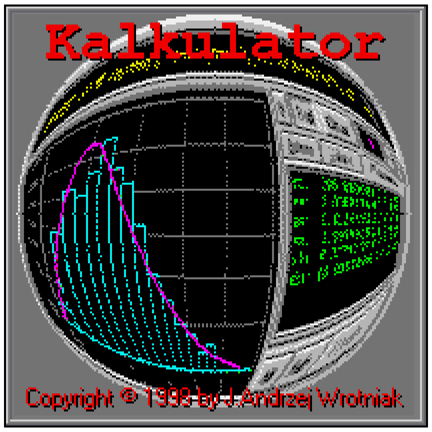
Welcome to Kalkulator



Welcome to Kalkulator! I hope you will like the program and find it useful. Please let me know what you like and dislike about it, so that I can keep improving it.

Kalkulator is shareware: have a look at the Registration and Support page. Then start from What Does Kalkulator Do and Evaluating Expressions.

In most situations pressing the F1 key will bring this help. Use it; it should answer most of your questions. You may also use the arrow buttons above to browse through the help pages. I have spent quite a lot of time writing this help; show your appreciation and read it!

> If you are not a heavy scientific and engineering math user, do not feel intimidated by the more advanced capabilities of the program. The features you do not use will not get into your way; just forget about them and use only those you need. After one hour with Kalkulator you will feel at home.
>
> (If, after that, you still feel Kalkulator is more than you need, try out its offspring: the Midget – it is smaller and simpler, yet quite smart; the chances are you may like it.)

If you like reading "cover to cover", use the browsing buttons, **[<<]** and **[>>]** at the top of this Help window to get a complete walk-through; this may save you some time and frustration later on. Otherwise, start from one of the following pages:

- **Quick start: an overview of the Main Window**
- **Table of Contents**
- **New in this version**
- **Credits and copyrights**

Kalkulator

About the name: this is not a "Kountry Kitchen" trying-to-be-cute stuff. Kalkulator (pronounced: "Cal-coo-lah-tor") is a legitimate Polish word; it can be translated into modern American English as "The Mother of All Calculators and There Ain't No Better One, Period, No Kidding".

Table of Contents

What does Kalkulator do?

At its simplest level, Kalkulator is a calculator. It will accept an arithmetic expression, calculate its value, display the result and store it for a later use. For example, entering

```
    5+3*log 1000
```

and pressing Enter will compute and display the expression value, 14, storing it as t (the Temporary Variable, used unless you specify another one).

Instead of pressing Enter, you may use the "**=**" key (or the **[=]** button) and choose a variable from a pop-up menu; this variable will be used to store the result.

Variables can be also used as components of expressions, as in

```
    2.5*ln(3+exp X)-sin(2A+3t)
```

Being able to see the whole expression before and after evaluation, you can modify and recompute it at any time. This already is enough to justify using Kalkulator instead of any alternative.

An expression can be more than just evaluated. Treated as a function of t, it can be used to compute integrals, derivatives, sums or roots. It can be also plotted on the screen. A number of expressions can be treated as a system of non-linear equations and solved.

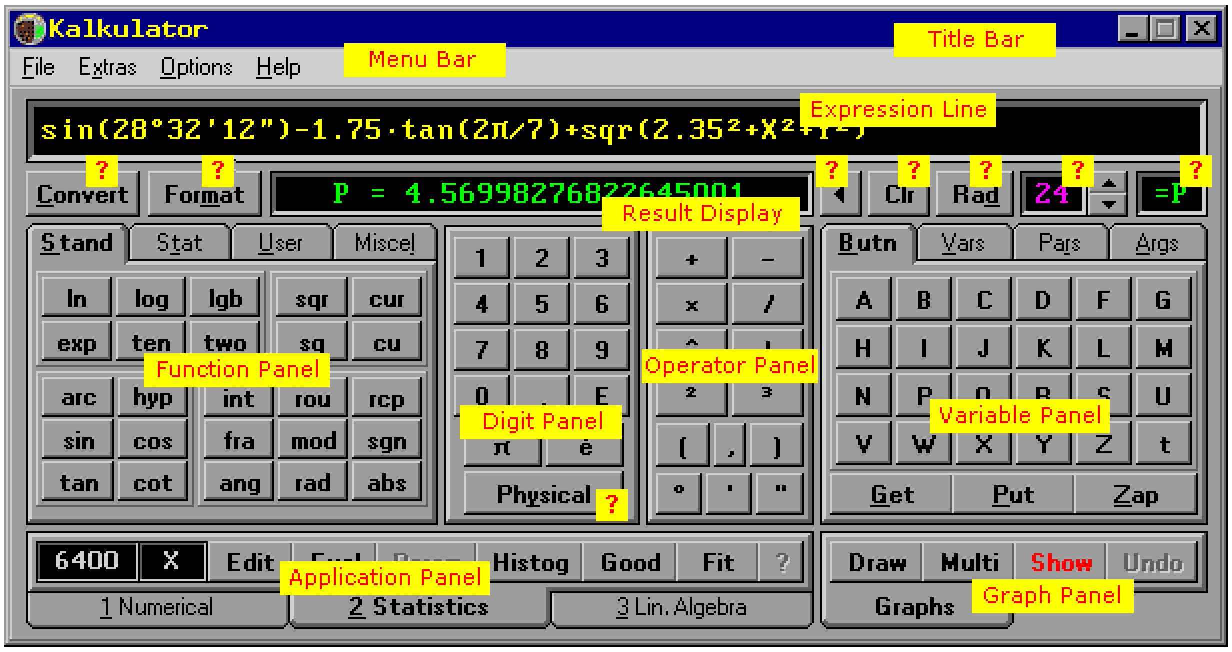
The calculation result can be converted between various measurement units, or displayed in a selected format (for example, as a fraction or as degrees, minutes and seconds).

Kalkulator also has a data buffer to store a number of data points; it can perform simple statistical operations on those points: computing population parameters, drawing histograms or scatter plots (with arbitrary multi-parameter curve or distribution fitting), etc. It will also recompute, using a given formula, all X or Y values in the buffer.

A separate window is provided to perform computer arithmetic operations on integer arguments of finite bit length and to translate numbers between binary, octal, hexadecimal and decimal (signed or not) representations.

Last but not least, Kalkulator can be also used for linear algebra: vector and matrix operations. This includes solving systems of simultaneous linear equations.

The Main Window



Click on the yellow stickers for more information about Main Window components.

See also: Customizing Kalkulator

This field shows the Result Variable, into which the expression value will be stored when you hit the Enter key. (Initially it is set to the Temporary Variable, **t**.)

Clicking here brings up a pop-up menu from which another variable can be chosen. Then the current expression value will be computed, and the result stored in the selected variable.

The number of the current expression line is shown here, and the arrow buttons allow to move to the next or the previous one. Clicking on the number will allow you to jump directly to a line selected from a pop-up menu.

This button toggles between the Radian and Degree angular modes.

This is a mouse equivalent of the Backspace
key (for keyboard-phobes only).

[Clr] clears the Expression Line (the current expression only). Pressing the Escape key does the same.

Click on **[Physical]** to access predefined physical constants. The value of the selected constant will be shown in the Result Display.

Click on **[Convert]** to recompute the value shown in the Result Display from one unit of measurement to another. Note: if the Result Display is empty, this button will be disabled!

[Format] is used to change the format of the Result Display (fixed-point, exponential, deg/min/sec etc.). The displayed value itself does not change. Note: if the Result Display is empty, the button will be disabled!

Title Bar

This is a regular Windows title bar. It can be used to move the Main Window around.

To save screen space, you can hide the Title Bar (and Menu Bar) by unchecking the "Menu & Title Bar" option in the Preferences dialog.

When the Title Bar is hidden, you may still drag the Main Window (and other windows) around the screen by pressing and holding the mouse at the window edge: the cursor changes into a grabbing hand to indicate that you are there.

To minimize Kalkulator when the Title Bar is hidden, click in the top left corner of the Main Window (the cursor will change into a pointing finger when you find the place).

Menu Bar

This is a regular Windows menu bar. Some less frequently used operations are accessible from here, see Menu Operations.

You may hide the Menu Bar (and the Title Bar) by unchecking the "Menu & Title Bar" option in the Preferences dialog.
When the Menu Bar is hidden, its functions are accessible from the Main Pop-up Menu .

Main Pop-up Menu

This menu pops up when you press the right mouse button over the Main Window edge or press F10. It provides all functionality of the Kalkulator's Menu Bar, at the same time saving screen space.

Expression Line

This is where the expression you are entering is being shown. The maximum length is 64 characters (128 in the registered version). The arrows at the right move between different expressions, and the current expression number is shown next.

Result Display

Here the result of the last operation is shown. This result can be reformatted to a another representation or precision with use of the **[Format]** button, or converted between different measurement units with use of **[Convert]**.

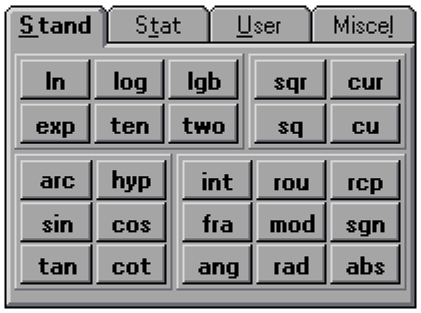
By clicking here (or pressing Shift-Enter) you will be able to send the result to any variable, chosen from a pop-up menu.

Function Panel

This panel contains buttons used to enter function names into expressions. It has four pages, selectable with the tabs at the panel top:

- Standard Function Page
- Statistical Function Page
- User Function Page
- Extra Function Page

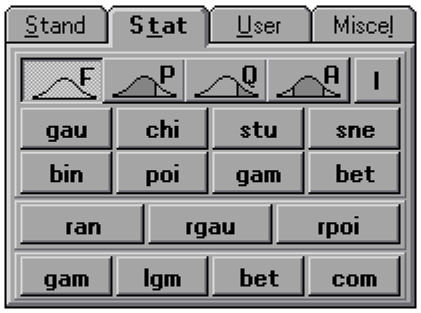
Standard Function Page



See also:

Standard functions

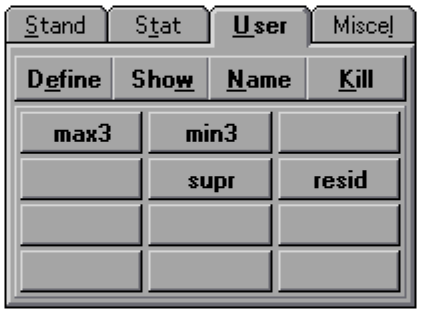
Statistical Function Page



See also:

Statistical distribution functions
Random number generators
Euler and related functions

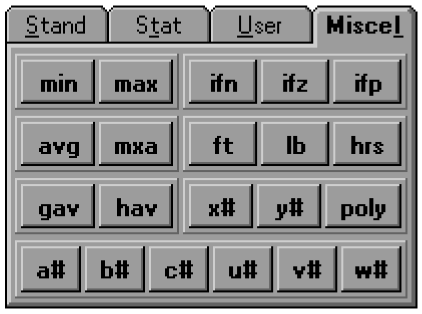
User Function Page



See also:

User functions

Extra Function Page



See also:

- Conditional functions
- Feet, pounds, hours
- Fetch functions
- Standard functions

Operator Panel

This panel contains the buttons for common arithmetic operators. It also includes the parentheses, the comma (used to separate multiple function arguments), and the buttons for entering symbols for degrees, minutes and seconds of angle.

Digit Panel

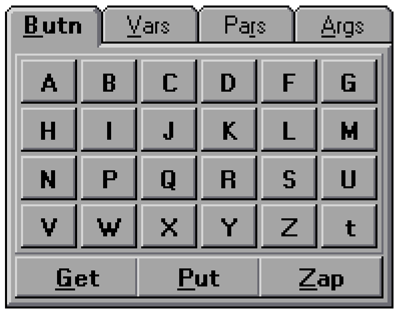
This panel, in addition to digit buttons, contains the decimal point and exponent, the constants: pi and Euler's e, and the button for accessing physical constants.

Variable Panel

This panel allows to access Kalkulator variables, statistical parameters of the data in the buffer, or multiple arguments, used in some numerical operations.
It has four pages:

- Variable Button Page
- Variable Browser Page
- Parameter Page
- Argument Page

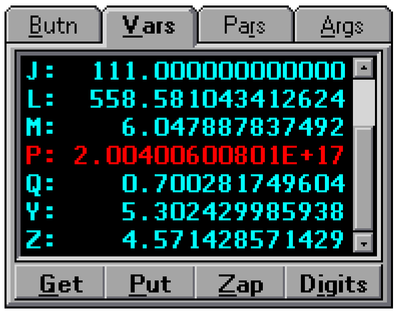
Variable Button Page



See also:

Variables
Accessing variables

Variable Browser Page

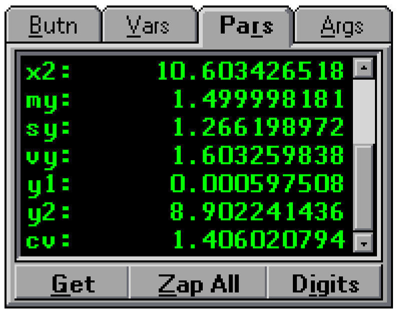


```
J:   111.000000000000
L:   558.581043412624
M:     6.047887837492
P: 2.00400600801E+17
Q:     0.700281749604
Y:     5.302429985938
Z:     4.571428571429
```

See also:

- Variables
- Accessing variables

Parameter Page

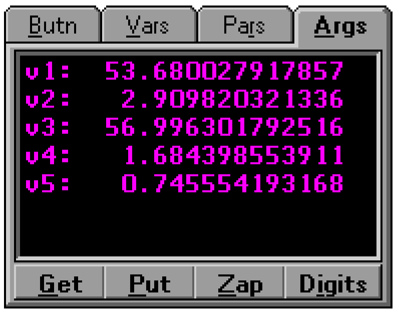


```
x2:     10.603426518
my:      1.499998181
sy:      1.266198972
vy:      1.603259838
y1:      0.000597508
y2:      8.902241436
cv:      1.406020794
```

See also:

Population parameters
Polynomial regression

Argument Page



```
v1:  53.680027917857
v2:   2.909820321336
v3:  56.996301792516
v4:   1.684398553911
v5:   0.745554193168
```

See also:

- Argument variables
- Systems of non-linear equations
- Function optimization
- Data point and distribution fitting

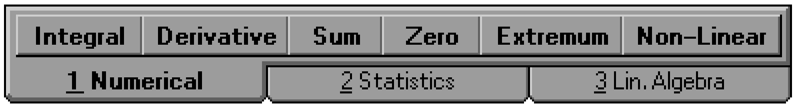
Application Panel

This panel has separate pages with buttons for accessing some of the advanced operations:

- Numerical Page
- Statistics Page
- Linear Algebra Page

The Application Panel can be hidden by going to the Preferences dialog and setting Layout to Standard or Minimal (as opposed to Advanced).

Numerical Page

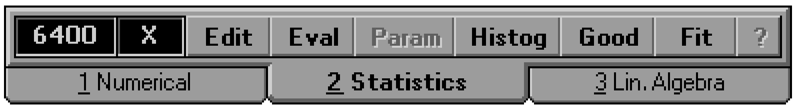


See also:

Numerical operations

Systems of non-linear equations

Statistics Page

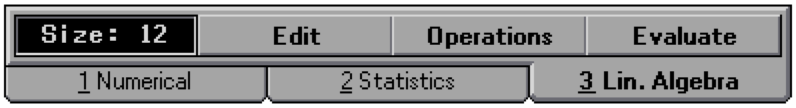


See also:

Statistical operations

Data Editor

Linear Algebra Page



See also:

Linear algebra

Graph Panel

This panel has buttons used to graph functions and to manipulate the contents of the graph canvas.

Evaluating expressions

First, enter the expression into the Expression Line. Use the mouse or type the expression in. In the latter case you will need to use special keystroke combinations to enter some of the symbols.

The expression syntax follows (mostly, at least) the customary "blackboard" notation. The maximum expression length is 64 characters or 36 tokens (operators, variables, constants or separators); in the registered version these limits have been increased to 128 characters or 72 tokens.

After the expression has been entered, you have to tell Kalkulator where to store the result: press the "=" key or click on the Result Variable display, then select a variable from a pop-up menu. Instead of doing that, you may just press the Enter key – the result will then be stored in the Result Variable as currently shown (the same will happen if you double-click on the Result Variable).

If the expression syntax is not proper, an error will be reported and you will be able to correct the mistake. The cursor will be placed at the error.

In no case should an error in an expression make Kalkulator crash or misbehave in any other way. All that may happen is an error alert, specifying the function or operator where the error occurred.

Otherwise Kalkulator will try to evaluate the expression. Some errors may occur at that stage (e.g., division by zero, or referring to a variable without a previously assigned value). If this is the case, another error alert will show up.

The result of the calculation will be shown in the Result Display. The **[Format]** button can be used to change the display precision and/or format.

The evaluated expression stays displayed in the Expression Line (unless the Auto Scroll option is on); it will be highlighted which means that it is subject to be erased by any subsequent input. Following the Windows standard, any cursor movement will remove the highlighting and allow editing of the expression without erasing it.

If the highlighted expression is erased by accident, pressing the Insert key will restore it to the original state. This, in turn, can be undone by pressing Insert again.

