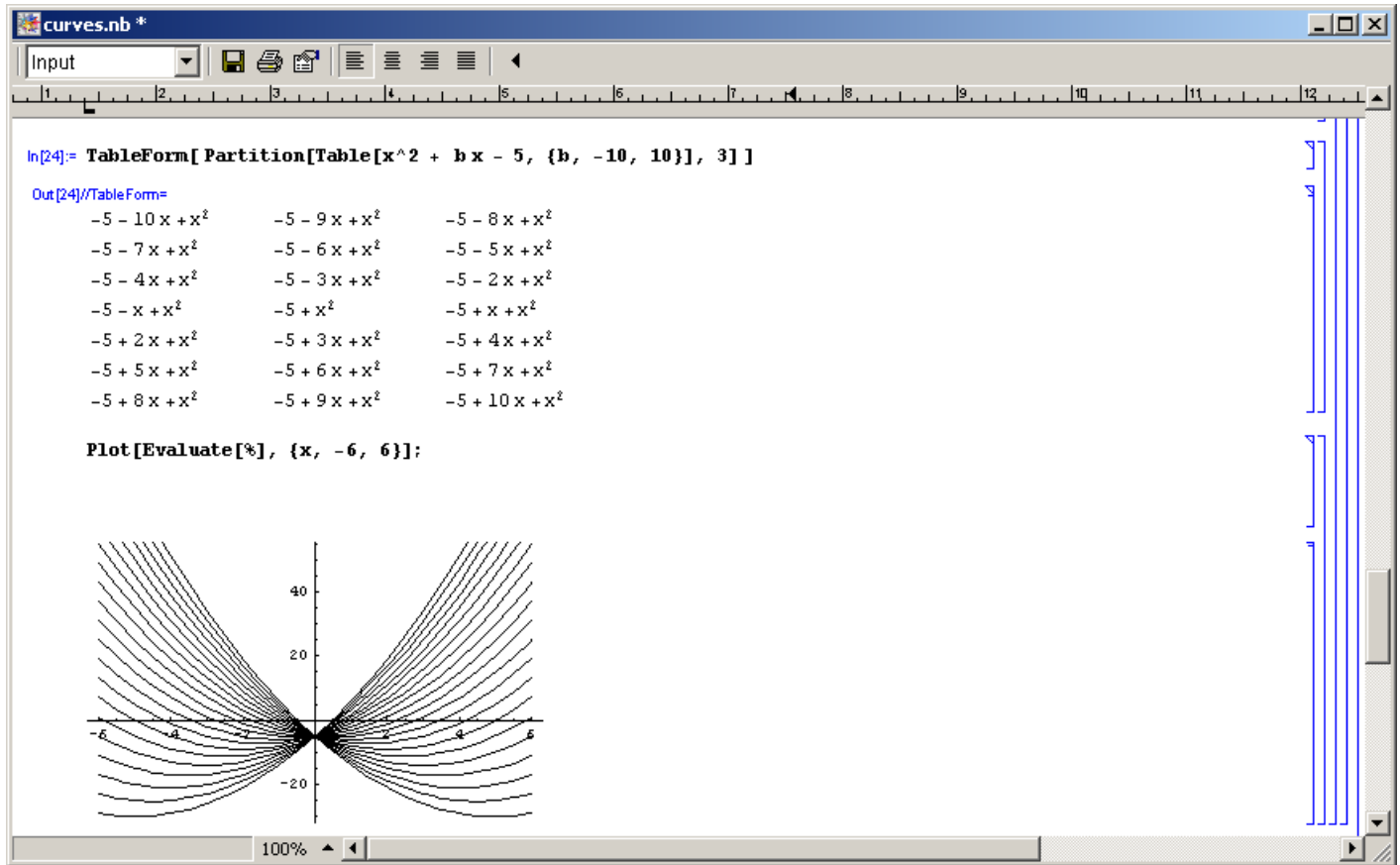


- Graphics -

ΓΡΑΦΙΚΕΣ ΠΑΡΑΣΤΑΣΕΙΣ ΑΡΙΘΜΗΤΙΚΩΝ ΔΕΔΟΜΕΝΩΝ



ΟΙΚΟΓΕΝΕΙΕΣ ΚΑΜΠΥΛΩΝ



Διαγράμματα κλίσης-εμβαδού σωστή τοποθέτηση στους άξονες

Help Browser

Go To: TableForm Back Hide Categories

Built-in Functions Add-ons The Mathematica Book
Getting Started/Demos Other Information Master Index

