

---

# User Manual for EQW: A Graphical Equation Writer for the TI-89 and TI-92+

---

**EQW Version 1.17**

**Program by E.W.**

User manual by Doug Burkett, Bhuvanesh Bhatt and Eddie Shore

equationwriter@yahoo.com

Created 26 Jan 2001

Revised 29 Oct 2001

Beta tested by Bhuvanesh Bhatt, Andrew Cacovean, Phillip Hendrickson and Eddie Shore

EQW program © 2001 E.W.

User Manual © 2001 Doug Burkett, Bhuvanesh Bhatt and Eddie Shore

$$\frac{\int_0^8 \left( \frac{a \sin(t) \cdot b \cos(t)}{\ln(\tan(b^t) \cdot \tan(t^b))} \right) dt + \sum_{k=1}^{\infty} \left( \frac{\sin(k)}{k} \right)}{\sqrt{\frac{a \cdot x^3 + b \cdot x^2 + c \cdot x + d}{\lim_{n \rightarrow \infty} \left( \frac{2 \cdot n + 1}{\cosh(n\pi)} \right)}}} \cdot \prod_{t=\theta}^{n \cdot \theta} \frac{d^2}{dt^2} \left( \frac{3}{2t} \right)$$

MAIN      RAD AUTO      FUNC 7/20

*Try entering this expression without EQW and without errors!*

All copyrights and trademarks are the property of their respective owners.

## Contents

<b>Introduction</b> .....	4
<b>Revisions</b> .....	6
<b>Installing and uninstalling EQW</b> .....	7
Installing EQW .....	7
Uninstalling EQW .....	9
Crashes and recovery .....	9
<b>Starting and exiting EQW</b> .....	10
Starting EQW .....	10
Exiting EQW .....	12
<b>Basic operations with EQW</b> .....	13
Summary of EQW keys and functions .....	15
Moving the cursor .....	17
Deleting expressions and characters .....	21
Using the built-in menus .....	24
Using cut, copy and paste .....	25
Using the UNDO feature .....	26
Scrolling large expressions in the display .....	28
Entering program and function names .....	29
Evaluating expressions with [F1] and [DIAMOND] [ENTER] .....	32
Applying functions with [F2] - [F7] .....	34
Using forms .....	35
Other features and tips .....	37
<b>Calling EQW from TI Basic programs</b> .....	38
<b>EQW Extensions</b> .....	41
E.W.'s sample <i>eqwuser()</i> extension .....	41
Using [DIAMOND] [F1] ... [DIAMOND] [F5] with <i>eqwuser()</i> .....	43
Pop-up menus in <i>eqwuser()</i> .....	44
Define EQW extensions as program name strings .....	45
A comprehensive <i>eqwuser()</i> example .....	45
<i>TI Menus and Equation Library</i> .....	49

## Contents, continued

<b>More examples</b> .....	50
Limitations on using <i>ans()</i> and <i>entry()</i> .....	50
Entering and editing matrices .....	52
Using indexed variables: lists, vectors and matrices .....	57
Working with strings .....	58
Define a function in EQW .....	63
Using <i>NewProb</i> in EQW .....	64
Symbolic manipulation and solving .....	65
Solving simultaneous equations with matrices .....	66
Use a form to evaluate an expression .....	68
Creating a form with conditional (test) operators .....	70
Nested multiplication .....	71
Creating and editing a list .....	72
Entering angles in degrees, minutes and seconds .....	73
Entering expressions with logical operators (and, or, not, xor) .....	74
Entering and evaluating a definite integral .....	77
Entering an indefinite integral (anti-derivative) .....	78
Entering derivatives .....	79
Entering a sum .....	81
Entering and evaluating a product .....	82
Entering a limit .....	83
Entering a differential equation .....	85
Using <i>augment()</i> .....	87
Entering vectors in polar, spherical and cylindrical coordinates .....	88
Using EQW with the Y= function editor .....	94
Use a pop-up menu of history display answers in EQW .....	95
Create a form for d°m's" in an EQW program .....	97
 <b>Fixing problems</b> .....	 99
"Syntax" error message .....	99
"Too few arguments" error message .....	99
"Missing (" error .....	100
EQW puts parentheses where I do not want them .....	100
I can't make the cursor go where I need to .....	100
I can't enter optional function arguments .....	101
How do I change the direction in a limit? .....	102
EQW will not let me delete a placeholder .....	102
I cannot edit a fraction in EQW .....	102
 <b>Wishlist</b> .....	 104

---

## Introduction

---

EQW is a graphical equation editor for the Texas Instruments TI-89 and TI-92+ calculators. With EQW, you create, edit and evaluate equations and expressions in a graphical format, instead of the usual entry-line format. EQW makes it is easier to create and edit expressions without entry errors. Expressions are displayed in 'pretty-print' format, as you enter them. You can use EQW as a calculation environment because EQW can also evaluate expressions and equations. This feature is possible because EQW supports the STO operator, the 'with' operator |, and most built-in functions and commands.

EQW extensions can be used to add features to EQW, and to customize it to your needs. EQW extensions are TI Basic programs that are executed while you are using EQW. You can use them to change mode settings, insert special characters, display a pop-up menu of the history display, and recall forms to EQW.

E.W. has won first prize for EQW, in the 2001 TI App development contest, 68000 platform division. The details are here:

<http://education.ti.com/student/AppContestWinners.html>

Note that this user manual does not apply to the prize-winning version of EQW, which is not yet released. A separate manual will be written for the flash app version.