See also:

Checking expression syntax
Expression syntax
Accuracy
Auto scroll
Implicit use of the last result

Checking expression syntax

You may check the syntax of the current Expression Line,
without actually computing it, by pressing Ctrl-Enter.
If an error is detected, a diagnostics will be displayed;
if the syntax is OK, the expression will just blink green.

Highlighted?

The background and foreground colors of highlighted text depend on your current Windows settings.
It is desirable to have these colors set differently than the Kalkulator display colors (green or yellow on black).

Expression syntax

Functions and operators are executed from high to low priority:

- High: power, square, cube, factorial, implied multiplication, functions
- Medium: multiplication and division
- Low: addition and subtraction, unary plus and minus

An operand with operators on both sides will associate with the higher priority operator of the two. If both priorities are the same, then the left operator (low and medium priority) or the right one (high priority) is applied first.

This order, following the customary "blackboard" notation, can be arbitrarily changed with parentheses. Of course, you may always do it when in doubt.

See also:

Expression syntax examples
Syntax restrictions
Upper and lower case
Implied multiplication

Expression syntax examples

```
2*sin 2x  means 2*sin(2*X)
2sin 2*x        2*sin(2)*X
a^b^cd          A^(B^(C*D))
a+b*cd^3.1      A+(B*(C*(D^3.1)))
a sin hx        A*sin(H*Xr)
asinh x         asinh(X)
ln x²           ln(X²)
```

Upper and lower case

Expressions do not have to be entered in a case-sensitive manner. If a lowercase letter does not make sense in a given context, it may be raised to uppercase (becoming a variable).

This is convenient: most of us prefer to type **2axsinyh** rather than **2 AX sin YH**, letting the computer to sort it out.

Note, however, that typing **2xasinhy** will be interpreted as **2X asinh Y**: the program recognizes **asinh** as a function name and promotes to uppercase only as few characters as necessary.

In case of doubt, you may always add a separating space, or enter a variable name in upper case (it will never be demoted to lower).

Implied multiplication

The multiplication operator, ×, between two operands or symbols can be skipped if

- the first one is a variable, a constant, a postfix operator or a closing parenthesis (except when used for function arguments), **and**
- the second one is a variable, a symbolic constant, a function, or an opening parenthesis.

The difference between explicit and implied multiplication is that the latter has higher priority.

Examples:

Legal	Equivalent to	Illegal
3VXZ	3*V*X*Z	VX3Z
2¶R	2*¶*R	2 3.14R
9 sin X	9*sin(X)	sin(Y)sin X
ln X exp Y	ln(X*exp(Y))	ln(X)exp Y
sin 2AB	sin(2*A*B)	sin(2A)B
sin X(A+B)	sin(X*(A+B))	sin(A+B)X
¶R^2	¶*(R^2)	R^{2}3.14
(A+B)(C+D)	(A+B)*(C+D)	ln(A+B)(C+D)

Note

The traditional notation is not quite consistent: **exp 3 ln X** is usually interpreted by a human reader as **exp(3×ln(X))** while **2 sin X cos X** as **2×sin(X)×cos(X)**. Kalkulator always chooses the first way, interpreting the latter example as **2×sin(X×cos(X))**.

See also:

Syntax restrictions

Syntax restrictions

The "blackboard" notation is not always clear and consistent. Therefore, to avoid ambiguous expressions, we have to introduce a number of (hopefully, reasonable) restrictions.

- High-priority operators: **^{2}**, **^{3}**, **!**, **^**, implied multiplication, cannot follow a parenthesis enclosing a function argument (or arguments)

For example, the sequence **log(x)^2** is illegal; it should be changed into **(log x)^2** or **log(x^2)**, depending on the intended meaning. On the other hand, **log x^2** is OK and means **log(x^2)**.

- Sequences like **sin^{2}x**, often used to denote **(sin x)^2**, are not recognized; the latter, explicit form has to be used.

See also:

Implied multiplication

Variables

Variables can be used as elements of expressions and to store results of operations.

- There are 23 "regular" variables, from **A** to **Z** (excluding E, T and O). They are entered in upper case, although Kalkulator will recognize them anyway.
- **t** (lowercase!) is the Temporary Variable, used to store the last result sent to the Result Display (even if another variable was also used to store the result).
- When an expression is interpreted as a function (e.g., user-defined, or in function plotting or in numerical operations), **t** is understood as the argument (dummy parameter) of that function. In multi-argument user-defined functions dummy variables **t1**..**t3** are used.
- There are also argument variables, **v1**..**v9**, used to store initial and final solutions in systems of non-linear equations, in function extremum search (optimization) and in statistical data fitting. (In non-registered copies the argument variables go up to **v3** only).
- The symbolic constants, **π** and **é**, behave in expressions exactly like variables, except that they cannot be assigned new values.
- The statistical parameters describing the points in the Data Buffer are also treated like read-only variables, i.e. once computed, they can be used as elements of expressions.

See also:

Accessing variables

Temporary Variable

Denoted as **t**, this variable is used to store the result of the last operation, shown in the Result Display**.** Even if the result is explicitly stored in another variable (**A**..**Z**), it will be stored in **t** as well.

Accessing variables

The buttons in the Variable Button Page can be used to enter variable names into the Expression Line. The same can be done by clicking on variable values shown in the Variable Browser Page, on values shown in the Parameter Page, or in the Argument Page.

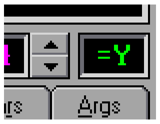
Setting, fetching and clearing variable values

The pages of the Variable Panel contain some buttons used for accessing or setting variable values (with variable selected from a pop-up menu, if applicable):

- **[Get]** – copy a variable value into the Result Display and into the Temporary Variable, **t**
- **[Put]** – store a given value (you will be prompted for it) in a variable
- **[Zap]** – clear all variables in the current page (so that they will have no values assigned)

Additionally, clicking on the Result Display (or pressing Shift-Enter) allows to store the last result in a variable selected from a pop-up menu.

Result Variable



The variable into which the expression value will be stored is shown at the far right below the Expression Line. Clicking here (or pressing the "=" key) will bring up a pop-up menu from which a new Result Variable can be chosen; it will also compute the expression value and store it.

Hitting the Enter key will evaluate the expression and store the resulting value in the Result Variable as currently shown (at the same time storing it in the Temporary Variable, **t**). Double-clicking on the Result Variable will do the same.

Every expression has its own Result Variable assigned. Scrolling through expressions will change the Result Variable display as appropriate.

Clearing the Expression Line (with **[Clr]** or the Escape key) will set the current Result Variable to the Temporary Variable, **t**. Restoring the expression will restore the variable assignment as well.

See also:

Evaluating expressions

Constants

Numeric constants

Constants are used as components of expressions. A constant consists of digits 0..9, an optional decimal point and an optional decimal exponent symbol, "E"; the exponent may be signed or not. For example, **1.5E-3** means the same as **0.0015**. A zero preceding the decimal point can be skipped, as in **.0015**.

A leading minus (or plus) sign is not treated as a part of a constant, but rather as an operator, although this is a purely academic distinction.

When an expression is reformatted before being computed, the constants in it may became reformatted as well, to a (possibly) simpler representation.

Important for European users: regardless of your regional settings, Kalkulator always uses a point, ".", as the decimal separator.

Angular constants

These are entered with use of digits and the symbols of degrees, minutes and seconds, available in the Digit Panel (to enter the degree symbol from the keyboard, use Ctrl-d).

An angular constant used in an expression will be converted to a number, corresponding to the angular value in radians or degrees, depending on the current Kalkulator angular mode.

Symbolic constants

There are two predefined symbolic constants: pi (3.14...) and Euler's e (2.71...). They are denoted with symbols **π** and **é**, respectively, available as buttons on the Digit Panel. To enter them from the keyboard, use the Ctrl-p and Ctrl-e keystroke combinations.

See also:

Physical constants

Physical constants

Kalkulator is aware of some elementary physical constants.

Clicking on the **[Physical]** button in the Digit Panel will bring a pop-up menu from which a constant can be selected. Its value will be stored in the Temporary Variable, **t**, and shown in the Result Display (from where it can be copied into any variable by a mouse click or by pressing Shift-Enter).

Kalkulator uses the 1986 CODATA recommended values, as posted by the National Institute of Standards at

http://physics.nist.gov/PhysRefData/codata86/codata86.html

with all decimal digits as published (even within the error margin).

Operators

Available operators

- The four basic operators: **+**, **-**, **×**, **/**
- The power operator: **^**
- Factorial: **!**
- Square and cube: **²**, **³**
- Implied multiplication: **·** (a tiny dot, sometimes omitted at all)

When entered from the keyboard, some of the operators will require special keystroke combinations.

The multiplication operator, shown as "**×**", is entered with the asterisk key, "*****", which is a commonly accepted procedure.

Operator priority

- High: **^**, **²**, **³**, **!** and **·**
- Medium: **×**, **/**
- Low: **+**, **-**

See also:

Expression syntax
Implied multiplication

Implied multiplication

The "×" operator can often be skipped, as in **3 sin 2X**. Kalkulator will then insert the implied multiplication operator, denoting it with a "·" if it makes the expression more readable - in this case, **3·sin 2X**. The implied multiplication has a higher priority than "×".

For detailed information, click here.

Power operator

This operator, shown as "**^**", raises the first operand to the power of the second, as in **X^Y**. If **Y** is not integer, then **X** has to be greater than zero.

Factorial

Usually defined, for integer arguments only, as *n!=1*2*...*n*. Kalkulator will compute it for non-integer arguments as well, using the Gamma function: *n!=gam(n+1)*. For integer arguments both definitions are equivalent.

Functions

Function arguments

Any arithmetic expression can be used as a function argument. Multiple arguments (in functions which expect them) are separated with commas.

Many of the functions accept just a single argument. If this argument is a number (constant), a variable, or another function call, then it does not need to be enclosed in parentheses. For example, **exp(3.8)-ln(X)** can be entered as **exp 3.8-ln X**, but in **sqr(1+X)** the parentheses are necessary; without them the expression would be understood as **sqr(1)+X**.

Function groups:

Standard functions

These functions, accessible from the Standard Page of the Function Panel, are usually considered "standard", whatever that means.

The following function groups can be identified here:

- Logarithms and exponents base e, 10 and 2: **ln**, **log**, **lgb**, **exp**, **ten**, **two**
- Roots and powers: **sqr**, **sq**, **cur**, **cu**
- Trigonometric and inverse: **sin**, **cos**, **tan**, **cot**, **asin**, **acos**, **atan**, **acot**
- Hyperbolic and inverse: **sinh**, **cosh**, **tanh**, **coth**, **asinh**, **acosh**, **atanh**, **acoth**
- Polar coordinate functions: **ang**, **rad**
- Other functions: **int**, **fra**, **rou**, **mod**, **abs**, **sgn**, **rcp**

Some other, "almost standard", functions can be found in the Extra Function Page:

- Smaller/greater choices: **min**, **max**, **mxa**
- Arithmetic, geometric and harmonic averages: **avg**, **gav**, **hav**

See also:

Angular mode
Extra Function Page

cosh x computes a hyperbolic cosine of *x.*

abs x computes the absolute value of *x.*

acos x computes an arc cosine of *x.*

acosh x computes an inverse hyperbolic cosine of *x.*

acot x computes an arc cotangent of *x.*

acoth x **c**omputes an inverse hyperbolic cotangent of *x.*

ang(x,y) computes the polar angle of a Cartesian point *(x,y)*, i.e. the angle between the x-axis and the line from *(0,0)* to *(x,y)*.

asin x computes an arc sine of *x.*

asinh x computes an inverse hyperbolic sine of *x.*

atan x computes an arc tangent of *x.*

atanh x computes an inverse hyperbolic tangent of *x*.

avg(x,y) computes the average of *x* and *y*: *(x+y)/2*

cos x computes a cosine of *x.*

cot x computes a cotangent of *x.*

coth x computes a hyperbolic cotangent of *x.*

cu x computes a cube (third power) of *x.*

cur x computes a cubic root of *x.*

exp x computes Euler's e to the power of *x.*

fra x   computes the fractional part of *x*.
For negative *x*, *fra(x)* will also be negative.

gav(x,y) computes the geometric average of *x* and *y*: *sqr(xy)*.

hav(x,y) computes the harmonic average of *x* and *y*: *2/(1/x+1/y)*.

int x computes the integer part of *x* (rounded towards zero).

lgb x computes a binary (base 2) logarithm of *x*.

ln x computes a natural (base e) logarithm of *x*.

log x computes a decimal (base 10) logarithm of *x*.

max(x,y) computes the larger value of *x* and *y*.

min(x,y) computes the smaller value of *x* and *y*.

mod(x,y) computes the remainder of the division of *x* by *y*.

mxa(x,y) chooses the argument with the greater absolute value.

rad(x,y) computes the polar radius of the Cartesian point *(x,y)*, i.e. the distance from *(0,0)* to *(x,y)*.

rcp x computes reciprocal of *x*, i.e. *1/x*.

rou(x,y) rounds *x* to the nearest whole multiple of *y*.

sgn x returns the sign of *x*: -1 for negative, 1 for positive and 0 for *x=0*.

sin x computes the sine of *x*.

sinh x computes a hyperbolic sine of *x*.

sq x computes a square (second power) of *x*.

sqr x computes a square root of *x*.

tan x computes a tangent of *x*.

tanh x computes a hyperbolic tangent of *x*.

ten x computes 10 to the power of *x*.

two x computes two to the power of *x*.

Conditional functions

The buttons for these functions are in the Extra Page of the Function Panel.

A conditional function accepts three arguments and, depending on the sign of the first argument, computes and returns the value of argument number two or three:

- **ifz** – "if zero", checks for the first argument being equal zero
- **ifp** – "if positive", checks for the first argument being positive (>0)
- **ifn** – "if negative", checks for the first argument being negative (<0)

The discarded argument is not evaluated at all; thus, for example, **ifz(X,1,sin X/X)** will be computed without an error for any value of **X**, including zero.

Note that functions for checking a "greater or equal" or "less or equal" condition are not necessary. Instead of invoking a hypothetical "if positive or zero" function **ifpz(x,a,b)** we may just use **ifn(x,b,a)**.

ifn(x,y,z) returns *y* if *x<0*, otherwise *z.*

ifp(x,y,z) returns *y* if *x>0*, otherwise *z.*

ifz(x,y,z) returns *y* if *x=0*, otherwise *z.*

User functions

An expression can be used to define a new function of one, two, or three arguments. This new function can then be used in other expressions.

Enter the function formula into the Expression Line, using **t** as the single function argument, or **t1** and **t2** as two arguments, or **t1**, **t2** and **t3** as three. Then click on **[Def]** in the User Page of the Function Panel. Choose a function slot from the pop-up menu, then enter the function name.

This name, consisting of lowercase letters and digits and starting from a letter, has to be different than any other function name.
If no syntax error is detected, the function will now be defined and its name will be shown in one of the user function buttons.

Nesting user functions

One user function may call another, but it cannot refer to itself, directly or indirectly. A check for that is made when an expression using these functions is evaluated.

Renaming and undefining user functions

This is done by clicking on **[Name]** and **[Clear]** in the User Page. You will not be allowed to rename or delete a user function if others are referring to it (you have to undefine those first).

Any references in expressions to a deleted or renamed function will now become invalid.

Displaying and modifying user functions

Clicking on **[Show]** in the User Page will display the function definition in the Expression Line. The function can now be modified and defined again.

Saving and loading user functions

The whole set of user functions can be saved to, or loaded from, a disk file. This way you can keep mini-libraries of functions defined for various applications. The file operations are accessed by clicking on **[File | Save user functions]** and **[File | Load user functions]**.

Euler and related functions

These functions are grouped in the bottom row of the Statistical Page of the Function Panel:

- **gam** is the Euler Gamma function, related to the factorial operator: *x!=gam(x+1)*.
- **bet** is the Euler Beta function
- **lgm** is the logarithm of Gamma (handles larger arguments)
- **com** is the combination function

Both **gam** and **bet** are complete Euler functions; the incomplete ones can be computed with use of the Gamma and Beta integral distribution functions, **pgam** and **pbet**.

gam x returns the Euler gamma function of *x*, for positive and non-integer negative arguments.

bet(x,y) computes the Euler Beta function of *x* and *y*, defined as *Gamma(x)*Gamma(y)/Gamma(x+y)*.

lgm x computes a natural logarithm of Gamma of *x*.

lgm x computes a natural logarithm of the Euler Gamma function of *x*.

com(n,k) computes the combinations of *n* by *k*, i.e. *n!/[k!(n-k)!]*, where *n* and *k* do not have to be integers.

Fetch functions

These functions allow you to access values from Data Buffer or the matrix and vector elements in order to use them as components of expressions. Their buttons can be found in the Extra Page of the Function Panel.

The data fetch functions are **x#** and **y#** (lowercase characters!). The latter function is available only in the buffer XY mode; an attempt to use it in the X mode will cause an expression evaluation error.

Similar fetch functions are used to access the values of linear algebra objects: **u#**, **v#** and **w#** for vectors, and **a#**, **b#** and **c#** – for matrices. The latter three functions use two arguments, first being the row, second – the column index.

An argument of a fetch function will be rounded to the nearest integer; its value can be from 1 to the number of points in the buffer or to the current size of linear algebra objects, respectively. A value from outside the legal range will cause an expression evaluation error.