Algebraic Computation Mathematical Functions Lists and Matrices Graphics and Sound Programming Input and Output

Basic Operations File Input File Output Import and Export Format Types Box Types

MathMLForm DisplayForm FullForm TreeForm MatrixForm TableForm

TableForm

- TableForm[list] prints with the elements of list arranged in an array of rectangular cells.
- The height of each row and the width of each column are determined by the maximum size of an element in the row or column.
- TableForm prints a single-level list in a column. It prints a two-level list as a two-dimensional table. More deeply nested lists are by default printed with successive dimensions alternating between rows and columns.
- Arrays in which all sublists at a particular level are not of the same length display as ragged tables.
- The following options can be given:

TableAlignments	Automatic	how to align entries in each dimension
TableDepth	Infinity	maximum number of levels to include
TableDirections	Automatic	whether to arrange dimensions as rows or columns
TableHeadings	None	how to label table entries
TableSpacing	Automatic	how many spaces to put between entries in each dimension
- TableForm acts as a "wrapper", which affects printing, but not evaluation.
- See *The Mathematica Book*: [Section 1.8.2](#) and [Section 2.8.8](#).
- See also: [ColumnForm](#), [MatrixForm](#), [GridBox](#), [GraphicsArray](#).

Further Examples

TableForm displays a list of lists as a table.

```
In[1]:= TableForm[{{a, b}, {c, d}}]
```

```
Out[1]/TableForm=
```

a	b
c	d

This centers the column.

```
In[2]:= TableForm[{a, bbbb, ccccc}, TableAlignments -> Center]
```

```
Out[2]/TableForm=
```

a	bbbb	ccccc
---	------	-------

This displays to a depth of 1.

```
In[3]:= TableForm[{a, {bbb, b}, cccc}, TableDepth -> 1]
```

```
Out[3]/TableForm=
```

a,	
----	--

Help Browser

Go To Evaluate Back Hide Categories

Built-in Functions

Add-ons

The Mathematica Book

Getting Started/Demos

Other Information

Master Index

Algebraic Computation

Mathematical Functions

Lists and Matrices

Graphics and Sound

Programming

Input and Output

Functional Programming

Pattern Matching

Rule Application

String Manipulation

Scoping

Evaluation Control

Evaluate

Unevaluated

HoldPattern

Hold

HoldComplete

HoldForm

Evaluate

■ Evaluate[expr] causes expr to be evaluated even if it appears as the argument of a function whose attributes specify that it should be held unevaluated.

■ Example: Hold[Evaluate[1 + 1]]→Hold[2].

■ You can use Evaluate to override HoldFirst, etc. attributes of built-in functions.

■ Evaluate only overrides HoldFirst, etc. attributes when it appears directly as the head of the function argument that would otherwise be held.

■ See The Mathematica Book: [Section 1.9.1](#) and [Section 2.5.5](#).

■ See also: [ReleaseHold](#)

Further Examples

Evaluate causes the list of functions to be constructed immediately, so that Plot will know that it needs to plot more than one curve.

In[1]:= Plot[Evaluate[Table[ChebyshevT[n, x], {n, 1, 8}]], {x, -1, 1}];

Evaluate is needed because Plot has the HoldAll attribute.

In[2]:= Attributes[Plot]

Out[2]:= {HoldAll, Protected}

If Evaluate is omitted, this example will generate an error, since Plot expects the first argument to evaluate to a number rather than to a list of numbers.

In[3]:= Plot[Table[ChebyshevT[n, x], {n, 1, 4}], {x, -1, 1}]

Evaluate is not necessary here, since Factor does not hold its arguments unevaluated.