EQW is a 'no-stub' assembly program. You do not need any shells or kernels to run EQW. If you try to run EQW with a shell, you will probably crash your calculator and lose the contents of memory.

This is a large manual, which does not mean that EQW is difficult to use. The manual is complete and many detailed examples are used to help you get the most out of EQW.

I use square brackets to indicate keys on the calculator. For example [ENTER] means the ENTER key. [DIAMOND] means the green diamond key. [LEFT], [RIGHT], [UP] and [DOWN] refer to the cursor movement keys, which are on the blue cursor keypad on the TI-92+. [SHIFT] is the alpha keyboard 'shift' key, that is, the key that results in 'A' instead of 'a'. Shifted key functions are identified by the function name, for example, [UNITS] instead of [DIAMOND] [UNITS]. I use [\*] to indicate the multiplication key, to avoid confusion with the 'x' character key. The *home screen* is the normal TI-89/92+ home screen. The *entry line* is the bottom line on the home screen, where you enter expressions and commands.

EQW can edit expressions, equations, lists and matrices. Unless otherwise specified, I use *expression* to refer to all of these.

Screenshots from the TI-92+ are used to illustrate this manual. Screenshots from the TI-89 are identical or similar. Most screenshots are from previous EQW versions and do not show the softkey menu at the bottom of the display. The screen shots are optimized to print well, and so may be hard to read on the screen. Use the Zoom feature in Acrobat Reader to enlarge the screen shots if they are difficult to read.

I assume that you are familiar with the basic operation of your calculator. If not, refer to the TI Guidebook or post questions on the TI discussion groups. The complete TI-89 and TI-92+ Guidebooks are available free in PDF format, at

<a href="http://education.ti.com/product/tech/89/guide/guides.html">http://education.ti.com/product/tech/89/guide/guides.html</a>	<i>for the TI-89</i>
<a href="http://education.ti.com/product/tech/92p/guide/guides.html">http://education.ti.com/product/tech/92p/guide/guides.html</a>	<i>for the TI-92+</i>

If you have questions, comments or bug reports about EQW, you may email Bhuvanesh Bhatt at

[equationwriter@yahoo.com](mailto:equationwriter@yahoo.com)

Bhuvanesh will forward correspondence to E.W.

The features in EQW are limited so that EQW requires less than 24K bytes for installation. This limitation is necessary for reliable operation on all hardware and AMS versions. Calculators with hardware version 2 running AMS 2.05 include a 'protection' scheme imposed by TI that causes hard crashes when running programs greater than 24K. This manual includes a 'wish list' with items submitted by the beta testers and other users. Some of these features may be added if TI removes or relaxes the 24K limit. E.W. is aware of the methods to bypass the 24K limit with AMS 2.05, but these methods obviously are not supported by Texas Instruments. Future AMS releases may break these bypass method and render EQW inoperable.

I thank E.W. for writing and releasing this program, which is a significant addition to the capability of the calculator. We would also like to thank the beta testers Bhuvanesh Bhatt, Andrew Cacovean, Phillip Hendrickson and Eddie Shore for valuable bug reports and suggestions. I also thank Bhuvanesh Bhatt for writing many of the calculus examples and the matrix example, and Eddie Shore for contributing many good examples. The manual is a collaborative effort with Bhuvanesh, Andrew and Eddie. I warmly thank them for their significant contributions, and for *greatly* reducing the amount of work on my part. This manual is a far better document, thanks to them.

I would also like to thank my wife, Janet K.S. Burkett, for her thorough proofreading of this version of the manual, and for many suggestions which improved the clarity of the descriptions.

This manual is distributed as an Adobe Acrobat Reader PDF file. You should be using Acrobat Reader 4.0, or greater, for best results. The file is bookmarked, so you can easily jump to a specific section. If the bookmarks are not automatically shown in a window at the left in Acrobat Reader, use the icon in the menu bar to display the bookmarks. The entries in the table of contents also act as links directly to the manual sections. Just click on the entry in the table of contents to jump directly to that section.

---

## Revisions

---

### Changes to EQW:

- Various minor display bugs fixed.
- Legends for [F1] - [F5] function keys (softkey menu) displayed at bottom of screen.
- [APPS] key executes *main\eqwuser()* program, if any
- [DIAMOND] [F1] ... [DIAMOND] [F5] keypresses are sent to *main\eqwuser()* program
- [DIAMOND] [1] ... [DIAMOND] [9] execute user programs *main\eqwprgm1()* ... *main\eqwprgm9()*, if they exist.
- [RCL] executes *main\eqwprgmr()*, if it exists.

### Changes to the User's Manual:

- Many typographical errors corrected.
- Updated email addresses and web site URLs.
- Mentioned E.W. winning the 2001 TI App development contest, in the Introduction.
- Updated some screen shots for EQW 1.17.
- Updated Uninstall section; mention global EQW variables and EQW extensions.
- Updated keys function table.
- New section added: *EQW Extensions*. Describes the *eqwuser()*, *eqwprgm1()* ... *eqwprgm9()* and *eqwprgmr()* programs, and the functions keys used to execute these programs. Also added mention of Steve Chism's *TI Menus* program, and Nevin McChesney's *Equation Library* program.
- In *Using forms*: added description of recalling forms in EQW with an EQW program, and building form strings with `char(255)`.
- In *Other features and tips*: Mentioned that any string can be built in EQW, including empty strings and those with the [ENTER] character. Mentioned that EQW uses parentheses only when mathematically necessary.
- Changed terminology from 'subscripted variables' to 'indexed variables' for lists, vectors and matrices.
- In *Working with strings*: changed description; all characters can be entered in strings. Provided examples of different methods.
- Simplified matrix example.
- New examples: Cylindrical/spherical vectors, using EQW with the Y= editor, pop-up history menu, and creating a form for d°m's".
- In *Fixing problems*: added Editing numerical fractions.
- In the Wishlist: removed accomplished wishes.