Examples

```
 (x#1+x#(i+1))/2
x#max(t-1,1)
```

Polynomial

There is one more function related to the data in the buffer: **poly**. It will use the current set of polynomial regression coefficients to compute a polynomial value for its argument. If the argument is **t**, then the polynomial is defined as **a0+a1×t+a2×t^2+...**

a#(i,j) returns the value of the (i,j)-th element of matrix **a**, where i is the row and j - the column.

b#(i,j) returns the value of the *(i,j)*-th element of matrix **b**, where *i* is the row, and *j* - the column.

c#(i,j) returns the value of the *(i,j)*-th element of matrix **c**, where *i* is the row, and *j* - the column index.

u#i returns the value of the *i*-th element of vector **u**.

v#i returns the value of the *i*-th element of vector **v**.

w#i returns the value of the *i*-th element of vector **w**.

x#i returns the *i*-th x-value
from the Data Buffer.

y#i returns the *i*-th y-value from the Data Buffer in the XY mode.

poly(x) computes a polynomial of *x* with coefficients given by data buffer parameters **a0**, **a1**... etc.

Feet, pounds, hours

This group consists of three very similar functions, accessible from the Extra Page of the Function Panel:

- **ft** converts feet and inches into feet
- **lb** converts pounds and ounces into pounds
- **hrs** converts hours, minutes and seconds into hours

Example

This is how to compute the area of a rectangle 6 ft 7 in by 4 ft 3 5/8 in:

```
ft(6,7)*ft(4,3+5/8)
```

ft(x,y) converts *x* feet and *y* inches into feet, as *x+y/12*.

lb(x,y) converts *x* pounds and
y ounces into pounds, as *x+y/16*.

hrs(x,y,z) converts *x* hours, *y* minutes and *z* seconds into hours, as *x+y/60+z/3600*.

Statistical distribution functions

Each of these functions, accessible   from the Statistical Page of the Function Panel, can be identified by specifying two features: the distribution and the function type.

The top row buttons determine the type of the distribution function entered by the subsequent button clicks.

When the function type has been chosen, clicking on one of the eight buttons below the icons selects the distribution.

Accuracy

All distribution functions are usually computed with at least sixteen good decimal digits. (In a few cases, near distribution tails, this cannot be achieved within the 19-digit internal accuracy; any of these exceptions are reasonable and justified.)

For detailed information, see:

Types of distribution functions
Supported distributions
Glossary of distribution functions
Fractional arguments to distribution functions

Types of distribution functions

The buttons in the top row of the Statistical Function Page determine the type of the distribution function entered by the subsequent button clicks.

The first four have their respective types shown by icons accompanied with a single letter. This letter denotes the function prefix as it will be shown in the Expression Line (and as it should be entered, in lowercase, from the keyboard). Our notation is similar to the one used in the Handbook of Mathematical Functions:

- the **[F]** button – a distribution density (or probability)
- the **[P]** button – a left distribution integral
- the **[Q]** button – a right distribution integral
- the **[A]** button – a central distribution integral (symmetric distributions only)

For the discrete distributions (Poisson and binomial) the term "distribution" density" should be replaced with "probability" and "integral" with "sum".

The rightmost button, **[I]**, used together with one of the discussed above (P, Q or A), denotes an inverse integral distribution.

See also:

Supported distributions

Handbook of Mathematical Functions
Abramowitz and Stegun, National Bureau of Standards.

Distribution density

Sometimes referred to as differential distribution function: $f(x) = dP(x)/dx$.

Left distribution integral

Also called a cumulative distribution function, *P(x)*. This is an integral up to *x* from *f(x)*, i.e. the probability that the random variate will assume a value less than or equal to *x*.

Right distribution integral

The integral of *f(x)* from *x* up, or the probability that the random variate will assume a value greater than *x*. Note that *Q(x)=1-P(x)*.

Central distribution integral

The integral from *m-(x-m)* to *x* of *f(x)*, i.e. the probability that the random variate will assume a value as close to the distribution center, *m*, as *x* or closer. For *m=0* this means just between -x and x.

Inverse integral distribution

More precisely: a function inverse to a cumulative (integral) distribution function. These functions compute a random variate value corresponding to a given probability integral.

For example, while the prefix "p" in the distribution function name means "find the integral probability for a given value of the random variate", the "ip" prefix means "find the random variate value for which the integral probability is such and such".

Supported distributions

The following statistical distributions are supported by Kalkulator. The three-letter distribution abbreviations listed below, are additionally prefixed with one or two letters denoting the distribution function type.

- **gau** – the Normal (Gaussian) distribution
- **chi** – the chi-square distribution
- **stu** – the t (Student) distribution
- **sne** – the F (Snedecor) distribution
- **poi** – the Poisson distribution
- **bin** – the binomial distribution
- **gam** – the gamma distribution
- **bet** – the beta distribution

See also:

Types of distribution functions

Beta distribution

Density: *fbet(x,a,b)* for *0<=x<=1*
and parameters *a>0* and *b>0*.

Binomial distribution

Probability: **fbin(x,n,p)** for *0<=x<=n* and *0<p<1* where *n* is the total number of tries and *p* is the probability of a success in a single try.

Chi-square distribution

Density: **fchi(x,r)**, for *x>=0* and *r>0* degrees of freedom.

F (Snedecor) distribution

Density: *fsne(x,r,s)* for *x>=0* and for *r* and *s* degrees of freedom.

Gamma distribution

Density: *fgam(x,a)* for *x>=0* and for the parameter *a>0*.

Normal (Gaussian) distribution

Density: fgau(x,m,s), where m is the mean value and s>0 - the standard deviation.

Poisson distribution

Probability: *fpoi(x,m)* for *x>=0* and mean *m>0*.

t (Student) distribution

Density: *fstu(x,r)* for *r* degrees of freedom.

Glossary of distribution functions

Here is the complete list of distribution functions provided by Kalkulator:

- Distribution density functions: **fgau**, **fchi**, **fstu**, **fsne**, **fgam**, **fbet**, **fpoi**, **fbin** (for discrete distributions: Poisson and binomial, these are probability, not density functions)
- Left distribution integral functions: **pgau**, **pchi**, **pstu**, **psne**, **pgam**, **pbet**, **ppoi**, **pbin**
- Right distribution integral functions: **qgau**, **qchi**, **qstu**, **qsne**, **qgam**, **qbet**, **qpoi**, **qbin**
- Central distribution integral functions: **agau**, **astu**
- Left inverse integral distribution functions: **ipgau**, **ipchi**, **ipstu**, **ipsne**, **ipgam**, **ipbet**, **ippoi**, **ipbin**
- Right inverse integral distribution functions: **iqgau**, **iqchi**, **iqstu**, **iqsne**, **iqgam**, **iqbet**, **iqpoi**, **iqbin**
- Central inverse integral distribution functions: **iagau**, **iastu**

See also:

Types of distribution functions
Supported distributions

agau(x,m,s) returns the integral from m-(x-m) to x of the Gaussian (Normal) distribution with mean m and standard deviation s.

astu(x,r) returns the integral from -*x* to *x* of the t (Student) distribution with *r* degrees of freedom.

fbet(x,a,b) returns the distribution density at *x* for a beta distribution with parameters *a* and *b*.

fbin(x,n,q) returns the distribution probability at *x* (integer) for a binomial distribution with parameters *n* (no. of tries) and *q* (single-try success probability).

fchi(x,r) returns the distribution density at *x* for a chi-square distribution with *r* degrees of freedom.

fgam(x,a) returns the distribution density at *x* for a gamma distribution with the parameter *a*.

fgau(x,m,s) returns the distribution density at *x* for a Gaussian (Normal) distribution with mean *m* and standard deviation *s*.

fpoi(x,m) returns the distribution probability at *x* (integer) for a Poisson distribution with a mean value *m*.

fsne(x,r,s) returns the distribution density at *x* for an F (Snedecor) distribution with *r* and *s* degrees of freedom.

fstu(x,r) returns the distribution density at *x* for a t (Student) distributions with *r* degrees of freedom.

ppoi(x,m) returns the probability sum from *0* to *x* (integer) of the Poisson distribution with the mean value *m*.

iagau(p,m,s) finds a value of *x*, for which *agau(x,m,s)=p*, for a Gaussian (Normal) distribution with mean value *m* and standard deviation *s*.

iastu(p,r) finds a value of *x*, for which *astu(x,r)=p*, for a t (Student) distribution with *r* degrees of freedom.

ipbet(p,a,b) finds a value of *x*, for which *pbet(x,a,b)=p*, for a beta distribution with parameters *a* and *b*.

ipbin(p,n,q) finds a value of *x*, for which *pbin(x,n,q)=p*, for a binomial distribution with parameters *n* and *q*.

ipchi(p,r) finds a value of *x*, for which *pchi(x,r)=p*, for a chi-square distribution with r degrees of freedom.

ipgam(p,a) finds a value of *x*, for which *pgam(x,a)=p*, for a gamma distribution with parameter *a*.

ipgau(p,m,s) finds a value of *x*, for which *pgau(x,m,s)=p*, for a Gaussian (Normal) distribution with mean value *m* and standard deviation *s*.

ippoi(p,m) finds a value of *x*, for which *ppoi(x,m)=p*, for a Poisson distribution with mean value *m*.

ipsne(p,r,s) finds a value of *x*, for which *psne(x,r,s)=p*, for an F (Snedecor) distribution with *r* and *s* degrees of freedom.

ipstu(p,r) finds a value of *x*, for which *pstu(x,r)=p*, for a t (Student) distribution with *r* degrees of freedom.

iqbet(p,a,b) finds a value of *x*, for which *qbet(x,a,b)=p*, for a beta distribution with parameters *a* and *b*.

iqbin(p,n,q) finds a value of *x*, for which *qbin(x,n,q)=p*, for a binomial distribution with parameters *n* and *q*.

iqchi(p,r) finds a value of *x*, for which *qchi(x,r)=p*, for a chi-square distribution with *t* degrees of freedom.

iqgam(p,a) finds a value of *x*, for which *qgam(x,a)=p*, for a gamma distribution with parameter *a*.

iqgau(p,m,s) finds a value of *x*, for which *qgau(x,m,s)=p*, for a Gaussian (Normal) distribution with mean value *m* and standard deviation *s*.

iqpoi(p,m) finds a value of *x*, for which *qpoi(x,m)=p*, for a Poisson distribution with mean value *m*.

iqsne(p,r,s) finds a value of *x*, for which *qsne(x,r,s)=p*, for an F (Snedecor) distribution with *r* and *s* degrees of freedom.

iqstu(p,r) finds a value of *x*, for which *qstu(x,r)=p*, for a t (Student) distribution with *r* degrees of freedom.

pbet(x,a,b) returns the integral from *0* to *x* of the Beta distribution with parameters *a* and *b*. It is equivalent to the incomplete Beta function *ibet(a,b,x)*.

pbin(x,n,q) returns the probability sum between *0* and *x* (integer) for a binomial distribution with parameters *n* and *q*.

pchi(x,r) returns the integral from *0* to *x* of the chi-square distribution with *r* degrees of freedom.

pgam(x,a) returns the integral from *0* to *x* of the Gamma distribution with the distribution parameter *a*. It is equivalent to incomplete Gamma function *igam(a,x)*.

pgau(x,m,s) returns the integral from -∞ to *x* of the Gaussian (Normal) distribution with mean *m* and standard deviation *s*.

psne(x,r,s) returns the integral from *0* to *x* of the F (Snedecor) distribution with *r* and *s* degrees of freedom.

pstu(x,r) returns the integral from minus infinity to *x* of the t (Student) distribution with *r* degrees of freedom.

qbet(x,a,b) returns the integral from *x* to *1* of the Beta distribution with parameters *a* and *b*.

qbin(x,n,q) returns the probability sum between *x+1* (integer) and *n*, for a binomial distribution with parameters *n* and *q*.

qchi(x,r) returns the integral from *x* to ∞ of the chi-square distribution with *r* degrees of freedom.

qgam(x,a) returns the integral from *x* to ∞ of the Gamma distribution with the distribution parameter *a*.

qgau(x,m,s) returns the integral from *x* to ∞ of the Gaussian (Normal) distribution with mean *m* and standard deviation *s*.

qpoi(x,m) returns the probability sum from *x+1* (integer) to ∞ of the Poisson distribution with the mean value *m*.

qsne(x,r,s) returns the integral from x to ∞ of the F (Snedecor) distribution with r and s degrees of freedom.

qstu(x,r) returns the integral from *x* to ∞ of the t (Student) distribution with *r* degrees of freedom.

Fractional arguments to distribution functions

While most of the distributions deal with random variates in the real domain, the binomial and Poisson distributions are usually considered for discrete (integer) arguments. Nevertheless, Kalkulator will compute these distribution functions also for fractional values, using the same formulae.

Similarly, some of the distribution parameters are in most applications integer (e.g., degrees of freedom). Kalkulator will, however, accept non-integer distribution parameters, and use the same formulae as in the integer cases.

(There are statistical methods using fractional degrees of freedom: e.g., the test for equality of two sample means without assumption of equal variances.)

Random number generators

These functions, accessible from the Statistical Page of the Function Panel, return randomly (strictly speaking: pseudorandomly) a different result every time they are used. The randomness is organized, however: for each generator its results are sampled from a distribution defined by the generator function and its arguments.

Three generators are provided, accessed with buttons in the Statistical Page of the Function Panel:

- **ran** – sampling from a uniform distribution
- **rgau** – sampling from a Normal (Gaussian) distribution
- **rpoi** – sampling from a Poisson distribution

Reseeding the generators

Every time you run Kalkulator, the sequence of basic pseudorandom numbers used in the generators described above starts from the same seed. This means, that if in two different sessions with the program you repeat the same sequence of generator function calls, the results will be identical.

This is not always desirable. This is why the registered version of Kalkulator allows you to initialize (reseed) the generators with any seed value of your choice. To do it, use the **[Options | Reseed generators]** menu entry.

ran x   returns a random value from
a uniform distribution between *0* and *x*.

rgau(m,s) returns a random value from a Gaussian (Normal) distribution with mean *m* and standard deviation *s*.

rpoi m returns a random value (integer) from a Poisson distribution with a mean value *m*.

Uniform distribution

A distribution with density constant in a given interval and zero outside of it.

Angular mode

The current angular mode is shown as **[Rad]** or **[Deg]** in a button to the right of the Result Display. Clicking on that button will change the mode.

Depending on this mode, Kalkulator assumes that arguments to trigonometric functions are expressed in radians or degrees. It will also return results of inverse trigonometric functions in the same units.

Changing the angular mode to degrees can often lead to some confusion. I would recommend staying in the radian mode and using the angular constants expressed explicitly in degrees or dms. Then the calculation results can be displayed in the desired angular representation as needed.

See also:

Formatting the result

Implicit use of the last result

After an expression evaluation is complete, its result will be always stored in the Temporary Variable, **t** (in addition to any variable you have chosen to store it). Therefore, by computing next **t+5** you will actually add 5 to the last result.

This process can be further automated by checking the "Implicit t" box in the Preferences dialog. With that option checked, whenever you start entering a new expression from one of the basic four operators: **+**, **−**, **×** or **/**, the **t** variable will be automatically appended in front of it. In the example above, you will have just to enter **+5**, and Kalkulator will change that to **t+5**.

The implicit entry of **t** will happen only if you are starting a new expression, i.e. when the Expression Line is either empty or highlighted.

Obviously, sometimes you have a legitimate need to start an expression from a unary **−**, like in **−2X^2−7**, and you would not like it to become **t−2X^2−7**. Tough luck, you cannot have both at the same time. Either disable the "Implicit t" option, or insert a **0** at the front, or type the space in front of the minus operator.

Auto scroll

Usually when an expression has been evaluated, it stays (highlighted) in the Display Line. You can, however, enable the autoscroll option (see the Preferences dialog).

With autoscroll enabled, upon a successful expression evaluation Kalkulator progresses to the next expression (wrapping back to 1 at the end). The contents of the Expression Line will be then highlighted, so that you can immediately start entering a new expression or evaluate the old one by just hitting the Enter key.

This allows you to set up a sequence of expressions (each one possibly using results of the previous ones), to be evaluated by repeatedly pressing Enter.

See also:

Evaluating expressions
Result Variable

Accuracy

The internal accuracy of individual arithmetic operations performed by Kalkulator is 19 good decimal digits or better. These errors will, however, accumulate in computations involving multiple operations, so that 18 good digits is a safer assumption.

Moreover, algorithms used to compute some less trivial functions may have their own, additional error margin.

Kalkulator displays up to eighteen significant digits of results. In a great majority of cases all these digits will be valid with a reasonably large safety margin.

(Keep in mind, however, that for any finite floating-point accuracy one can easily invent an example of a computation where the relative result error will be 100% or more – i.e., all decimal digits wrong.)

See also:

Problem with Microsoft System Agent
The Pentium bug

Formatting the result

Clicking on the Format button to the left of the Result Display offers (from a pop-up menu) a selection of various display formats.

- **Default** – a reasonable compromise, the choice between floating and fixed point is made based on number readability. The values are shown with the full accuracy of 16 significant digits, and the trailing insignificant zeros are omitted.
- **Fixed Point** – a fixed-point representation, with a selected number of digits after the decimal point (if the selected number would result in more than 16 significant digits, it will be appropriately reduced).
- **Exponential** – a floating-point display, with a selected number of significant digits (up to 16) and at least two digits in the decimal exponent.

Any of these three settings becomes "permanent" – it will hold for all future calculations until revoked. The other settings reformat just the current result.