In[4]:= Factor[Evaluate[x² - 1]]

Out[4]:= (-1 + x) (1 + x)

Evaluate has no effect unless the enclosing expression is evaluated.

In[5]:= If[test, (Evaluate[2 + 2]), Evaluate[2 + 2]]

Out[5]:= If[test, (Evaluate[2 + 2]), 4]

This shows one way to put an evaluated expression into a position that would otherwise be held unevaluated.

In[6]:= ReplacePart[If[test, {Evaluate[2 + 2]}, 4], 2 + 2, 2]

Out[6]:= If[test, 4, 4]

Go To

Partition

Back

Hide Categories

Built-in Functions

Add-ons

The Mathematica Book

Getting Started/Demos

Other Information

Master Index

Algebraic Computation

Mathematical Functions

Lists and Matrices

Graphics and Sound

Programming

Input and Output

...

List Construction

Element Extraction

List Testing

List Operations

Structure Manipulation

Vector Operations

Partition

Flatten

Split

FlattenAt

Thread

Sequence

...

Partition

- `Partition[list, n]` partitions `list` into non-overlapping sublists of length `n`.
- `Partition[list, n, d]` generates sublists with offset `d`.
- `Partition[list, {n1, n2, ...}]` partitions a nested list into blocks of size $n_1 \times n_2 \times \dots$.
- `Partition[list, {n1, n2, ...}, {d1, d2, ...}]` uses offset `di` at level `i` in `list`.
- `Partition[list, n, d, {kL, kR}]` specifies that the first element of `list` should appear at position `kL` in the first sublist, and the last element of `list` should appear at or after position `kR` in the last sublist. If additional elements are needed, `Partition` fills them in by treating `list` as cyclic.
- `Partition[list, n, d, {kL, kR}, x]` pads if necessary by repeating the element `x`.
- `Partition[list, n, d, {kL, kR}, {x1, x2, ...}]` pads if necessary by cyclically repeating the elements `xi`.
- `Partition[list, n, d, {kL, kR}, {}]` uses no padding, and so can yield sublists of different lengths.
- `Partition[list, nlist, dlist, {klistL, klistR}, padlist]` specifies alignments and padding in a nested list.

■ Example: `Partition[{a,b,c,d,e,f}, 2] → {{a, b}, {c, d}, {e, f}}`.

■ All the sublists generated by `Partition[list, n, d]` are of length `n`. Some elements at the end of `list` may therefore not appear in any sublist.

■ The element `e` in `Partition[{a,b,c,d,e}, 2] → {{a, b}, {c, d}}` is dropped.

■ `Partition[{a,b,c,d,e}, 3, 1] → {{a, b, c}, {b, c, d}, {c, d, e}}` generates sublists with offset 1.

■ All elements of `list` appear in the sublists generated by `Partition[list, n, 1]`.

■ If `d` is greater than `n` in `Partition[list, n, d]`, then elements in the middle of `list` are skipped.

■ `Partition[list, 1, d]` picks out elements in the same way as `Take[list, {1, -1, d}]`.

■ `Partition[list, n, d, {kL, kR}]` effectively allows sublists that have overhangs that extend past the beginning or end of `list`.

■ `Partition[list, n, d, k]` is equivalent to `Partition[list, n, d, {k, k}]`.

■ Common settings for `{kL, kR}` are:

<code>{1, -1}</code>	allow no overhangs
<code>{1, 1}</code>	allow maximal overhang at the end
<code>{-1, -1}</code>	allow maximal overhang at the beginning
<code>{-1, 1}</code>	allow maximal overhangs at both beginning and end

■ Example: `Partition[{a,b,c,d}, 2, 1, {-1, 1}] → {{d, a}, {a, b}, {b, c}, {c, d}, {d, a}}`.

■ `Partition[list, n, d, {kL, kR}, padlist]` effectively lays down repeated copies of `padlist`, then superimposes one copy of `list` on them, and partitions the result.

■ Common settings for `padlist` are:

<code>x</code>	pad with repetitions of a single element
<code>{x₁, x₂, ...}</code>	pad with cyclic repetitions of a sequence of elements
<code>list</code>	pad by treating <code>list</code> as cyclic (default)