---

## Installing and uninstalling EQW

---

### Installing EQW

Use a zip extractor program (such as WinZip or PKzip) to extract the EQW program files from the distribution file EQW.ZIP. The EQW.ZIP distribution file contains these files:

eqw.89z	EQW main program for the TI-89
eqwx.89z	Launcher (use ONLY with AMS versions 2.03 or earlier) TI-89
eqw.9xz	EQW main program for th TI-92+
eqwx.9xz	Launcher (Use ONLY with AMS versions 2.03 or earlier) TI-92+

*(The following programs are not required by EQW, but add useful features. Do not open files marked with \* in Graph Link, just use Link, Send in Graph Link to send the programs. .9xp programs can be used on the TI-89 also.)*

eqwuser.89p *	Custom menus demonstration program for the TI-89
eqwuser.9xp *	Custom menus demonstration program for the TI-92+
eqwusera.9xp	eqwuser() program to toggle Exact/Auto and Deg/Rad modes
eqwuserb.9xp	Comprehensive eqwuser() toolbar program
eqwprgm2.9xp	Pop-up history menu
eqwprgm3.9xp	Toggle Exponent, Complex and Vector modes
eqwprgm4.9xp	Insert special characters
eqwprgm8.9xp	Insert d°m's" form
eqwprgmr.9xp	Recall form with [RCL]
eqwguide 1_17.pdf	This user's guide
basic_ext.zip	Utility functions for TI Basic extensions; see below

These utility functions are included in the *basic\_ext.zip* file:

rcdclip.89z	Return clipboard contents (TI-89)
stoclip.89z	Save argument to clipboard (TI-89)
sendtext.89z	Send string argument to entry line (TI-89)
rcdclip.92z	Return clipboard contents (TI-92+)
stoclip.92z	Save argument to clipboard (TI-92+)
sendtext.92z	Send string argument to entry line (TI-92+)

These utility functions are not needed by EQW. They are provided so you can write TI Basic programs which interface with EQW and provide additional functionality. *sendtext()* is included only because it is a useful, interesting function, and it is not used with EQW.

Separate program versions are provided for the TI-89 and the TI-92+. You must use the correct version for your calculator:

eqw.89z	<i>for the TI-89</i>
eqw.9xz	<i>for the TI-92+</i>

The most current version of the TI-89/92+ operating system (called the AMS) is 2.05. We strongly recommend that you upgrade your calculator to AMS 2.05, because it is more stable, reliable, and several AMS bugs have been fixed. You can upgrade the AMS either from a personal computer (PC), or from another calculator which already has AMS 2.05 installed. If you install the AMS from a PC, you will need the TI Graph Link cable and software.

If you are using AMS version 2.03 or earlier on either calculator, you must use a *launcher*, and these are included in the ZIP file:

eqwx.89z	<i>Launcher for the TI-89</i>
eqwx.9xy	<i>Launcher for the TI-92+</i>

You need the TI Graph Link cable and software to download the programs to your calculator. The GraphLink software is free and available here:

<http://education.ti.com/product/accessory/link/features/features.html>

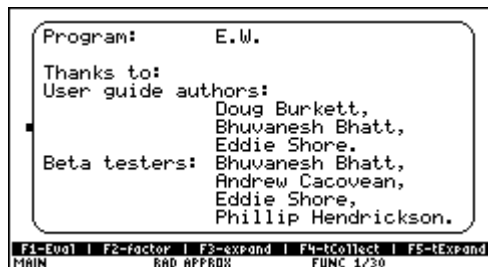
Do not try to open the *eqw* or *eqwx* files in the Graph Link software. These are compiled C object code files, and cannot be opened. Instead, just use the Link, Send menu options to send the files to your calculator.

EQW requires about 24000 bytes to install. EQW can be installed in any folder. EQW and the launcher may be archived to save RAM; refer to the *TI-89/TI-92+ Guidebook* for instructions to archive the programs.

You can determine the version number of EQW by pressing [MODE], which shows this screen:



Press [ESC] to clear the screen, or press [ENTER] for the second page:





## ***Uninstalling EQW***

To uninstall EQW, unarchive it if necessary, then delete it. If you use the launcher program, unarchive it and delete it, too. EQW creates global variables *main\eqwrun* and *main\eqwcom*; delete these. If you added any EQW extensions (*eqwuser()*, *eqwprgm1()* ... *eqwprgm9()*, *eqwprgmr()*) in the *main\* folder, delete these.

## ***Crashes and recovery***

There are no known conditions that cause EQW to crash (or 'freeze up', or 'lock up', or 'black bar'). It is remotely possible that you might discover such a condition. If this happens to you, please email Bhuvanesh Bhatt ([equationwriter@yahoo.com](mailto:equationwriter@yahoo.com)) with a description of the crash, and the conditions that caused the crash. He will forward the information to E.W.

It should be impossible to enter an expression that the calculator operating system (OS) cannot recognize. Again, it is remotely possible that you might discover one. If so, the display will show

<<...>>

If this happens, *do not* attempt any more operations with EQW, or you may crash the calculator. Instead, press [DIAMOND] [CLEAR] to clear the EQW screen, or press [INS] to undo the operation. If you follow this procedure, you can continue to safely use the calculator. If this happens to you, please email Bhuvanesh with a description of the expression that caused the <<...>> display.