- **Fraction** – the value is shown as a whole part and a fraction, e.g., 3.125 becomes "3 1/8". The largest denominator is 1000 and it will be used if the value cannot be exactly represented with a smaller one.
- **Deg-Min-Sec** – the value is shown in degrees, minutes and seconds of angle.
- **Degrees** – the value is shown as degrees of angle (with decimal fraction, if any).
- **Minutes** – minutes of angle.
- **Seconds** – seconds of angle (format used by astronomers).

The four angular formats above are affected by the current angular mode setting, Rad or Deg, as the result is assumed to be an angle, originally expressed in radians or degrees, respectively.

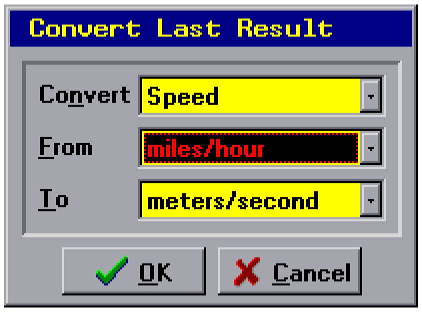
- **Feet & Inches** – the result is assumed to be in feet and it will be shown as feet, inches and fraction. For example, 1.24 will become "1 2 7/8" (one foot, two inches and 7/8 of an inch).
- **Lbs & Ozs** – the result is treated as pounds and shown as pounds, ounces and fraction.

The maximum denominator value for inches or ounces is 64.

- **Hrs-Min-Sec** – the result is treated as time in hours and displayed as e.g., 3:32:41

Obviously, depending on the displayed value and the format used, the display may be rounded. This never affects the actual result value, stored internally with 19 significant digits of accuracy.

Unit conversion



The value shown in the Result Display can be converted from one kind of measurement units into another. This will affect not only the display, but also the actual value of the variable to which it refers.

Clicking on the **[Convert]** button will bring up the Unit Conversion Window. It has three combo boxes with drop-down lists of items to choose from.

- The type of units, i.e. the physical quantity being converted (area, mass, etc.)
- The units to convert from – only these relevant to the selected type will be available
- The units to convert to (ditto)

The actual conversion will take place when we exit the window by clicking on **[OK]**. Note that the variable storing the result will also be affected by the conversion.

Note: although the temperature is not, strictly speaking, a physical quantity, temperature conversion has been also included.

Keyboard

In the Main Window, most of the keyboard input goes into the Expression Line: all characters, Left and Right arrow keys, Home and End, Delete and Backspace. (Characters illegal in expressions will be just ignored, and characters without standard keyboard equivalents can be entered with use of special keystroke combinations.)

Additionally, all menu entries and many buttons have keyboard shortcuts defined in a standard Windows way: Alt + the underscored letter key.

To allow for using Kalkulator entirely without a mouse, some Main Window operations have been assigned to keyboard shortcuts which are not obvious at a glance:

- **Up** and **Down** arrows – switch to the previous/next expression
- **Page Up**, **Page Down** – scroll through the values displayed in the Variable Panel (if the Browser, Parameters, or Argument page is shown)
- **Alt-Del** or **Escape** – clear the Expression Line (storing the expression in a buffer); if the Expression Line is already empty, Escape will copy to it the contents of the buffer.
- **Alt-Ins** – copy the buffer into the current line. If the line is not empty, it will be stored in the buffer (so that, effectively, a swap takes place).
- **Ins** – restore the current line to its last (if any) highlighted state. To reverse this operation press Ins again.
- **Enter** – compute the current expression, storing the result in the Temporary Variable, **t**.
- **Ctrl-Enter** – check the current expression syntax, without evaluating it
- **Shift-Enter** – copy the value from Result Display into a variable (chosen from a pop-up menu)
- **Alt-Left**, **Alt-Right** – redmark the previous/next button in the current page of the Application Panel
- **Alt-Enter** is equivalent to clicking on the redmarked Application Panel button
- **F1** – call Help (Table of Contents)
- **F10** – show the Main Popup Menu
- **F12** – copy the Result Display into the Windows clipboard, so that it can be pasted into any Windows application supporting clipboard operations

When the Graph Canvas is visible, the **Enter** key changes its meaning, being used to hide the canvas. (Enter does not make much sense with the Result Display hidden.) At the same time, the F12 key will copy the Graph Canvas to the Windows clipboard.

The clipboard functions of **Shift-Del** (cut), **Ctrl-Ins** (copy) and **Shift-Ins** (paste) as defined in the standard Windows environment, can also be used in Kalkulator, applied to highlighted fragments of the Expression Line.

See also:

Mouse

Keystroke combinations

Ctrl-p	enters pi, **π**
Ctrl-e	enters the Euler's e, shown as **é**
Ctrl-d	enters the degree symbol, **°**
Ctrl-2	enters the square, ^{2}
Ctrl-3	enters the cube, ^{3}

Mouse

Kalkulator uses the mouse very much like any other Windows 3.1 or Windows 95 program. In particular, all menu items and all functions assigned to buttons are accessible with single clicks of the left mouse button.

The non-obvious uses of the left mouse button include:

- Clicking in the Browser or Parameter Page of the Variable Panel enters the selected variable (or a statistical parameter) into the Expression Line at the current cursor position.
- Clicking on the Result Display allows to store the displayed value in a variable, selected from a pop-up menu.

The right mouse button has been assigned a few special functions:

- When the Menu Bar has been removed with use of the Preferences dialog, clicking the right mouse button at the edge of the Main Window (the mouse cursor will change to show you are there) will bring up the Main Pop-up Menu.
- A right click in the Variable Panel (any of the pages) will show a pop-up menu with the value of the object (variable or parameter) under the cursor, displayed with the full accuracy.
- A right click within any of the data grids will display the selected value in the same fashion.

Right mouse clicks are among the few Kalkulator operations which do not have keyboard equivalents.

A few Kalkulator operations entering symbols into the Expression Line cannot be performed with the mouse and have to be done from the keyboard:

- Entering the user function arguments, **t1**, **t2** and **t3**
- Entering the argument variables **v1**...**v9**
- Entering the statistical parameters

In the last two cases a symbol can be entered by clicking on its line in the appropriate page of the Variable Panel (but it has to be shown there to start with).

See also:

Keyboard

Numerical operations

Most of these operations are performed on a single-argument function *f(t)*, defined by the current expression, displayed in the Expression Line. In these cases the dummy variable **t** has to be used as the function argument. (Exceptions from this rule are function optimization and systems of nonlinear equations, described elsewhere.)

Clicking on **[Integ]**, **[Deriv]**, **[Sum]** or **[Zero]** in the Numerical Page of the Application Panel will first bring a dialog box to define the parameters of the operation (e.g. argument limits, required accuracy). When these parameters have been entered, the operation is carried out, and upon its completion the result is stored in the Temporary Variable, **t**.

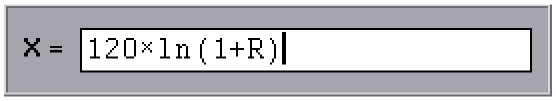
The computation can be interrupted while the red timer box is being displayed, by pressing any key (or clicking inside that box).

See also:

Computing integrals
Computing derivatives
Computing sums
Finding function zeros
Systems of nonlinear equations
Function optimization
Entering numerical values

Entering numerical values

At some times Kalkulator will ask for some floating-point numerical values (e.g., plot range, integration interval limits, a new value for a variable, etc.) to be entered into editable fields of a dialog box (the one below is just an example).



The program will accept here not just literal values (i.e. numbers in a usual notation), but also any legal arithmetic expressions, following the Kalkulator expression syntax (with functions, variables and all). In case of syntax or evaluation errors, the appropriate diagnostics will be provided.

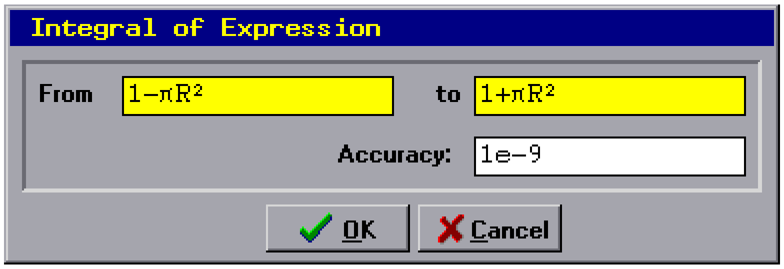
(In other words, any entry field becomes by itself a mini-Kalkulator. In order to compute a value and store it in a variable it is not necessary to go via the Expression Line: you may just click on **[Put]** in the Variable Panel and type the expression into the data entry box!)

If the dialog box stays open after the entered expression has been evaluated (this is the case with some of the dialog boxes), the entry field will show just its value, not the expression itself. The display is limited to twelve decimal digits or so, but as long as the displayed text remains unmodified, the value is remembered with full Kalkulator accuracy. Once you type anything into the field, its contents will be re-parsed and evaluated again. (The same is true when the dialog is shown the next time, showing the previously entered values.)

The fields expecting integer (as opposed to floating-point) values do not accept expressions.

Computing integrals

After a click on **[Integ]**, Kalkulator will compute a numerical integral of the current expression treated as a function of the dummy variable **t**.


Integral of Expression

From 1−πR²   to 1+πR²

Accuracy: 1e−9

A dialog box will prompt you for the integral limits, *t1* and *t2*, and for the desired accuracy parameter, *d*.

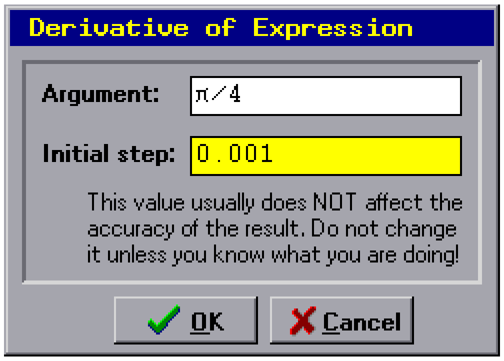
The iterative computation process terminates when the difference between two consecutive approximations, *h*, is less than *d*(1+abs P)*, where *P* is the last iteration result. This means that for *P<<1* the value of *d* denotes an absolute accuracy, while for *P>>1* -- a relative one.

See also:

Numerical operations
Entering numerical values

Computing derivatives

When the **[Deriv]** button is clicked on, Kalkulator will compute the derivative of the current expression, treated as a function of the dummy variable **t**.



Argument: π/4

Initial step: 0.001

This value usually does NOT affect the accuracy of the result. Do not change it unless you know what you are doing!

You will be prompted for the value of **t**, at which the function derivative, *df(t)/dt*, is to be computed.

The other field in the query dialog is the initial step of the step-adjusting algorithm used here (this is not the denominator of the finite increment approximation!). For most of well-behaved functions it never needs adjustments.

The result will be stored in the Temporary Variable, **t**.

Obviously, if you attempt to compute the derivative at a point where the function is not continuous, funny things may happen.

If you would like to learn more about the algorithm used here, refer to the Numerical Recipes, an indispensable source on numerical methods.

See also:

Numerical operations
Entering numerical values

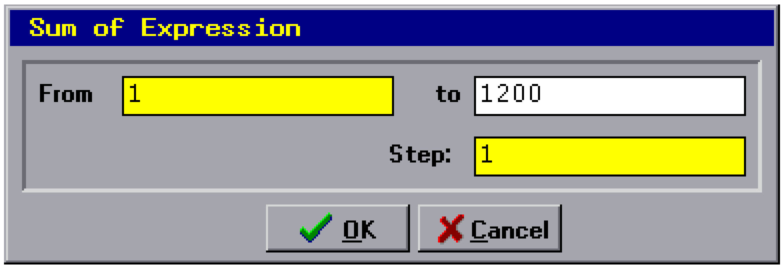
Numerical Recipes, The Art of Scientific Computing
by W.H.Press, B.P.Flannery, S.A.Teukolsky and W.T. Vetterling,
Cambridge University Press, many editions in C, Pascal and FORTRAN.

The book is an absolute must for anyone serious about scientific or engineering number crunching (and quite a delight ro read, too!).

Set your Web browser to **http://www.nr.com**, from where you can even download the complete book in the Acrobat format and read it on screen, or print one copy for your personal use.

Computing sums

Kalkulator will compute an indexed sum with term values defined by the current expression, with the dummy variable **t** used as the term index.



From: 1
to: 1200
Step: 1

After clicking on **[Sum]**, a dialog box will ask you for the index start and end value, *t1* and *t2*, and for the step, *dt*.

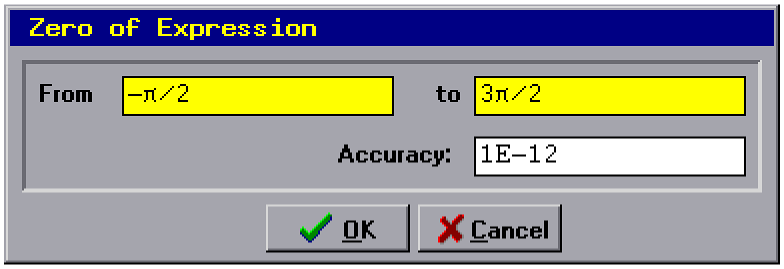
The sum will be computed as *f(t1)+f(t1+dt)+...+f(t2)*.
More precisely, if *(t2-t1)/dt* is not an integer, then the last value of **t** will be *t1+k*dt*, where *k* is the largest integer for which *t1+k*dt < t2+dt/2*. This assures that any rounding errors are handled properly.

See also:

Numerical operations
Entering numerical values

Finding function zeros

This operation finds the argument value for which the current expression, treated as a function of the dummy variable **t**, equals zero: *f(t)=0*.



From −π/2 to 3π/2

Accuracy: 1E−12

After a click on **[Zero]**, a dialog box will ask for the interval *(t1,t2)*, bracketing the solution, and for the desired absolute accuracy, *dt*.

The interval *(t1,t2)* has to contain a single solution (or an odd number of solutions, in which case one of them will be found). If this is not the case, an error will be diagnosed.

See also:

Numerical operations
Entering numerical values

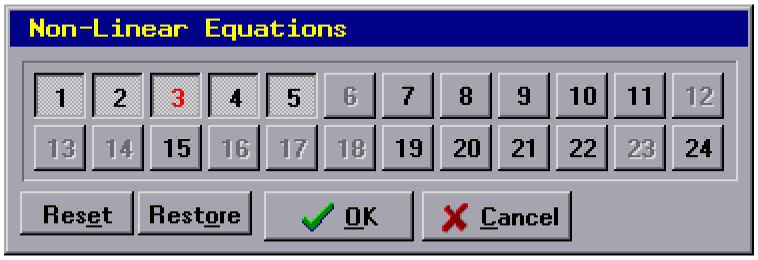
Nonlinear equations

Kalkulator will solve (or at least attempt to solve) a system of up to nine simultaneous equations of the form

f1(x1,x2..xn) = 0
f2(x1,x2..xn) = 0
....
fn(x1,x2..xn) = 0

with functions *f1, f2..f9* defined by *n* of the expression lines, and with the argument variables **v1**..**v9** used in place of *x1..x9*. (Non-registered copies handle up to three equations with arguments up to **v3**.)

First, the argument variables have to be set to some initial values (an educated guess). This is done with the **[Put]** button in the Argument Page of the Variable Panel. The total number of arguments has to be equal to the number of equations, and all arguments have to be used in the system, but a single equation does not have to use all.



Clicking on **[Non-Linear]** in the Numerical Page of the Application Panel will now bring up a dialog box with 24 (or six) buttons, corresponding to the Kalkulator expression lines. If a line is empty, its button will be disabled.

The first click on a button shows the corresponding expression in the Expression Line; the second click toggles the button state up or down. Buttons in the "down" state denote our choice of expressions defining *f1..fn.*

[Reset] brings all buttons to the "up" position, while **[Restore]** returns them to the original status, as just before the dialog was shown.

Clicking on **[OK]** starts the computation process. The solutions, if found, will be stored back in **v1**, **v2**... and displayed in the Argument Page.

The Newton algorithm is used to find the solutions. With no limitation on the form of the left hand side functions, there is no way to predict which of the possible multiple solutions will be reached from the given initial approximation. The system may have no solutions at all; even if it does, for a given starting point the algorithm may be unable to find any. Usually the better the initial guess is, the greater the chances of reaching a solution. You may try again from a different initial combination of **v1**, **v2**...

These limitations are not something specific to Kalkulator; these are just facts of life in numerical analysis.

See also:

Checking the quality of solution

Checking the solution quality for non-linear equations

At any moment when the argument variables **v1**..**v9** have some values assigned, you can evaluate any expression using them.

This is handy in checking the quality of solution to a non-linear equation system. Use the arrow keys to bring an expression into the Expression Line and hit the Enter key. The value of the expression will be shown in the Result Display. If you do it after having solved the system, the result (which in the ideal case should be equal to 0) will reflect the accuracy of the solution.

The Newton algorithm used in solution search stops when all equations give the result within 1E-12 from zero, so the displayed values should be within this range. This can be improved, if necessary, by clicking on the **[Non-Linear]** button again; this way you may bring the accuracy to the limits of machine precision.

Function optimization

A common problem is to find an extremum (minimum or maximum) of a function of multiple arguments, *f(x1,x2,...)*. To solve it with Kalkulator, enter the function *f* into the Expression Line, using the Argument Variables **v1**, **v2**... as *x1, x2*... (this limits us to functions of up to nine arguments; up-to three in non-registered copies).

You will have to set **v1**, **v2**... to some reasonable initial guess from which the search will begin.

Then clicking on **[Extremum]** in the Numerical Page of the Applications Panel and choosing between a minimum and a maximum from a popup menu will start the computation.

The solution, if found, will be stored back in **v1**, **v2**..., and the corresponding value of *f* will be shown in the Result Display.

During the computation pressing any key will suspend the operation, allowing you to resume or abort. In the latter case, you may decide to keep the last (i.e., best) result reached.

Obviously, with no limiting assumptions on the shape of *f*, the function may have one extremum of the desired kind, or multiple ones, or none at all. Therefore there can be no guarantee that Kalkulator will find the "right" extremum (or any extremum at all). The usual pitfalls of the procedure may be:

1. Finding a local extremum while a "better" one exists elewhere;
2. Running into the blue infinity;
3. Getting stuck (or fooled into assuming an extremum has been found) in a region where *f* has a very shallow slope, especially if it meanders around in the multidimensional space.

Usually these dangers become more imminent when the number of arguments increases. The best protection is to have a reasonably good initial approximation; some understanding of your function behavior. In case of doubt, restarting the algorithm from a different initial guess makes sense as well.

> These are not limitations of Kalkulator or of the downhill simplex algorithm it uses; these just the facts of life in the multidimensional jungle! For a good discussion of joys, dangers and techniques of extremum search, and a review of commonly used algorithms, refer to the Numerical Recipes, a book worth its weight in gold.

Argument variables

The symbols **v1**, **v2**, ... **v9** denote *argument variables*, used by Kalkulator in problems involving multi-argument functions: systems of non-linear equations, function optimization, data point and distribution fitting .

If such a function has *n* parameters, the expression describing it has to depend on the first *n* of these variables. For example, when searching for a minimum of a three-argument function, you have to use **v1**, **v2** and **v3** in your expression, not **v1**, **v2** and **v4**, etc.

The dependency can be direct (i.e., the argument variables are used explicitly in the expression) or indirect (used like inside user-defined functions to which the expression refers).

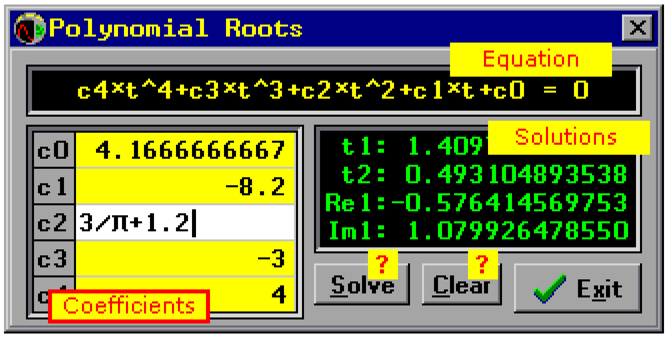
If too many of the argument variables are used, or too few, or if there are gaps in sequence, an appropriate error diagnostic will be shown when you try to execute the operation. (Checking the syntax by hitting Ctrl-Enter will not detect anything wrong here!)

The argument variables are displayed in the Argument Page of the Variable Panel; from here their values can also be accessed, modified, assigned or erased.

Before performing the numerical operations using these variables, you have to assign them some initial values - an "educated guess". The better this guess is, the greater the chances that the operation will find a solution, and the faster this solution will be find. The solution itself will be stored back in those variables.

The argument variables with assigned values can be used in Kalkulator expressions like the "regular" variables **A**..**Z**. This allows you to check the quality of the solution or to use the results in other calculations. There are, however, no corresponding buttons; the argument variables can be entered into expressions either by typing or by clicking on the values displayed in the Argument Page.

Polynomial equations



c4*t^4+c3*t^3+c2*t^2+c1*t+c0 = 0

c0	4.1666666667
c1	-8.2
c2	3/π+1.2
c3	-3
c4	4

```
t1:  1.409
t2: 0.493104893538
Re1:-0.576414569753
Im1: 1.079926478550
```

Selecting the **[Extras | Polynomial roots]** menu option brings up the Root Window, used to find real and complex roots of polynomial expressions of up to the fourth degree (up to the third degree in non-registered copies).

In other words, the program finds solutions of an equation

```
    c0 + c1*t + c2*t^2 + c3*t^3 + c4*t^4 = 0
```

where at least one of the real coefficients **c1**..**c4** has to be non-zero. The coefficients are entered into the yellow data grid at the left.

Upon clicking on **[Solve]**, the solutions are found analytically (i.e. without using any numerical approximations) and displayed in the panel at the right.

The real roots are denoted as **t1**..**t4**. If the equation has complex roots, their real and imaginary parts are shown as *Re1* and *Im1* (and *Re2* and *Im2*, if applicable). Each real/imaginary pair defines a pair of conjugate complex roots, e.g., *(Re1,-Im1)* and *(Re1,+Im1)*, sometimes also denoted as *Re1-i*Im1* and *Re1+i*Im1*.

Note: the **t1**..**t4** symbols used in this window for the roots do not have anything to do with the **t1**..**t3** parameters of user-defined functions. Their values can be stored in Kalkulator variables by clicking the mouse in the solution display.

See also:

Data Grids

The **[Clear]** button clears all fields
in the Root Window.

When the **[Solve]** button is clicked upon,
the roots of the current equation are computed
and displayed.

Polynomial equation

This field shows the form of the polynomial equation whose roots will be found by clicking on **[Solve]**.

Polynomial coefficients

The real coefficients **c0**..**c4** of a polynomial equation are entered here. Any legal Kalkulator expression is accepted.

Solution display

The values of computed roots (or their real and imaginary parts) are shown here.

Clicking the left mouse button allows to store the selected value in one of the Kalkulator variables; the right button is used to view the selected value with full accuracy.

Plotting functions

Plotting a function on an empty canvas

First, enter the function into the Expression Line, using t as the function argument (a "dummy parameter"). Then click on **[Graph]** in the Graph Panel. You will be asked for the x- and y-limits by the Graph Range dialog, from which you also can set one or both axes to a log scale.

The Graph Canvas will now show up, on which the drawing takes place, obscuring a large part of the Main Window. The drawing process can be interrupted by pressing any key.

Hiding and showing the Graph Canvas

Click on **[Hide]** (or press Enter) to hide the Graph Canvas without erasing the graph. Click on **[Show]** (the same button, renamed at that time) to display it again.

Adding a function plot to an existing graph

While the graph canvas is active, you may click on **[Add]** to add the function defined by the current Expression Line to the canvas. Obviously, this time you will not be asked for the range. You also can add a function plot to an existing histogram or scatter plot, but not vice versa. The last added plot can be removed from the canvas by clicking on the **[Undo]** button.

Coordinate readout

A left mouse click inside the canvas invokes a pop-up menu with coordinates of the selected point. The menu allows you to store the x- and/or y- values of the point in the variables **X** and/or **Y**.

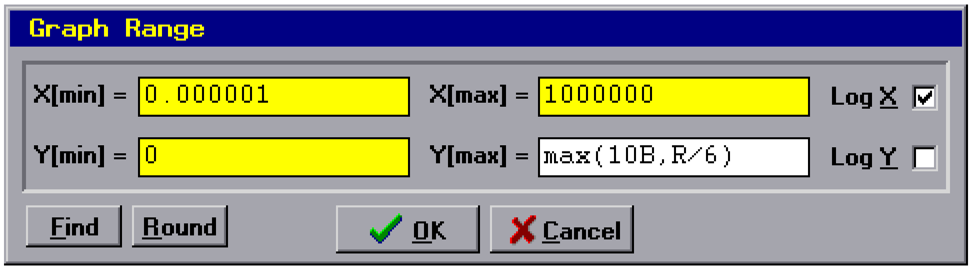
Copying canvas to clipboard

Pressing the F12 key while the Graph Canvas is up, copies the canvas to the Windows clipboard. From here it can be pasted into any application accepting the clipboard Paste command.

See also:

Multiple function plots
Graph range
Graph options

Graph range


Graph Range

X[min] = 0.000001 X[max] = 1000000 Log X ✓

Y[min] = 0 Y[max] = max(10B,R/6) Log Y ☐

The Graph Range dialog box is used to enter x- and y-limits for function plots and regression line graphs. Any valid arithmetic expressions can be entered.

Clicking on **[Find]** will estimate the y range for the x-limits previously entered, except for the case of regression graphs, where the x range will also be estimated. On a similar basis, the **[Round]** button will round the limits to "reasonable" values.

The buttons at the right set the x- and/or y-scale to logarithmic. Obviously, this requires that the corresponding range limits are positive.

Kalkulator imposes some limitations on the graph limits (x or y), otherwise an error message will be shown.

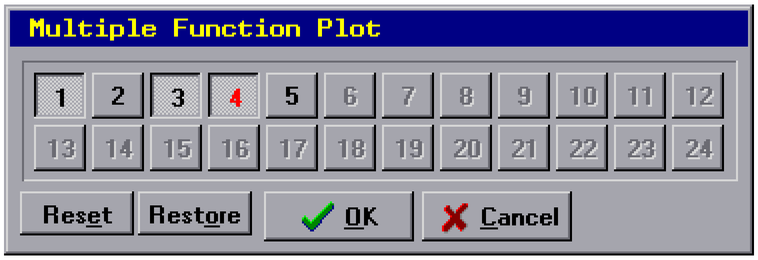
- In the linear scale the relative range width has to be not less than 1E-6. For example, while the range (0.999999,1.0) still fits within this limitation, the range 0.9999995,1.0) does not.
- In the logarithmic scale the range has to cover at least a factor of two; for example (4.0,8.0) but not (4.0,7.9).

See also:

Entering numerical values
Plotting functions
Polynomial regression

Multiple function plots

To graph a number of functions at once (and, more importantly, to apply the auto-range feature in the Graph Range dialog to all of them at one time), click on the **[Multi]** button in the Graph Panel when no graph is being shown. (Clicking on that button when the graph canvas is active will add a number of function plots to the existing canvas.)



This dialog box will allow you to select the expressions to graph. The first click on a line button will bring that line to the display (showing its number in red), and the second will actually include it into the plot or exclude it, what is shown by the button being down or up, respectively.

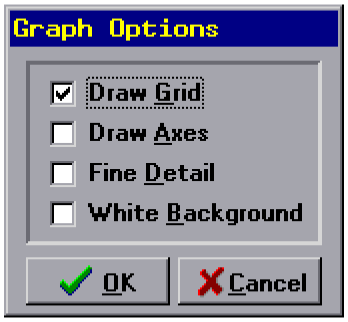
Instead of the mouse, you may use the cursor keys to move around the line number buttons, and the space key to select or deselect a line number.

[Reset] brings all buttons up, while **[Restore]** returns them to their original states.

Clicking on **[OK]** with at least one line selected will bring up the Graph Range Dialog, like in case of single plots. Now, however, the **[Find]** button will find the common y-range for all the selected expressions.

Note: in non-registered versions of Kalkulator you will be able to graph up to three expressions at a time; in the registered one - any and all of the twenty-four available.

Graph options


Graph Options

[✓] Draw Grid
[ ] Draw Axes
[ ] Fine Detail
[ ] White Background

The **[Options|Graph]** menu entry brings up a dialog box allowing you to set the following:

- **Grid** – with this option enabled, a reasonably spaced grid will be drawn in the background
- **Axes** – (yes, that's plural of "axis"!), solid lines will be drawn for x=0 and y=0
- **White Background** – on some monitors graphs may look better this way; this can also be useful when copying the a graph to the Windows clipboard for use in another program.
- **Fine Detail** – this tells Kalkulator to be more cautious adjusting the plotting step, so that discontinuities, sharp peaks and dips can be drawn more accurately, albeit more slowly.

You can always invent a function to fool any graphing program. Kalkulator, however, is more difficult to fool than most. It adjusts the plotting step to the local line curvature and ignores non-computable points (singularities), still trying to approach them as close as reasonably possible.

To compare Kalkulator against any other function graphing program you have, try plotting **sin(1/t)** for **t** from -1 to 1 with the Fine Detail setting at "on". Another good example is **int t**.

Statistical operations

A variety of statistical operations can be performed on values stored in the Data Buffer

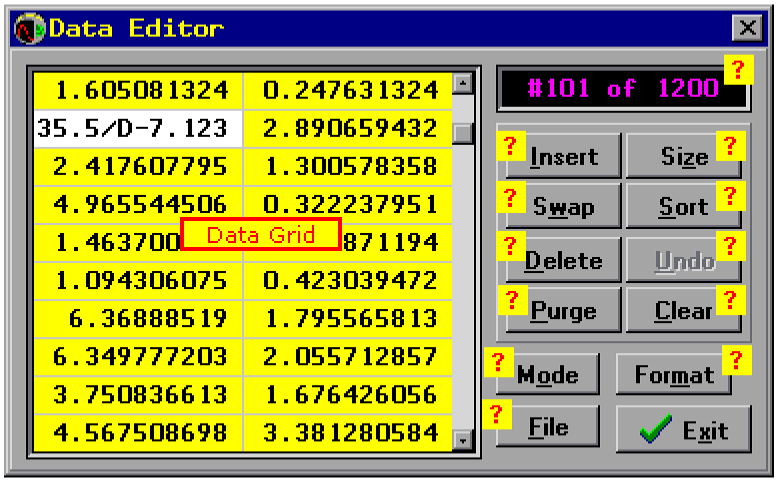
The buffer may be in one of two modes: X (single data values) or XY (value pairs). The current mode is shown as **[X]** or **[XY]** in an indicator in the Statistics Page of the Application Panel. Clicking on that indicator can be used to change the mode; the same can be done with the **[Mode]** button in the Data Editor. To open the Data Editor, click on the **[Edit]** button.

Topics:

Data Buffer

This is where Kalkulator stores the data points submitted to statistical operations. The capacity of the buffer is 128 single-value points (in the X mode) or 64 value pairs (XY mode); in the registered copies these limits are raised to 6400 and 3200, respectively.

Data Editor



1.605081324	0.247631324
35.5/D-7.123	2.890659432
2.417607795	1.300578358
4.965544506	0.322237951
1.463700	871194
1.094306075	0.423039472
6.36888519	1.795565813
6.349777203	2.055712857
3.750836613	1.676426056
4.567508698	3.381280584

From here you can manipulate the contents of the Data Buffer.

The grid at the left displays the data (X and Y columns shown in this picture). More about using it can be found in the Data Grids topic.

Click on the yellow stickers for information on particular elements of the Data Editor window.

Instead of entering the data manually, you can also use the current expression to compute the values of points in the X or Y column. This feature is accessed via the **[Eval]** button in the Statistics Page of the Operation Panel.
The data in the buffer is saved together with the program status when you exit Kalkulator, to be automatically read back when you run it the next time. The registered version also allows to save the data to a separate text (ASCII) file and to load it from there.

See also:

Computing data points

The **[Clear]** button clears the buffer.
The data is lost.

Clicking on **[Delete]** (or pressing the Delete key) will remove the selected data point from the buffer (in the XY mode both X and Y value will be deleted).

This button, accessible only in the registered copies of Kalkulator, allows to save the buffer contents to a plain text (ASCII) file or to read the buffer from such file.

This button changes the data display format between fixed-point and exponential. The third, "default" option uses the format depending on the displayed value; it also drops the trailing zeros from the display.

Clicking on the **[Insert]** button (or pressing the Insert key) will insert a new data point before the selected one.

This field shows the index of the selected data point, as well as the current number of points in the buffer.

The **[Mode]** button toggles between
X and XY modes

Clicking on **[Purge]** will delete all bad data points from the buffer.

The **[Size]** button will set the number of data points in the buffer to a new value (within the buffer maximum size). If the new number is smaller than the current one, the points at the end will be discarded. If it is larger, then new bad data points will be appended at the end.

Clicking on **[Sort]** will sort the data points
in the ascending sequence of x- or y-values.

The **[Swap]** button (available in the XY mode only) will swap the x- and y-values in the buffer.

Clicking on **[Undo]** (if it is enabled at the moment) will insert the last deleted data point back into the buffer.

Data Grid

You can view and edit the contents of the Data Buffer here. In the XY mode, the left column shows x-values, the right one – y-values. One of the data points will be always selected (highlighted).

Computing data points

The expression in the Expression Line can be used to compute new values for the points in the Data Buffer. To do it, use **[Eval]** in the Statistics Page of the Application Panel.

A pop-up menu will give you a choice between computing all x-values or only those tagged as bad data; in the XY mode there will be also these options for y-values.

Kalkulator will compute the expression value and assign it to every data point as chosen above.

All this would not have much use, if not for the fact that the expression defining the computed values may refer (via fetch functions **x#** and **y#**) to the existing data points, and to the index, denoted with the dummy variable **t**, of the point being computed. Thus, for example, the expression

```
y#t+.5x#t
```

used to compute all y values, will replace every y in the buffer with its previous value incremented by half of the value of the corresponding x. In other words, this is a full-fledged column arithmetic.

Another common use of this feature would be to replace all bad data values with a given constant, e.g., zero.

For some purposes (not only educational) it may come handy to use the random number generators in computing the data points in the buffer.

Bad data

When points in the Data Buffer are being computed, some of them may end up without values assigned, because of computation errors. These points will be flagged as bad data, denoted in the Data Editor as "?".

This is also how the new points, created when the number of points in the buffer is being changed to a larger value, are marked.