---

## Starting and exiting EQW

---

### Starting EQW

In this section I assume that you start EQW without the launcher. If you must use the launcher, then you need to use `eqwx()` instead of `eqw()`.

You can start EQW with either an empty screen, or you can pass it an expression to edit. To start EQW with an empty screen, use

`eqw()`

There are two ways to start EQW with an expression to edit: with or without double quotes, like this:

`eqw(expression)`                      *Edit 'expression'*

`eqw("expression")`                      *Edit 'expression' without simplifying*

If you start EQW without an expression, the screen looks like this:



The softkey menu at the bottom of the display shows five of the available softkeys, [F1] through [F5], which are used to apply common functions to expressions in EQW. [F6], [F7] and [F8] also execute softkey functions, but they are not shown in the softkey menu. For more details, refer to the section below, *Applying functions with [F2] - [F7]*. You can display all the softkey functions by pressing [MODE].

The cursor shows that you are ready to start entering your expression. If you start EQW with this expression

`eqw(a+b/x)`

then the screen looks like

A screenshot of the EQW screen showing the expression  $\frac{b}{x} + a$ . The expression is displayed in a large font, and the cursor is positioned at the end of the expression.

Note that the order of the terms has changed, but the expression is the same. This type of simplification is also performed when you enter the expression in the calculator entry line. To keep the original form of the expression and prevent simplification, use double quotes around the expression:

`eqw("a+b/x")`

and the EQW screen will show

$$a + \frac{b}{x}$$

The original order of the terms is maintained. Using quotes to avoid simplification is particularly useful when you want to use a previous result to create a new expression. For example, suppose you have the expression

```
limit(k/(k+2),k,1,1)
```

If you start EQW with this expression it will be simplified to 1/3, which is probably not what you wanted. To avoid this, use double quotes around the expression:

```
eqw("limit(k/(k+2),k,1,1)")
```

then EQW shows this:

$$\lim_{k \rightarrow 1^+} \left( \frac{k}{k+2} \right)$$

and you can edit the expression as needed.

You can also start EQW with `ans(n)` or `entry(n)` as arguments, to edit expressions from the history area. Here *n* is the number that indicates which level of the history area will be used. If you have expressions stored as variables, you can edit those simply by starting EQW with the variable name. Some examples:

<code>eqw(ans(1))</code>	<i>edit the most recent answer from the history display</i>
<code>eqw(entry(2))</code>	<i>edit the entry in the second history level</i>
<code>eqw(xx)</code>	<i>edit the contents of variable 'xx'</i>

You can also use the built-in TI-89/92+ *copy* and *paste* features to copy expressions into EQW. Before starting EQW, highlight the expression you want to edit in EQW and press [COPY]. Start EQW, then press [PASTE]. The copied expression will be pasted into EQW.

If you use EQW often, it may be worthwhile to start it with a keyboard program. To start EQW with [DIAMOND] [1], use this program:

```
kbdprgm1
Prgm
main\eqw()
EndPrgm
```

This program assumes that EQW is installed in the *main* folder. If you have EQW installed in a different folder, change *main* to the name of the installation folder. If you use the `eqwx()` launcher instead of `eqw()`, you will need to change the line `main\eqw()` to `main\eqwx()`. To use a different key instead of [1] to start EQW, change `kbdprgm1` to `kbdprgmx`, where *x* is the number of the key you want.

### ***Exiting EQW***

You can exit EQW and return to the home screen by pressing either [ENTER] or [ESC]. If you press [ENTER], the expression in EQW is pasted to the entry line through the clipboard. If you press [ESC], the home screen is displayed, but the expression in EQW is *not* placed in the clipboard, and it is *not* returned to the entry line. It is simply discarded.

Since the expression is returned through the clipboard, you can paste the expression as needed with [PASTE], which is [DIAMOND] [V] on the TI-92+. This is a convenient way to paste expressions to the Y= editor or other built-in applications. Also, if you run EQW from Lars Frederiksen's RPN program, EQW returns the expression to RPN's entry line.

The automatic simplification of the TI-89/92+ may change the appearance of the expression returned to the entry line. You can prevent automatic simplification by pressing ["] to convert the expression to a string, before exiting EQW.

You cannot exit EQW with [ENTER] if the expression includes place-holders that are not filled in. You can exit EQW with [ESC], in this case. If you want to leave the place-holders in the expression and return it to the entry line, first convert it to a form in EQW. Refer to the section *Using Forms* below for more details.

---

## Basic operations with EQW

---

This section is an overview of the basic operation of EQW. Additional detailed examples of specific operations are given in the *More Examples* section.

The main benefit of EQW is that your expression is displayed in 'pretty print' form as you enter it. This means that it is easy to change operations and arguments, and you can immediately see that your changes are correct.

One way to learn EQW is to just start using it to enter and edit some equations. If you get stuck or can't figure out how to perform an operation, check the summary of EQW keys and functions in this section, below. If that doesn't help, try the examples in this manual. If you still can't get EQW to do what you want, try posting your question on the TI89/92+ discussion group at

[http://www-s.ti.com/cgi-bin/discuss/sdbmessage.cgi?databasetoopen=calculators  
&topicarea=TI-89/92+Plus&do\\_2=1&viewmethod=Thread](http://www-s.ti.com/cgi-bin/discuss/sdbmessage.cgi?databasetoopen=calculators&topicarea=TI-89/92+Plus&do_2=1&viewmethod=Thread)



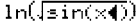

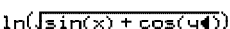


Another way to learn EQW is to read through this manual and work the examples. I have tried to explain EQW clearly, but my explanations may not reveal just how easy it is to use EQW. If you work through the examples, you will learn how to do everything that EQW can do.

In general, you just type your expression variables and operations as needed. However, since you can use EQW to select expressions, you rarely need to type any parentheses: EQW provides them as needed. First, an example. If we want to enter this expression:

$$\ln\left(\sqrt{\frac{\sin(x)+\cos(y)}{\tan(x^2+y^2)}}\right)$$

then we start EQW and use these steps:

### Example: Creating an expression

Keystrokes	Result	Description
[LN]		The <i>ln()</i> function is entered. The large empty rectangle (called the outline box) shows the selected expression. The small black square (called the place holder) shows that no argument is entered for <i>ln()</i> . Note that EQW provides the parentheses.
[SQRT]		The square root operator is entered inside the parentheses for <i>ln()</i> .
[SIN] [x]		The <i>sin()</i> function and its argument 'x' are entered. Again note that EQW provides the parentheses. The arrow cursor shows the insertion point for more characters, if needed.
[UP]		The complete <i>sin(x)</i> expression is marked, as indicated by the inverse shading. This means the next operation will be applied to <i>sin(x)</i> .
[+] [COS] [y]		Add <i>cos(y)</i> to <i>sin(x)</i>
[UP] [UP]		Press [UP] twice to mark the entire function
[/]		Pressing [/] places the cursor in the denominator of a new fraction, with a place holder.

[TAN] [x] [^] [2]	$\ln\left(\frac{\sin(x) + \cos(y)}{\tan(x^2)}$	Note that the cursor is in the exponent of x
[UP]	$\ln\left(\frac{\sin(x) + \cos(y)}{\tan(\text{█})}\right)$	Mark the argument of $\tan()$
[+] [y] [^] [2]	$\ln\left(\frac{\sin(x) + \cos(y)}{\tan(x^2 + y^2)}$	Finish entering the argument for $\tan()$ . The expression is complete. Press [ENTER] to send it to the entry line if needed.

Note that:

- Pressing a function key displays the function and both parentheses, and places the cursor in the function argument between the parentheses.
- The left-arrow cursor shows the current insertion point. This may be the point that you want to type more text, or it may not. If not, you need to use [UP] to mark the expression, then apply the next operation to the complete expression. You can repeatedly press [UP] to mark increasingly higher levels of the expression.

The tables on the following pages show the keys which can be used in EQW. For keys used to enter and edit matrices, see the example *Entering and editing matrices*.

### Summary of EQW keys and functions

[UP], [DOWN]	Move one level up or down in the expression
[2nd] [UP]	Jump to highest level of the expression
[2nd] [DOWN]	Jump to lowest level of the expression
[LEFT], [RIGHT]	Move cursor forward or backward in the expression
[SHIFT] [LEFT]	Scroll left, to view large expressions
[SHIFT] [RIGHT]	Scroll right, to view large expressions
[SHIFT] [UP]	Scroll up, to view large expressions
[SHIFT] [DOWN]	Scroll down, to view large expressions
[2nd] [LEFT]	Move cursor to start of the expression
[2nd] [RIGHT]	Move cursor to end of the expression
[CUT]	Cut the marked expression ([DIAMOND] [X] on the TI-92+)
[COPY]	Copy the marked expression ([DIAMOND] [C] on the TI-92+)
[PASTE]	Paste a copied or cut expression at the cursor. ([DIAMOND] [V] on the TI-92+)
[INS]	Undo the last operation (up to 10 operations)
[BACKSPACE]	Delete characters or expression marked by the cursor
[CLEAR]	Clear the selected expression; replace it with a place-holder
[DEL]	Delete a function with a single argument, but leave the argument
[DIAMOND] [CLEAR]	Clear the entire EQW screen
[STO]	Insert the 'store' operator →
[RCL]	Execute <i>main\eqwprgmr()</i> if it exists. Refer to <i>Running programs from EQW</i> .
[F1]	Evaluate the marked expression
[F2]	Apply <i>factor()</i> to marked expression
[F3]	Apply <i>expand()</i> to marked expression
[F4]	Apply <i>tCollect()</i> to marked expression
[F5]	Apply <i>tExpand()</i> to marked expression
[F6]	Apply <i>comDenom()</i> to marked expression
[F7]	Apply <i>propFrac()</i> to marked expression
[F8]	Insert a <i>Define</i> form at the cursor
[MODE]	Display 'About' and help screens
[MATH], [VAR-LINK], [CATALOG], [UNITS], [CHAR]	Display built-in menus