Population parameters

Clicking on **[Param]** in the Bottom Row will compute the parameters of the points in the Data Buffer:

- **nn** – the number of data points
- **mx**, **my** – mean values of x and y (if applicable)
- **sx**, **sy** – standard deviations
- **vx**, **vy** – variances
- **x1**, **x2**, **y1**, **y2** – the minimum and maximum values
- **cv** – the covariance (XY mode only)

Parameter values are shown in the Parameter Page of the Variable Panel. Fresh values are displayed in green; when the data in the buffer changes, the color will change to olive.

When you build a histogram of data, four more parameter variables will be defined and added to the Parameter Page:

- **nh** - number of histogram bins
- **h1**, **h2** - histogram limits
- **dh** - bin width

The parameters will not be automatically recomputed when the buffer contents changes; this has to be done explicitly again (unless you invoke one of the operations resulting in parameter recomputation, like histogram drawing or goodness of fit evaluation). This means that old parameter values can still be used after data points have been modified (for example, you can use them in computation of new data values).

Parameter symbols can be used in expressions like variables. Clicking on a parameter value in the scrolling list will enter its symbol into the Expression Line.

See also:

Parameter Page
Polynomial regression
Variables

cv

The covariance of x and y in the Data Buffer.

mx

Mean value of *x* in the Data Buffer.

my

Mean value of *y* in the Data Buffer.

nn

Number of points in the Data Buffer.

sx

Standard deviation of x in the Data Buffer.

sy

Standard deviation of y in the Data Buffer.

vx
Variance of x in the Data Buffer.

vy

Variance of y in the Data Buffer.

x1

The smallest value of x in the Data Buffer.

x2

The largest value of x in the Data Buffer.

y1

The smallest value of y in the Data Buffer.

y2

The largest value of y in the Data Buffer.

nh

Number of bins in the last histogram drawn or computed.

h1

The lower limit of the last drawn or computed histogram.

h2

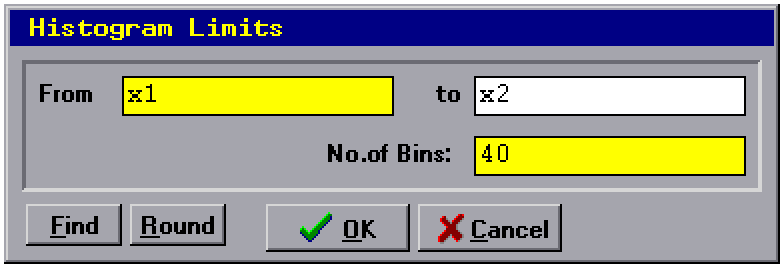
The upper limit of the last drawn or computed histogram.

dh

The bin width of the last drawn or computed histogram.

Drawing a histogram

Clicking on **[Hist]** in the Statistics Page of the Application Panel (in the X mode only; in the XY mode the button is labeled as **[Regr]**) will draw a histogram of the points in the Data Buffer.



From: x1 to: x2

No.of Bins: 40

A dialog box will ask for the histogram limits and the number of bins (maximum: 100). You can use the **[Find]** and **[Round]** buttons to let Kalkulator do the work here: **[Find]** will find the range of data points and **[Round]** will round the range limits outwards to some sensible "round" values.

The histogram parameters will be stored in parameter variables: **nh**, **h1**, **h2**, and **dh**, accessible from the Parameter Page of the Variable Panel. The statistical parameters of the population also will be recomputed in the process.

See also:

Entering numerical values
Goodness of fit
Data point and distribution fitting

Polynomial regression

The **[Regr]** button in the Statistics Page is used to compute polynomial regression coefficients for the XY data points in the Data Buffer, i.e. to find the best least-squares polynomial fit to the data.

You can choose the polynomial degree from a pop-up menu (up to 2 in the trial version, up to 9 in registered copies of Kalkulator). This choice will be limited to values lower than the number of data points in the buffer, i.e., in order to draw a line of n-th degree you will need at least n+1 data points. The menu has also an option to just draw the points, without computing the regression.

The computed regression coefficients will be shown in the Parameter Page of the Variable Panel. They are denoted as **a0...a9** (depending on the polynomial degree; in non-registered copies up to **a2** only) and can be used as components of expressions.

After the regression coefficients have been computed, you will be given an option to plot the points and the regression line on a cleared graph canvas or to add only the regression line to the existing graph. In the former case you will be asked for a plot range.(There is also a **[Cancel]** option which would abandon plotting.)

The function poly can be used at any time to compute polynomial values for the most recently computed coefficients (if any) and for a given argument value.

See also:

- Population parameters
- Parameter Page
- Variables
- Fetch functions
- Plotting functions
- Goodness of fit

a0…a9
Polynomial regression coefficients

Regression coefficients

Variables **a0**, **a1** ... **a9** (depending on the degree), computed for the XY data buffer with use of the **[Regr]** button.

Goodness of fit

Clicking on the **[Good]** button in the Statistics Page of the Application Panel will compute the statistics reflecting how well does the current expression line describe your points in the Data Buffer.

The value of the "goodness function" describes how well does the data fit our function; a smaller value means a better fit. Its definition depends on the current statistical mode.

In the X mode the goodness statistics shows how well is your point population fitted with a given distribution; in the XY mode - how well are the *(x,y)* points fitted with a given regression function *y* = *f(x)*.

When the computation is done, you will be offered an option to graph the data points and the fitting function.

See also:

Goodness of a distribution fit
Goodness of a regression fit
Data point and distribution fitting

Goodness of a distribution fit

In the X mode we are dealing with a population of data values of the same kind. The process will check how well does this population fit a distribution function defined by the current expression.

> It is assumed that the Expression Line contains a distribution density function (in case of discrete distributions: a probability function). Kalkulator makes a crude check of this assumption, checking whether the function normalizes to the total probability of 1; if not - you wil be given a warning which you can ignore (if you really know what you are doing) or not.

The pop-up menu showing up in response to the click on the **[Good]** button offers a choice between two measures of the goodness of distribution fit:

Chi-square

This measure uses the data binned into a histogram; the program will prompt you for the histogram limits and for the number of bins (these values will be stored in the parameter variables shown in the Parameter Page).

The chi-square statistics is then defined as

$$\sum (Ni - N \cdot Pi)^2/n$$

where N is the total number of data points, Ni - their number in the i-th bin, and Pi - the distribution probability over that bin. The sum runs over n bins (combined, if necessary, to provide $N \cdot Pi > 8$) and the bin probability is estimated with use of the trapezoid rule, with a step equal to the original bin half-width.

The bins with $N \cdot Pi < 8$ will be combined with their neighbors to the right (the righmost ones: to the left, obviously), until the expected number of points reaches 8; this is the common procedure in chi-square evaluation. If this happens, n will equal to the effective number of bins (i.e., that after joining) so that it can be used in the computation of the number of degrees of freedom.

To see the effective n, click on the **[?]** button in the Statistics Page right after the computation is completed.

> Note: Kalkulator does not know whether it is dealing with discrete or continuous variable, therefore in the former case it is recommended that (i) histogram bins are of unit width, centered around whole numbers, like in (0.5,1.5) etc., or (ii) bin width is significantly greater than 1.

Log likelihood

This goodness statistics is defined as logarithmic likelihood function per data point:

$$-\sum \ln f(xi)/N$$

where f is the distribution density (or probability) function and the sum runs over all N data points, xi.

Goodness of a regression fit

In the XY mode the measure of goodness describes how well are our *(x,y)* points fitted with a given function *y = f(x)*.

That function is defined by the current expression using the dummy variable **t** instead of *x*.

The pop-up menu gives us a choice between two statistics available. In both cases the sum runs over all data points.

Sum of squares

The measure of goodness of fit is defined here as sum of squares of y-deviations of the points from the *f(x)* curve, divided by the number of points:

$$\sum [\, y_i - f(x_i) \,]^2 / N$$

where the sum runs over all *N* data points *(xi,yi)*.

Sum of absolute values

Similar to the above, except that absolute values are used instead of squares. This statistics is less sensitive to the outliers.

$$\sum \mathrm{abs}(y_i - f(x_i))/N$$

Data point and distribution fitting

If a regression function or a distribution density function depends on a number of free parameters, we may try to fit our data with that function, i.e., to find such set of these parameters' values for which some measure of fit quality (sometimes referred to as goodness of fit) reaches the "best" value (usually a minimum). This process is often referred to as *multiparameter data fitting*.

Our regression or distribution density function can be written as *f(x,p1,p2...)* where *x* is the function argument, and *p1*, *p2* ... - adjustable parameters, and the problem boils down to minimizing the goodness of fit function with respect to *p1*..*pn*.

Kalkulator can fit a regression or distribution density function with up to nine free parameters (three in non-registered copies). To do it, use the following procedure:

- Enter the regression function (in the XY mode) or distribution density (X mode) into the Expression Line, using **t** as the function argument and **v1**, **v2**... as the adjustable parameters. These parameters have to be used in a contiguous way; e.g., **v3** can be used only if **v1** and **v2** are used as well.
- Enter the initial values (your best educated guess) of **v1**, **v2**... into the Argument Page of the Variable Panel.
- Click on the **[Fit]** button in the Statistics Page of the Applications Panel. a pop-up menu will present you with a choice of the goodness of fit measure:
  - For the regression fit (XY mode): sum of squared deviations or sum of absolute deviations (see: Goodness of a regression fit)
  - For the distribution fit (X mode): the chi-square or maximum likelihood (see: Goodness of a distribution fit).

When you make your selection, Kalkulator will start crunching the numbers (unless some errors have been detected). The solution (i.e. the "best" values of of **v1**, **v2**...) will be displayed in the Argument Page, and the value of the goodness function - in the Result Display.

Depending on the fitting function, number of parameters, and on the selected goodness measure, the process may take from a fraction of a second to a few minutes (or even hours, if your demands are not reasonable: for example, optimizing the maximum likelihood for 64000 data points; you should be using a Cray for such tasks!).

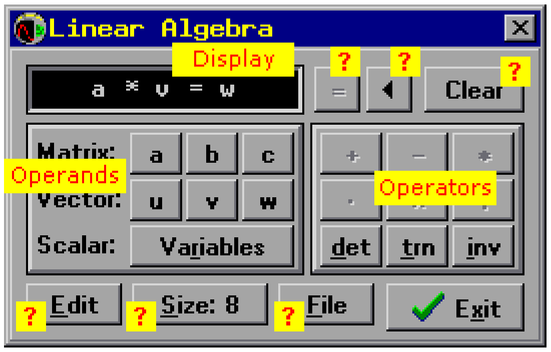
During the computation the intermediate best results will be updated in the Result Display and Argument Page and at any moment pressing a key will allow you to suspend the operation and to abort it if you choose so.

When the process is completed, you will be given an option to graph the data and the best fit.

Note: this problem really is a case of multi-dimensional function optimization. Therefore there is no warranty that the process will find the global minimum (or any minimum at all), and the outcome may depend on your initial guess of the argument values, especially when the number of adjustable parameters is greater than three or so. For more remarks on the joys and pitfalls of extremum search - see the chapter on function optimization.

As it is the case with all powerful tools (and this is one of them; try to find another program fitting your data with *any* function!), this feature works best in hands of an educated user. Kalkulator will do for you all necessary number-crunching, but not your thinking!

Linear algebra



To perform operations on vectors and matrices, open the Linear Algebra Window by clicking on the **[Extras | Linear algebra]** menu entry.

From here, enter one or two operands and an operator, then click on the **[=]** button (or press the "=" key) and select from a pop-up menu the object to store the result.

All buttons have keyboard equivalents; the only ones which may not be obvious, are: "x" for cross-product, "." for dot-product, "/" for linear equations systems and Escape to clear the display. Pressing Escape again when the display is clear, will exit the dialog.
Click on the yellow stickers for explanation of various elements of the Linear Algebra Window.

See also:

Vectors and matrices
Linear algebra operators
Vector and matrix editor

This button allows to change the size of vectors and matrices used in linear algebra operations (they will be cleared in the process, with all elements set to zero).

Press **[=]** to select vector, matrix or scalar to store the operation result.

Clicking on the **[File]** button allows to write all vectors and matrix objects to a text file, or to read them from such file. This feature is available only in registered copies of Kalkulator.

[Edit] activates the Vector and Matrix Editor, used to view or modify the values of vectors and matrices.

Press this button to back up one step.

Clicking on **[Clr]** will clear the displayed operation.

Display

The operation you are entering is shown here. After the result is computed, the display will become dimmed.

Linear algebra operator buttons

The buttons in this group are used to enter linear algebra operators. To avoid syntax errors, only the buttons corresponding to legal (in the current context) entries are enabled at any given moment.

Linear algebra operands

These buttons are used to enter operands of linear algebra operations: vectors, matrices, and scalars. The scalars are represented by Kalkulator variables, chosen from a pop-up menu.

Vectors and matrices

Linear algebra operations use, in addition to Kalkulator variables **A**..**Z** (scalars), six data objects: three vectors (**u**, **v** and **w**) and three matrices (**a**, **b** and **c**).

All matrices are square, and there is no visible distinction between row and column vectors. The character of a vector is always obvious from the context, e.g., in the **a×v** operation the vector **v** is understood to be a column – this multiplication does not make sense for **v** being a row vector.

The size of matrices and vectors (the same for all objects) can be set to anything between 2 and 4 (in registered copies: 16) by clicking on **[Size]** in the Linear Algebra Window. This operation will also set all vector and matrix elements to zero.

Individual elements of matrices and vectors can be used in Kalkulator expressions: their values are accessed with fetch functions **a#**, **b#**, **c#**, **u#**, **v#** and **w#**.

Linear algebra operators

These operators are accessible from the Linear Algebra Window. They can be divided into two groups. In the listings below M stands for any matrix (**a**, **b** or **c**), V for any vector (**u**, **v** or **w**) and S for any scalar (Kalkulator variable **A**..**Z**).

The first group, unary operators, accept a single operand in the prefix notation.

- **det** computes the determinant of a matrix:

  ```
  det M = S
  ```

- **trn** transposes a matrix, i.e. changes rows into columns and vice versa:

  ```
  trn M = M
  ```

- **inv** performs matrix inversion:

  ```
  inv M = M
  ```

The second group are binary operators, accepting two operands in the infix notation.

- **+** computes the sum of two objects of the same kind:

  ```
  M + M = M      V + V = V
  ```

- **–** does the subtraction in the same manner
- ***** finds the product of two operands (not necessarily of the same kind), according to the standard linear algebra rules. The following combinations are possible:

  ```
  M * M = M      M * V = V      V * M = V      V * V = M
  M * S = M      S * M = M      V * S = V      S * V = V
  ```

  In the case of M*V the vector is treated as a column (and the result is a column vector as well), while for V*M both the vector operand and the result are row vectors – these are the only combinations allowed by the rules of linear algebra.

  For the product of two vectors the first one is understood to be a column, and the second one, a row (an outer product).

- **·** is the inner product (the dot-product) of two vectors:

  ```
  V · V = S
  ```

  Note that u·u computes the square of the norm (length) of u.

- **×** is the cross-product of two vectors, defined only for vectors of size 3:

  ```
  V × V = V
  ```

- **÷** is the operator to solve systems of simultaneous linear equations. The operation

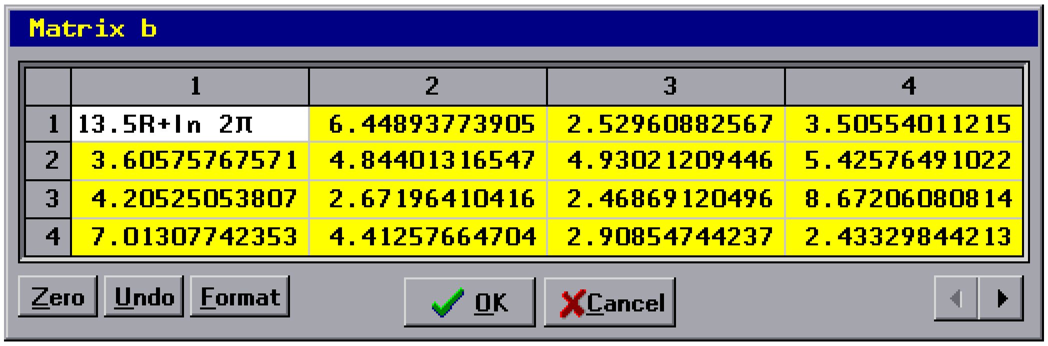
  ```
  M ÷ V = V
  ```

  finds the solution vector of a system defined by the matrix and vector arguments. For example, the operation **a÷v=u** will solve the system usually written down as **a*u=v** (now you see why Kalkulator uses here an operator looking somewhat like division).

There are no rules against storing the result of an operation in one of the operands: the operation **a*u=u** will be performed properly; the old elements of u will be overwritten only when no longer needed in the multiplication.

A matrix or a vector can also be assigned to another, as in **a=b** or **v=u**. Note that the destination object follows the "**=**" here.

Vector and matrix editor



	1	2	3	4
1	13.5R+ln 2π	6.44893773905	2.52960882567	3.50554011215
2	3.60575767571	4.84401316547	4.93021209446	5.42576491022
3	4.20525053807	2.67196410416	2.46869120496	8.67206080814
4	7.01307742353	4.41257664704	2.90854744237	2.43329844213

This editor (shown here for a 4*4 matrix) uses a data grid to enter values of vector or matrix elements.

- **[Zero]** sets all elements of the edited object to zero.
- **[Undo]** restores the object to the original state, as it was just before the editor was invoked.
- **[Format]** changes the display format between fixed-point and exponential. The third, "default" option uses the format depending on the displayed value; it also drops the trailing zeros from the display.
- The arrow buttons (visible only if needed) move to the left or to the right.
- **[OK]** exits the editor making the changes permanent.
- **[Cancel]** exits as well, undoing all the changes (the Escape key may be used to the same effect).