### Summary of EQW keys and functions, continued

[+], [-], [*], [/], [^], [√]	Basic arithmetic operators
[(-)]	Negation (sign change) operator. Also used to change <i>limit()</i> direction
[x <sup>-1</sup> ]	Reciprocal (TI-92+ only)
[SIN], [COS], [TAN], [SIN <sup>-1</sup> ], [COS <sup>-1</sup> ], [TAN <sup>-1</sup> ]	Trigonometric functions
[∠], [π], [∞], [i]	Angle operator, pi, infinity, complex unit
[°]	Degree operator (use [,] to add minutes and seconds)
[=], [≠], [<], [>], [≤], [≥]	Conditional operators
[LN], [e <sup>x</sup> ]	Natural logarithm functions
[d], [∫], [∑], [Σ]	Derivative, integral, differential and summation (TI-92+) operators
[!]	Factorial. TI-92+ is [2nd] [W]. TI-89 is [DIAMOND] [/]
[>], [ ]	Unit conversion symbol, unit prefix symbol
<hr/>	
[ANS], [ENTRY]	<i>ans()</i> and <i>entry()</i> functions
[&]	String concatenation TI-92+ is [2nd] [H]. TI-89 is [DIAMOND] [*]
[#]	Indirection operator. TI-92+ is [2nd] [T]. TI-89 is [CHAR] [3] [3]
<hr/>	
[DIAMOND] [ENTER]	Evaluate the marked expression in Approx mode
<hr/>	
[DIAMOND] [,]	Insert a variable index such as <i>k</i> in <i>list[k]</i>
<hr/>	
[DIAMOND] [1] ... [DIAMOND] [9]	Execute programs <i>mainleqwprgm1()</i> ... <i>mainleqwprgm9()</i> , if they exist. Refer to <i>Running programs from EQW</i> .
<hr/>	
[DIAMOND] [F1] ... [DIAMOND] [F5]	These keystrokes are passed to <i>mainleqwuser()</i> . Refer to <i>Running programs from EQW</i> .
<hr/>	
[[]]	Insert a matrix
[;]	Add a row to a matrix, when an element or row is marked
[{]	Insert a list
["]	Convert marked expression to string
[,]	Add optional arguments to marked expression. Add arguments to functions with a variable number of arguments. Add elements to lists; add columns and rows to matrices.
<hr/>	
[APPS]	Execute <i>mainleqwuser()</i> if it exists. Refer to <i>Running programs from EQW</i> .
<hr/>	
[ENTER]	Exit EQW and paste the current expression to the home screen command line; or move cursor to next placeholder.
[ESC]	Exit EQW; do not paste the current expression to the command line
<hr/>	
[OFF]	Turn the calculator off. When you next turn the calculator on, EQW resumes.



## Moving the cursor

You will want to move the cursor in an expression to correct mistakes or to change the expression. EQW provides several cursor-control keys to accomplish this. I will use a few terms that make the description easier to understand:

- an *expression* is the complete expression shown on the EQW screen.
- a *subexpression* is part of an expression
- an *element* is smallest part of an expression that can be selected.

For example, if the expression is  $4*a*b + 3*c*d$ , then the subexpressions are  $4*a*b$  and  $3*c*d$ , and the elements are 3, 4, a, b, c and d.

This section may seem like a lot of words to describe something as simple as moving the cursor, and there is a good reason for that. Moving the cursor and selecting subexpressions is easy, but, like some other easy things in life, it seems to take a *lot* of words to completely describe it. So, please don't be put off by the length of this section. EQW *is* easy to use, and this cursor movement description may make it seem more difficult than it really is!

Expressions are internally stored in the same format that is used by the TI-89/TI-92+. This means that the cursor movement might not always be what you expect, but there is always a way to move the cursor to the desired subexpression or element. While this description attempts to describe cursor movement, the best way to learn is to try editing some expressions. Later in this section I describe the cursor movement in terms of the internal representation of the expression.

EQW has three different cursor styles: an arrow cursor, an outline box and inverse shading. An expression which is shown in inverse shading is called *marked*.

This figure shows the arrow cursor pointing at the *b* element:



a + b◀

When the arrow cursor is shown, you can edit the element by typing more characters. You can use the [BACKSPACE] key to delete the element.

This figure shows an outline box surrounding the *b* element:



a + b

When an element is surrounded by the outline box, you can immediately type new characters to change the element. You may press [BACKSPACE] to delete the entire element. The outline box always surrounds a single element, and may include a negation symbol. You may use [LEFT] and [RIGHT] to move the outline box, one element at a time. Use [2nd] [LEFT] and [2nd] [RIGHT] to move the outline box to the beginning and end of the expression, respectively. Press [BACKSPACE] to change the outline box to the arrow cursor, then you can change or delete the element.

This figure shows inverse shading of the *a+b* expression:



a + b

Inverse shading shows that the subexpression is *marked*. When an expression is marked and you press [BACKSPACE], the expression is deleted. You can also press a function key, to apply the

function to the marked subexpression. For example, if you press [SIN] when  $a+b$  is marked, the result is  $\sin(a+b)$ .

[UP] and [DOWN] are used to change which parts of the expression are marked. [UP] marks more of the expression, and [DOWN] marks less of the expression. For example:

#### Using [UP] and [DOWN] to mark subexpressions

Key press	Display	Description
(None)	$\frac{-b + \sqrt{b^2 - 4 \cdot a \cdot c}}{2 \cdot a}$	Start with the outline box cursor around b
[UP]	$\frac{-b + \sqrt{b^2 - 4 \cdot a \cdot c}}{2 \cdot a}$	
[UP]	$\frac{-b + \sqrt{b^2 - 4 \cdot a \cdot c}}{2 \cdot a}$	
[UP]	$\frac{-b + \sqrt{b^2 - 4 \cdot a \cdot c}}{2 \cdot a}$	
[UP]	$\frac{-b + \sqrt{b^2 - 4 \cdot a \cdot c}}{2 \cdot a}$	
[UP]	$\frac{-b + \sqrt{b^2 - 4 \cdot a \cdot c}}{2 \cdot a}$	The complete expression is marked
[DOWN]	$\frac{-b + \sqrt{b^2 - 4 \cdot a \cdot c}}{2 \cdot a}$	
[DOWN]	$\frac{-b + \sqrt{b^2 - 4 \cdot a \cdot c}}{2 \cdot a}$	The cursor automatically changes to the outline box at the lowest level of the expression

This example shows that [UP] and [DOWN] do not always produce symmetrical results. In general, you can move the outline box cursor to any subexpression, then use [UP] to mark the subexpression.

Some additional explanation may help clarify cursor movement. The cursor has two basic modes:

1. *Expression* mode, in which the cursor moves from one expression (or sub-expression) to another.
2. *Element* mode, in which the cursor moves among elements of an expression or sub-expression.

EQW switches between these modes automatically, so it is not necessary for you to control the mode. This automatic switching makes EQW easier to use. However, in some cases the cursor mode automatically changes, and you will need to change the mode to do what you want to do. You manually change the mode with [UP] and [DOWN].

When an expression is marked, the cursor is always in expression mode. When the cursor is an outline box, it may be in either mode. If the outline cursor stops unexpectedly at the end of an expression and you cannot make it move farther, the cursor is in expression mode. In this case, press [DOWN], and you can again move the cursor. Pressing [DOWN] when the outline cursor is shown, or pressing [2nd]

[DOWN] when it is not, changes the mode to *element*. Pressing [UP] always changes the cursor to expression mode.

### Cursor movement, in terms of expression representation

This section describes cursor movement in terms of the internal representation of expressions in EQW. You do not need to read or understand this section to use EQW efficiently. However, this description explains the logic behind cursor movement in EQW.

As mentioned above, EQW stores expressions in the same format that is used internally by the TI-89/92+. The format is called *polish notation*, in honor of the Polish mathematician, Jan Lukasiewicz, who developed it. For more information on Lukasiewicz, see

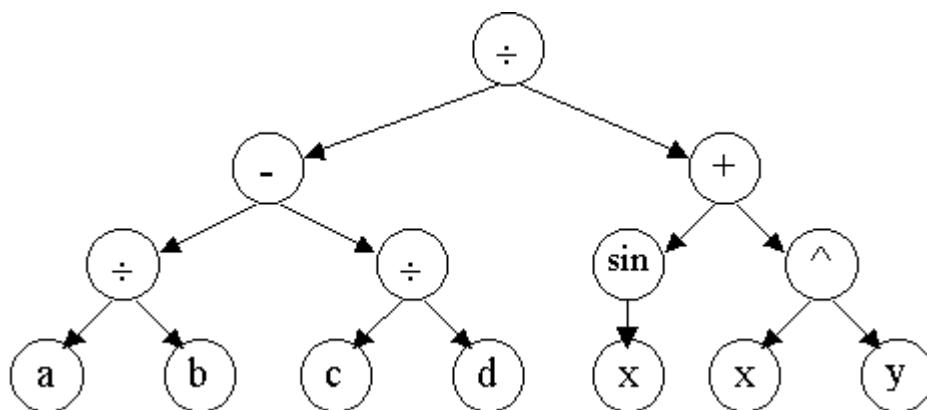
<http://www-groups.dcs.st-and.ac.uk/~history/Mathematicians/Lukasiewicz.html>

Polish notation can describe any expression unambiguously, without parentheses. In practical terms, the method is particularly useful when the expression is saved in a *stack*, which is a type of data structure. As an example, this expression

$$\frac{\frac{a}{b} - \frac{c}{d}}{\sin(x) + x^y} \quad \text{or} \quad ((a/b) - (c/d)) / (\sin(x) + x^y)$$

would be represented in polish notation as: / - / a b / c d + sin x ^ x y

Expressions can also be shown as a *tree diagram*. The nodes (circles) of the diagram show the variables and operators. This tree diagram shows the expression above.



The operation of the cursor movement keys can be described in terms of the tree diagram. The diagram shows that the expression has four levels. The top-most level is the division operator. The bottom level consists only of variables. This is not *always* true for all expressions: some may have variables at other levels.

To try this example, start EQW and use these keystrokes to enter the expression:

[a] [/] [b] [UP] [-] [c] [/] [d] [UP] [UP] [/] [SIN] [x] [UP] [+] [x] [^] [y]

Now press [2nd] [UP]. [2nd] [UP] always marks the entire expression, because [2nd] [UP] moves the cursor to the *top* of the expression. Press [DOWN], and the numerator  $(a/b)-(c/d)$  is marked, because the cursor is at the *second* level of the expression. Press [RIGHT], and the denominator is marked, because the denominator is on the same level as the numerator: [LEFT] and [RIGHT] move the cursor (or mark the subexpressions) on the *same* level. Press [LEFT], and the numerator is marked again.

Press [DOWN]. The expression  $a/b$  is marked, because pressing [DOWN] moves the cursor down one level: the cursor is at the third level from the top. Press [RIGHT], and  $c/d$  is marked. Press [LEFT], and  $a/b$  is marked again.

Press [DOWN] to move the cursor to the lowest level of the expression, and  $a$  is marked. The cursor changes to the outline box, because the subexpression  $a$  is an element: it contains no operations. If you press [RIGHT] repeatedly, the cursor moves to the variables  $b$ ,  $c$ ,  $d$ ,  $x$  and so on, because these are all on the lowest level of the expression.

In the same way that [DOWN] moves the cursor down expression levels, [UP] moves the cursor up in the expression. Move the cursor to the  $x$  in  $\sin(x)$ , then press [UP]. The expression  $\sin(x)$  is marked, since it is one level above  $x$ . Press [UP] again, and the complete numerator of the expression is marked: the cursor has moved to the second level from the top. Press [UP] once more, and the complete expression is marked, since the cursor moves to the top level.