Computing matrix and vector elements

You can use a formula defined by the current expression to compute all elements of a selected matrix or vector. The expression may (and usually will) depend on the element index (or indices).

First, enter the expression into the Expression Line; for a vector use **t** as the element index; for a matrix, use **t1** for row and **t2** for column.

Next, click on **[Evaluate]** in the Linear Algebra Page of the Application Panel and select a matrix or a vector from the pop-up menu. Within a fraction of a second your matrix or vector will be filled with computed values.

For example, entering the expression **t1+.1t2** will generate a matrix like

```
1.1   1.2   1.3   ...
2.1   2.2   2.3   ...
...
```

while **ran 1** will fill a matrix or a vector with random values uniformly distributed over (0,1).

If an arithmetic error occurs during computation, the element(s) for which it happens will be marked as bad values, but the computation will continue until all others are computed.

Bad values

When matrix or vector elements are being computed, some of them may end up without values assigned, because of computation errors. These points will be flagged as bad values, denoted in the data grid as "?".

Data Grids

These grids are used to manipulate the elements of vectors and matrices, and the contents of the Data Buffer. They provide a uniform user interface to these tasks.

At any moment, one cell of a grid is selected (drawn highlighted). Pressing any alphanumeric key (or Enter) will start editing that cell.

Data grid cells accept not only numerical input: any legal Kalkulator expression can be entered.

When a cell is being edited, pressing Enter again (or moving to another cell with arrow keys, Tab or mouse) will confirm the changes. This will be prevented if the entry contains an error; an appropriate error alert will be displayed instead.

If Escape is pressed before the editing of a given cell is complete, the previous contents of the cell will be restored. Pressing Escape when no cell is being edited will exit the editor of which the data grid is a component.

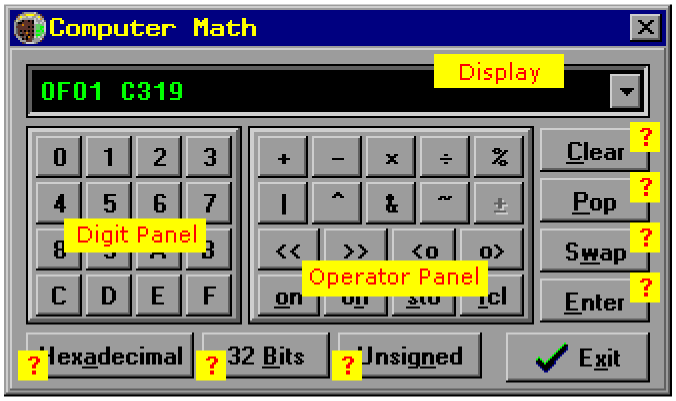
Moving around a grid is done with arrow keys (the left and right arrow will work this way only when no cell is being edited). The Tab key is also used to move on to the next cell. The Page Up and Page Down keys retain their customary meanings, and Ctrl-Home or Ctrl-End can be used to go to the first or to the last cell in the grid.

Although grid cells display numerical values with limited accuracy, the data remains stored with the full precision in which it was entered or computed. Even if you start editing a cell (e.g., by accident) the data will not be truncated to the displayed accuracy if you leave the cell without modifying its text. To see the full-length representation, click on a cell with the right mouse button

Computer math

The computer math includes arithmetic performed on integer numbers represented as computer words of finite bit length. These operations are grouped in the Computer Math Window which is accessible by clicking on the **[Extras | Computer math]** menu option.

The operations are entered with use of the Reverse Polish Notation, quite common in this kind of applications.



Click on the yellow stickers for more information about Computer Math Window components.

See also:

- Computer math operators
- Computer math mode settings
- Stack operations
- Computer math keyboard shortcuts

[Pop] removes the top entry from the stack, moving all others up by one place.

This button toggles between signed
and unsigned arithmetic modes.

[Swap] swaps the two top stack entries.

[Enter] pushes the contents of the Display onto the stack, moving all stack entries down by one place.

This button switches between various input/display modes (binary, decimal, etc.), reformatting the current Display and pushing its contents onto the stack.

This button is used to set the word length
(8, 16 or 32 bits).

The **[Clear]** button removes all entries from the stack and clears all memory registers.

Operator Panel

This panel groups buttons used to perform arithmetic operations on stack entries.

Digit Panel

Buttons in this panel are used to enter digits. Depending on the current mode (binary, octal etc.) some of them will be disabled.

Display

This is where the numbers you type appear. Any time the stack changes, its top element will also be shown here. If the Display is highlighted, it means that (i) its contents has been pushed on the stack and (ii) any new keyboard or mouse input will erase whatever is being displayed.

This button at the right shows the list of all stack elements. Clicking on one of them will push its copy onto the stack at the top.

Computer math operators

These operators are denoted mostly with symbols like those used in the C programming language and can be divided into three groups. The examples below are given in the hex mode.

Operators applied to two arguments from the stack

These operations remove the two arguments from the top of the stack and then push the result back there.

- **+**, **−**, *****, **÷** – addition, subtraction, integer division (integer)
- **%** – modulo (remainder from a division), e.g. **B,4%** evaluates as 3
- **|** – bitwise OR, e.g. **2,1|** evaluates as 3
- **^** – bitwise exclusive XOR, e.g. **2,3^** gives 2
- **&** – bitwise AND, e.g. **3,1&** results in 1

The results of division and modulo operations depend on the current sign mode.

Operators applied to a single stack argument

The single-argument operations replace the top of the stack (used as the argument) with the operation result.

- **~** – bitwise NOT, e.g. **0~** gives FF if the word length is 8 bits.
- **±** – negation: **2±** gives FE (simply -2 in the signed decimal mode); this operator is disabled in the unsigned mode; in the signed mode it is used to enter negative values.

Operators using one stack argument and one from a pop-up menu

These operations replace the top of the stack with the result.

- **<<** and **>>** – shift left or right by n bits, e.g. **7>>2** gives 1 (the two rightmost bits are lost)
- **<o** and **o>** – rotate left or right by n bits, e.g. **7o>2** gives C1 (the two rightmost bits wrap over to become leftmost)
- **on** and **off** – set or clear the n-th bit (0-th being the least significant one), e.g. **7 off 2** results in 3.

The Operator Panel includes also the **[sto]** and **[rcl]** buttons, used for manipulation of memory registers.

See also:

Reverse Polish Notation
Stack operations
Computer math keyboard shortcuts

Reverse Polish Notation

The Reverse Polish Notation is commonly used in computer applications. It may seem a little funny at first; for example, to add 17 and 32 you have to enter **17,32+**. Then, to divide the result of that operation by 4, just enter **4/**. (People using HP calculators find this a way of life and swear by it.)

In Kalkulator's Computer Math Window the **[Enter]** button is used to separate the arguments (instead of the comma used in the example above). More exactly, [Enter] evaluates the number typed into the Display and pushes it onto the stack. Pressing one of the operation buttons performs that operation on the top two stack entries (sometimes just a single top entry), removes the argument(s) from the stack, and pushes the result onto the stack.

For example, in order to compute *(32+7)*(12-9)/5* you may enter the following (the display and stack are shown in the decimal mode):

Input	Display	Stack
32 [Enter]	32	32
7 +	39	39
12 [Enter]	12	12,39
9 -	3	3, 39
*	117	117
5 /	23	23

The stack is eight entries deep, more than enough for most practical purposes.

Stack operations

The stack is just a pile of numeric values (entries) used to store arguments to arithmetic operations; the operation results are also stored there. It works on the "last-in, first-out" basis and can hold up to eight entries.

The Computer Math Window provides a number of buttons performing explicit stack operations:

- **[Enter]** – push the current display on the stack (after interpreting it as a numeric value in the current display/input mode and, if applicable, truncating it to the current word length). The new value is put on the top of the stack, while previous stack entries are moved down by one place (the last one may thus get lost). If the contents of Display is not a valid numeric value, an error will be reported.
- **[Pop]** – remove the top entry from the stack; all other entries are moved up by one place.
- **[Clear]** – remove all entries from the stack (clearing the display as well)
- **[Swap]** – exchange the top stack entry with the next one: e.g., the stack "3, 4, 5, 6" will become "4, 3, 5, 6".

The whole stack can be viewed as a drop-down list by clicking on the button right of the Display. Then, clicking on any entry in the list will push its copy on the stack.

The stack is also affected by the arithmetic operations. The two-argument ones will remove the arguments from the top of the stack and then push the result onto it, while the one-argument operations (or those accepting an extra value from a pop-up menu) will replace the top entry with the operation result.

See also:

Reverse Polish Notation
Computer Math operators
Memory registers

Memory registers

The Computer Math Window has ten memory registers which can be used for number storage. The registers are numbered with digits 0..9.

To store the top of the stack in a memory register, click on the **[sto]** button and select the register from a pop-up menu. A similar operation with the **[rcl]** button will recall the contents of the selected register and push it on the top of the stack.

Computer math mode settings

These settings affect the word length (important in case of result overflow or shift and rotation operations) and how the results are displayed and input interpreted.

Any time one of the settings is changed, Kalkulator applies the change to the bit patterns stored on the stack (but not in memory registers): the words may be truncated or left-padded with zeros and then reinterpreted in the new mode.

Sign mode

The sign mode affects how the bit patterns are interpreted, how some of the arithmetic operations are performed and also how the decimal numbers are displayed. It has two settings:

- **Unsigned**: numbers are treated as positive only 0..255 (in 8-bit length), 0..65535 (16-bit) and 0..4294967295 (32-bit)
- **Signed**: the ranges are –128..+127 (8-bit), –32768..32767 (16-bit) and –2147483648..+2147483648 (32-bit)

The division **[/]** and remainder **[%]** operations will return negative results in the signed mode when their arguments are of different signs.

The **[±]** operator is disabled in the unsigned mode.

Display/Input mode

- **Binary** – base 2, valid digits are 0 and 1
- **Octal** – base 8, valid digits are 0..7
- **Hexadecimal** – base 16, valid digits are 0..F
- **Decimal** – base 10, valid digits are 0..9

For example, if the word length is 8 bits, the hexadecimal "FF" is octal "377", binary "1111 1111", decimal "255" or "-1", depending on the sign mode.

Word length

Three word lengths can be set: 8, 16 and 32 bits (if you start hollering, I will consider adding 64-bit words in one of the future updates). When the world length is changed, all stack entries will be affected: padded with zero bits or truncated.

Obviously, operation results exceeding the currently set word length will be always truncated from the left (and so will be the input). For example, using the decimal notation, "10000, 10 *" will give 100000 in the 32-bit mode, 34364 in 16 bits or 160 in 8 bits.

Computer math keyboard shortcuts

All Computer Math Window buttons have keyboard equivalents. The ones which are not obvious are:

- **Enter** – push the Display contents onto the stack
- **Down** or **Alt-Down** – show the stack contents in a drop-down list
- **Escape** or **Alt-Del** – clear the Display, storing its old contents in the copy buffer (pressing Escape when the Display is empty will paste into it the current contents of the buffer).
- **Alt-Ins** – swap the copy buffer and the contents of Display
- **<** and **>** – left and right shift operators (with a pop-up menu as above)
- **[** and **]** – left and right rotation operators (a pop-up menu will ask for the auxiliary argument)
- **Alt-Minus** – the change-sign operation, i.e. the **[±]** button
- **F1** – invoke the on-line Help
- **F12** – copy the Display into the Windows clipboard, so that it can be pasted into any Windows application supporting clipboard operations

Menu operations

These operations are accessible from the Menu Bar and from the Main Pop-up Menu.

File

- **Load user functions** – loads user functions from a disk file; all currently defined ones will be cleared in the process
- **Save user functions** – saves user functions to a disk file
- **Clear** – clears everything: expressions, variables, user defined functions, data buffer, graph canvas.
- **Exit** – exits the program. There will be a warning and an option to save the program status to a disk file. The status is automatically read next time you run the program, so that you can continue the previous work.

The registered version has two more entries here:

- **Read status** – loads the Kalkulator status (settings, expressions, user functions, variables, vectors and matrices, data buffer, graph canvas) from a **.kal** file, created on the last exit from the program or by explicit use of **[File | Save]**.
- **Write status** – saves the Kalkulator status (see above) to a **kalkul.kal** file.

In non-registered copies, this file is saved on exit and read when the program is executed again, but you cannot have multiple ***.kal** files.

Extras

This menu provides access to windows performing some specialized operations. Some of these windows are also accessible from pages of the Application Panel.

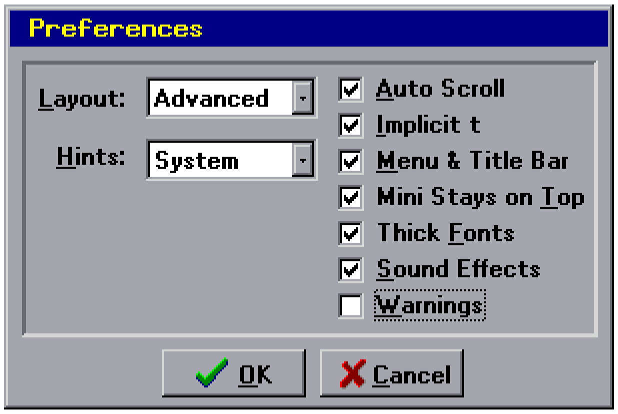
- **Linear algebra** – brings up the Linear Algebra Window
- **Polynomial roots** – invokes the Root Window to solve polynomial equations
- **Computer math** – shows the Computer Math Window

Options

- **Preferences** – shows the dialog to customize some aspects of Kalkulator's behavior
- **Graph options** – displays the dialog to change some graph settings
- **Reseed generators** – allows to reseed the random number generators
- **Show last time** – displays the elapsed time of the last timed operation (e.g., numerical integration), in case you are curious about it

There is also the usual Help menu.

Customizing Kalkulator


Preferences

Layout: Advanced

Hints: System

☑ Auto Scroll
☑ Implicit t
☑ Menu & Title Bar
☑ Mini Stays on Top
☑ Thick Fonts
☑ Sound Effects
☐ Warnings

The appearance and behavior of the Main Window can be adjusted by clicking on the Preferences entry in the Options menu. A dialog box will present you with a number of choices.

All these settings are saved in the **kalkul.kal** file when you exit the program, and loaded back when you run it again – there is no need to customize Kalkulator every time you use it.

- **Layout** - changes the Main Window layout between **Advanced** (the whole Main Window shown), **Basic** (no Application Panel) or **Minimal** (just the Expression Line, Result Display and the buttons next to it).
- **Hints** - choice between **System** (Windows standard) or **None**.
- **Auto Scroll** – toggles the autoscroll option.
- **Implicit t** – enables automatic insertion of the Temporary Variable, **t**, at the beginning of an expression
- **Menu & Title Bar** – removes or restores the Menu Bar and the Title Bar.
- **Mini Stays on Top** – when this box is checked, Kalkulator stays on top of all other windows when in the Minimal layout.
- **Thick Fonts** – toggles between thin and bold fonts in monospaced displays. (If your monitor is not very sharp, then you will probably prefer bold fonts. On the other hand, thin fonts look great on active-matrix displays.)
- **Sound Effects** – toggles sound effects on and off.
- **Warnings** – disables or enables the program warnings. With warnings disabled, the program status file will be always saved upon exit.

16- and 32-bit Kalkulator

Kalkulator exists in two variants:

- 16-bit, developed for **Windows 3.1** (3.11, etc.)
- 32-bit, running only on **Windows 95** and **NT** (in native 32-bit code)

Version 2.0 is the last major upgrade to Kalkulator/16. I am planning to provide maintenance updates (mostly bug fixes, if any), but not much more, for a year or so - but this is it.

The 16-bit support tales lots of my time which I would rather spend adding new features to the program; it is also becoming more cumbersome, as the 16- and 32-bit development tools are diverging more with every new release. Last but not least, it holds me from using some techniques available only in 32-bit environment.

Registered users of Kalkulator/16 can switch to Kalkulator/32 at no cost: just download the 32-bit version and it will recognize your original registration file, **kalkul.key**. Compare it to upgrade policies for all other programs you have used.

Known problems

Problems with the "old" Windows 95 System Agent

Microsoft admitted (in a quite understated way) that the System Agent, included with the Win95 Plus, messes up some floating-point calculations. The problem has been fixed in one of the Windows 95 Service Packs.

The symptoms are simple: the floating point accuracy drops from 19 to 16 decimal digits. Most applications will never notice this, as they are usually limited to the 16-digit precision of the C/C++ **double** type.

I was able to check this out – indeed, this is the case. My recommendation is to turn the System Agent off when using Kalkulator (and who needs this thing anyway?), or to get the fixed **sage.dll** file from the Microsoft Web page.

More floating-point problems in Windows

This is not the end of it. Exactly the same problem may sometimes arise *without* the System Agent running. It seems to depend on the programs runnng in the background (compression, virus protection, installation watch, etc.) By reshuffling these programs I was able to alleviate the problem on some installations. Looks like some other system DLL messes things up again - and I won't help it.

Kalkulator work-around

Kalkulator runs a simple test for the accuracy loss upon startup. If the problem is detected, the display precision setting will be limited to 16 (instead of 18) digits. You can check it by clicking on the [Format] button and then on "Fixed Point" or "Exponential" in the pop-up menu.

The precision loss is also accounted for in some Kalkulator operations (fractions, graph boundary rounding).

To check for yourself, type in (twenty 3's after the decimal point):

```
1/3-.33333333333333333333
```

and hit Enter (Kalkulator will reformat the expression, truncating the decimal fraction from 20 to 18 digits). The displayed result (use the exponential format!) should be about 2E-20, but on malfunctioning systems it will be about 5E-17.

Pentium bug

The infamous bug in some of the early Pentium processors is related to some division operations. Kalkulator detects a faulty processor and uses additional code to work around the problem. (Frankly speaking, it was the good people of Borland who took care of this: the Delphi compiler, used to develop this program, does it automatically.)

Frequently Asked Questions

If you have a question, ask. There are no stupid questions; only the answers can be.

Running from CD-ROM

Yes, you can, as long as the files are not packed (zipped). You will, however, have to live with two limitations:

- You cannot save any files (Kalkulator will write only to its own directory, nowhere else)
- Trial license expiration notice will be always displayed, as if you have been using the program for more than the trial period.

And yes Kalkulator is one of the last non-trivial Windows programs you can run from a floppy, too.

Regression degree

To do polynomial regression of the n-th degree you need at least n+1 points in the data buffer.

How do I uninstall Kalkulator?

Just delete the Kalkulator directory with all the files in it. The program does not put any files anywhere else.

How do I change colors?

You don't. The Kalkulator window colors are not changed from within the program; they are bound to your desktop colors, which can be changed from the Windows Control Panel.

The graph colors, on the other hand, are hardwired to some preset values. If there is a need, I may provide a color-setting option in one of the next releases. Let me know how important you think it is.

The button fonts look bad on my screen. What's wrong?

This means that the Kalkulator fonts have not been loaded.
The program should have complained about this with an error message.
Make sure that the files **kalkuld.fon**, **kalkulm.fon**, **kalkulp.fon** and **kalkult.fon** are in the Kalkulator home directory.

If no error message about missing **.fon** files has been shown, then obviously **Windows 3.1** is running out of resources.
Close as many applications as you can, then restart Kalkulator.
This problem should not occur under Windows 95.

Help fonts

Your Windows may be using a non-US character set. While the basic characters (a..z, A..Z, 0..9) are the same in all Latin-derived character sets, the language-specific sets differ in what characters are used for ASCII values above 127.

The affected characters are those entered with special keystroke combinations.

The program itself is not affected by this problem, as Kalkulator (as opposed to the Windows help engine) uses its own fonts, supplied with the program.

Multiple program instances

For the 16-bit version of Kalkulator: no. When you try to run another instance while the program is already running, this will be detected and the attempt will be just ignored.

For the 32-bit version: yes, as many instances as you wish (it is quite difficult to run out of system resources under Windows 95, so why not?).

What are you planning for the next version?

The next major upgrade of Kalkulator (expected in Summer '98) will include systems of ordinary differential equations (Runge-Kutta method). Don't worry: if you don't need this feature, it will be hidden behind a single small button...

In the meantime Kalkulator may see some updates, with minor improvements and bug fixes (if any bugs show up, that is).

Sound effects do not work?

Your computer may be lacking a sound card – this is the case if other Windows programs are also mute (although some programs may use the built-in speaker to make feeble beeps).

A simple way to emulate a sound card is to install a software driver **speaker.drv** from Microsoft. It was written for Windows 3.1, but works just fine with Windows 95 as well and can be downloaded for free. I'm using it on one of my notebooks without any problems.)

New in this version

This page contains links to topics which are new or significantly changed in the recent Help updates. (This is not a complete update history; it does not include bug fixes and improvements not affecting the use of the program; see the file **whatsnew.txt**.)

New in Version 2.1

Just three months after the major upgrade to Version 2.0, two very powerful operations have been added, hidden inobtrusively behind just two new buttons in the Applications Panel:

- Function optimization: extremum search for multi-argument functions.
- Fitting a distribution or a set of XY data points with an arbitrary function with respect of up to 9 free parameters.

Minor changes:

- The maximum data buffer size (32-bit registered version only!) has been increased from 6400 to 64000 points (32000 points in the XY mode).
- The goodness of fit function is now normalized to the number of data points (XY mode, X mode for maximum likelihood) or to the effective number of histogram bins (X mode for chi-square).
- The argument variables have been renamed from **t1**..**t9** to **v1**..**v9**.
- The **[Zap]** buttons now clear all variables in the displayed page.

Credits and copyrights

Kalkulator is copyright © 1988-98 by J. Andrzej Wrotniak. All rights reserved.

This software has been developed using Borland's Delphi and contains parts of code copyright © 1983-97 by Borland International.

The tabbed notebook widget used in Kalkulator is a part of the Classic Component Set copyright © 1996 by Classic Software from Australia. The program also uses the Applicata component by Ginteras Pikelis from Lithuania.

Most of the sound files used in Kalkulator have been created by Ted Tatman (thanks!) and are in public domain.

Kalkulator inherited many design ideas and a significant amount of code from two of my previous projects:

- from El_Cal, the Elementary Calculator for the Atari ST series (1988-94) – general concept, large parts of the user interface design, lots of well-tested numerical code
- from Ex, a Scientific Calculator for the HP LX Palmtop computers (1992-96) – expression parser, most of the underlying math engine

Some of the numerical algorithms used here have been adapted from the Numerical Recipes (extremum search, the Newton method, derivatives computation, Euler functions).

Important: read the enclosed disclaimer before using the program!

Disclaimer

This is what my lawyer told me to put here in a very small print:

Kalkulator is licensed on the "as is" basis.

Although the author has extensively tested the software and reviewed the documentation, he makes no warrant or representation, explicit or implied, with respect to this package, its quality, performance, merchantability or fitness for a particular purpose, in any hardware or software environment.

In no event shall the author be liable for direct, indirect, special, incidental or consequential damages arising out of use, misuse or inability to use the software or documentation.

In addition, drinking excessive amounts of coffee when using this program may lead to insomnia, headaches and loss of hair.

Pronounced "An-Jay". One of the most misspelled and mispronounced names in the United States.

Ex-physicist, formerly specializing in computer
Monte Carlo techniques applied to cosmic rays.
Taught physics, numerical methods and computer
science at University of Lodz, Poland.

In the U.S. since 1984, makes his life in software engineering.
Currently the Chief Scientist at Aerospace Engineering &
Research in Maryland, helping to bring the air traffic control
software out of its 19th century state. One day may get a
real job and a haircut.

Ex

The current version 1.51 (includes linear algebra) is available for $30. Works on HP 95/100/200 LX.

Evaluation copy can be downloaded from the CompuServe (go HPHAND, Lib.7) or from my personal Web page. One of the top CIS Palmtop Forum downloads.

Registration and support

Kalkulator is distributed as shareware: you are granted a trial license for a period of 30 days to see whether you will find the program useful and worth the price I'm asking. After that you have the options of either registering (purchasing) Kalkulator, or removing it from your system.

If you decide to keep Kalkulator, registering will bring you additional benefits: enhanced features, free updates and technical support. Last but not least, the program will get rid of the "nag screen" which starts showing up after one month expires.

Technical support is offered not only to registered users, but also (to a reasonable extent) during the trial period. If you have questions, problems or suggestions, feel free to send me email or write to my postal address.

And, most importantly, let me know what you like and dislike about the program: what new features would you like to see, what do you find inconvenient, confusing or mathematically incorrect (nah!). Your feedback is most valuable in making Kalkulator the program I want it to be.

Last but not least – if you like the program, tell your friends about Kalkulator and give them a copy (unregistered, of course, i.e. without your key file). You will do everyone a favor. Thanks.

See also:

How to register
Passing copies on to others
How to get the newest version
Vendor Information

My email address

CompuServe: **70611,2552**

Internet: **pan_andrzej@compuserve.com**

My Web page

My personal Web pages start at

http://www.freeflight.com/wrotniak/

The Kalkulator distribution and support page is

http://www.freeflight.com/wrotniak/kalkul.html

My address

Dr. J. Andrzej Wrotniak
2057 Happy Lane
Crofton, MD 21114-1917
USA

Passing copies on to others

You are allowed (and encouraged!) to pass unregistered copies of Kalkulator along as long as you do not alter the original distribution files and do not charge money for the program.

Distributing of the original .ZIP archive (or a floppy with its freshly unpacked contents) is recommended.

Your personal registration file, **kalkul.key**, is not redistributable.

The registered version

Really, there is no separate "registered version". Any copy of Kalkulator will become "registered" when it finds in its directory a small registration key file, **kalkul.key**. This file contains the license owner name (displayed on the program startup) and the proper verification information. It will stop working if modified in any way.

After registration, some of the program features will become enhanced:

- An arithmetic expression can be up to 120 characters or 72 tokens long (up from 60 or 36, respectively)
- You can switch between 24 expression lines, not just six
- The maximum size of vectors and matrices in linear algebra operations is 16 (as opposed to 4)
- Systems of up to nine (not three) non-linear equations are allowed
- The polynomial regression goes up to the ninth degree, not to the second
- The maximum number of arguments (free parameters) in function optimization, or in data point and distribution fitting, is 9 (not three)
- Up to 24 (not just three) simultaneous function plots are possible
- Polynomial roots up to the fourth (not just the third) degree are computed
- The data buffer stores 64000 or 32000 data points (X or XY mode), not 128 or 64 (this is true in the 32-bit version; under the "old" Windows 3.1 the limits are 6400 and 3200)
- Multiple status, or **.kal**, files can be used
- User-defined functions can be saved to, and read from, a disk file
- The data buffer and linear algebra objects can be written to an ASCII file or read from one; this allows to exchange the data with other programs
- You can reseed the random number generators

The registration remainder ("nag screen") will, of course, no longer show up.

See also:

How to register

Nag screen

Kalkulator will not stop working after the trial license expires. It will just display, at the program startup, the screen with licensing information, urging the reluctant user to make the choice between registering the program or removing it from his/her system.

kalkul.key

This is your personal registration key file; it will be recognized by Kalkulator, activating its enhanced features.

Copy this file to the same directory where Kalkulator is; also be sure to make a safe backup of it.

Whenever you get (download, steal, borrow) a new version of Kalkulator, remember to put **kalkul.key** together with the other files.
The key file currently distributed will work with all future versions of Kalkulator, 16- or 32-bit.

How to register

You may spend your money on Kalkulator (now that you've tried it and you know whether it is worth the expense) in one of the following ways, CompuServe and Web orders being the fastest; direct by mail the cheapest. Click on the green header for details:

Direct registration by mail

I will be glad to receive your direct registration by mail, accompanied with a check or money order.

Registration on the World Wide Web

You can use this method to pay with a credit card. The transaction is handled for me by a long-time, respectable shareware distributor from Texas, **Public Software Library** (PsL).

Registering by phone or fax

PsL also handles Kalkulator credit card orders (Visa, Master Card, American Express) submitted by phone or fax.

Do not send me your credit card number; I cannot forward it to PsL. To stay in line with the bank requirements, PsL has to receive your order directly from you.

Registration on the CompuServe

This option can be used only by CompuServe members. You should get your key within 24 hours.

Regardless of which of the above methods you are using, please include the following details with your registration:

- Your shipping address, if you would like to have a diskette mailed (**please state clearly if this is the case**, and then also mention which Windows you are running).
- Your email address, if applicable, in which case I will just send you the registration key file, **kalkul.key** and/or the manual entry code via email – this is the fastest (and cheapest, for me at least!) way.

Your registration covers both the 16-bit (Windows 3.1) and 32-bit (Windows 95) mutations of Kalkulator. For corporate and institutional users site licenses are available.

See also:

Registered version: enhanced features

Direct orders by mail

To order Kalkulator directly, send a check or a money order for $40 (U.S. currency, drawn on a U.S. bank) to my mail address:

> **Dr. J. Andrzej Wrotniak**
> **2057 Happy Lane**
> **Crofton, MD 21114-1917**
> **USA**

Include your name and address, as well as the email address, if applicable. I will use it to send you the Kalkulator registration key.

Ordering Kalkulator directly, you may use a 50% discount on any or all of my other shareware programs:

- The Midget calculator for just an extra $5
- Spheric (a spherical geometry calculator) for an extra $15
- Mr.Matt (an addictive little game) for an extra $6

Evaluation copies are available on my Web page.

Note: Usually your registration is processed on the day I receive it, but from time to time I may be traveling overseas for a week or two. While I can process electronic registrations from almost any place in the world, these received by regular mail would have to wait until I'm back. In such case I **always** post a notice under "What's new" on my Web page; please check.

Registration on the CompuServe

Members of the CompuServe can register Kalkulator on-line with use of the Shareware Registration Forum (GO SWREG).

The registration numbers are

- For the 16-bit version: #8743, file name – **kalkul.zip**
- For the 32-bit version: #10794, file **kalk32.zip**

The registration fee of $47.06 will be included into your next CompuServe bill.

Registering either version of Kalkulator covers both; the distinction is purely for bookkeeping purposes.

Please note that none of the discounts mentioned elsewhere in this document are available when registering this way.

Ordering on the Web

Set your browser to

http://www.freeflight.com/wrotniak/kalkul.html

and from there, click on the PsL registration button. This will take you to the **PsL Electronic Registration Service**, directly to the Kalkulator secure order form. Then fill in the order blank and follow the directions.

The all-inclusive registration fee when using this service is $46.88.

Your registration key will be usually sent to you within 24-36 hours.

As PsL handles only the monetary aspect of the registration, do not try to contact them for technical support, product availability or order status information. For these purposes use my email address.

Registrations by phone or fax

The registration price in this case is $46.88, all-inclusive.

Ordering by phone

Phone orders are accepted by PsL from Monday to Thursday from 7 a.m. to 6 P.M., Friday from 7 a.m. to 12:30 P.M. Central Time (London minus 6 hrs). The voice numbers are

800-2424-775 and **713-524-6394**

Order by the product name and number: **Kalkulator, #14667**.

Note: these phone numbers are for credit card orders only!

Do NOT use these numbers to get technical support or to check the status of your order (assuming such a need arises), or availability of the newest version. PsL handles only the monetary side of the transaction, passing your order to me within 24 hours; from that moment I'm your only point of contact. They are not prepared to handle any other questions or problems.

Ordering by fax

You can fax your registration to PsL:

713-524-6398

Include your shipping address, billing address (if different), card number, expiration date, phone number and signature. Order by the product name and number: **Kalkulator, #14667**.

Site licenses

Site licenses are priced at $100 plus $15 per user (e.g., for 20 users $100 + 20 * $15 = $400).

A site license covers all updates and upgrades for a period of one year; after that time new versions can be licensed at 50% off.

Updates and upgrades

For registered users of Kalkulator, updates and upgrades are free for the lifetime of the program, as long as you can download the current version from my Web page, CompuServe or another source.

After you download the new version , the program will recognize your registration, provided that your registration key file, **kalkul.key**, is present in the Kalkulator directory (make sure to keep a safe backup of the key file!).

> Users who registered Version 1.40 or earlier: do not worry, your registration key will still work with Versions 2.0 and above. The original upgrade policy has been changed and (surprise!) you do **not** need to pay a penny for a new version!

How to get the newest version

The newest version of Kalkulator can be downloaded at no cost from one of the following sources:

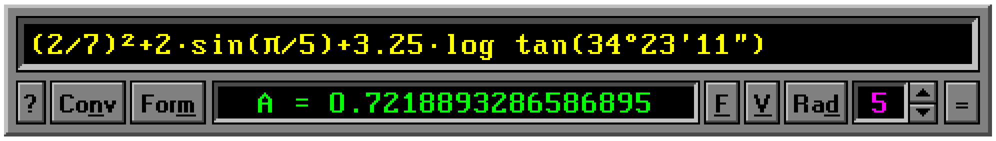
- The Kalkulator page on the **World Wide Web**:

  http://www.freeflight.com/wrotniak/kalkul.html

  Updates show up here before they can be found anywhere else.
- **CompuServe**: the Science Forum (GO SCIMATH), Library 5 (Mathematics), file **kalkul.zip** (16-bit) or **kalk32.zip** (32-bit).

Midget

For those who do not need the most advanced numerical features of Kalkulator, but still would like to use a smart and capable calculator accessory, there is a simpler alternative: the Midget.


```
(2/7)²+2·sin(π/5)+3.25·log tan(34°23'11")
A = 0.7218893286586895
```

Midget may be used as a handy replacement for the calculator included with Windows (3.1, 95 or NT).

What Midget does is to evaluate expressions (using the same powerful internal engine and providing the same accuracy as Kalkulator). It also has built-in unit conversion. Using it is a no-brainer. Just type in what you want to compute and hit Enter.

Midget is a $10 shareware. You can download an evaluation copy from the CompuServe (the Science forum, Math Library) or from my Web page.

Vendor information

Non-registered copies of Kalkulator can be distributed by BBS operators, on-line services, disk and CD-ROM vendors after obtaining my explicit permission.

This permission is not required for vendors and BBS operators approved by the Association of Shareware Professionals. In these cases contacting me, although not required, is still recommended, as I may be able to provide the most recent version of the program.

In any case it is required that

- All files in the original distribution archive are included without alteration, as listed in the Packing List section of the accompanying **readme.1st** file.
- The recipient of the program is informed in advance that the distribution fee paid to the vendor is not equivalent to purchasing the program: a registration is required if the recipient continues using Kalkulator after the expiration of the trial period.

See also:

My address

My email address