Because of the expression storage method and the operation of EQW, the cursor can sometimes get 'stuck' and refuse to move. For example, press [2nd] [UP], then press [DOWN], three times, so that  $a'$  is selected. Press [RIGHT] six times, so that  $y$  is selected. Pressing [RIGHT] at this point has no effect, since there are no more elements to the right of  $y$  in the expression tree.

The cursor can also get stuck even when there are elements that *could* be selected when [LEFT] or [RIGHT] is pressed. With  $y$  still selected, press [LEFT] twice, so that the  $x$  in  $\sin(x)$  is selected. Press [UP] to move up one level, and mark  $\sin(x)$ . Press [RIGHT] to mark  $x^y$ . Press [LEFT], so that  $\sin(x)$  is marked again, then press [LEFT] again: the  $x$  in  $\sin(x)$  is selected with the outline cursor. In this case, pressing [LEFT] has actually moved the cursor *down* one level. Pressing [RIGHT] in this case has no effect, because EQW has *automatically* moved the cursor down one level. However, pressing [LEFT] selects  $\sin(x)$ , actually moving up one level. Pressing [LEFT] at this point alternates between selecting  $x$  and  $\sin(x)$ .

When the cursor gets stuck like this, you can always press [UP] to move up one level, and continue to move the cursor as needed. In terms of a strictly self-consistent user interface this operation may seem awkward, because the level is automatically changed without [UP] or [DOWN] being pressed. However, this default operation of EQW is often the most efficient because EQW usually marks the most appropriate expression or element.

If you are still having difficulty getting the cursor to go where you want, there is a simple but effective way to move to any element: press [2nd] [UP] to mark the entire expression, then press [2nd] [DOWN]. The outline box cursor will be placed at the first expression element, and you can repeatedly press [RIGHT] to move the cursor to each element.

## Deleting expressions and characters

These keys can be used to delete expressions, subexpressions and characters:

[BACKSPACE]	Delete one character at a time
[DEL]	Delete a function but not the function's arguments
[CLEAR]	Similar to [BACKSPACE], but can delete simple expressions
[DIAMOND] [CLEAR]	Clear the entire screen

Note that you can undo any deletion with [INS].

[DIAMOND] [CLEAR] clears the entire screen, regardless of the cursor type or position.

[BACKSPACE] and [DEL] are used to delete characters and elements. [BACKSPACE] is used when the arrow cursor is shown, and one character is deleted for each [BACKSPACE] press. For example:

Start with this expression	abc + xyz
Push [BACKSPACE] once	abc + xy
Push [BACKSPACE] again	abc + x
Push [BACKSPACE] again. Note that arrow cursor changes to outline box and place holder	abc +
Push [BACKSPACE] again. Note the the '+' operator is deleted.	abc
Push [BACKSPACE] again. Note that the outline box changes to the arrow cursor.	abc

[BACKSPACE] can also be used when the outline box is shown: in this case, the outline box changes to an arrow cursor, as shown in the last row of the example above.

If [BACKSPACE] is pressed when a subexpression is marked, the subexpression is deleted and replaced with the place-holder:

Start with this expression	abc + def + hij
Push [LEFT] once. The outline box is around the def subexpression.	abc + def + hij
Push [UP] to mark the subexpression	abc + def + hij
Push [BACKSPACE]. The expression is deleted, and the outline box is shown around a placeholder.	abc +

You can now enter a new subexpression.

[CLEAR] is used to delete complete subexpressions, and replace them with a place-holder as this example shows:

Start EQW and press these keys to enter the expression:  
[a] [b] [c] [\*] [d] [e] [f] [+] [g] [h] [i]

Suppose you really wanted variable xyz instead of abc. Press [LEFT] [LEFT] so that the outline box surrounds abc.

Press [CLEAR]. The variable name abc is cleared, and a place-holder is shown.

Press [x] [y] [z] to enter the desired variable name. The expression is complete.

abc · (def + ghi)

MAIN

RAD AUTO

FUNC 4/30

abc · (def + ghi)

MAIN

RAD AUTO

FUNC 4/30

· (def + ghi)

MAIN

RAD AUTO

FUNC 4/30

xyz · (def + ghi)

MAIN

RAD AUTO

FUNC 4/30

The [DEL] key is used to delete a function and leave the argument of the function. This feature only works with functions of a single argument. For example:

Start with this expression. We want to change the square root to cos()	$\sqrt{a+b} + \sin(c)$
Press [UP] twice to mark the square root expression.	$\sqrt{a+b} + \sin(c)$
Press [DEL]. The square root function has been deleted.	$a+b + \sin(c)$
Press [COS]. Since the 'a+b' subexpression is still marked, the cos() function is applied as needed.	$\cos(a+b) + \sin(c)$

We also want to change $\sin(c)$ to $\sinh(c)$ . Press [RIGHT] once, to mark $\sin(c)$ .	$\cos(a + b) + \sin(c)$
Press [DEL] to delete the $\sin()$ function.	$\cos(a + b) + \square$
Press [2nd] [MATH] [Hyperbolic] [sinh] to select the $\sinh()$ function. The expression is complete	$\cos(a + b) + \sinh(c)$

In the last step of this example, we used the MATH menu to insert the  $\sinh()$  function. Alternatively, we could have just typed the  $\sinh()$  function name:

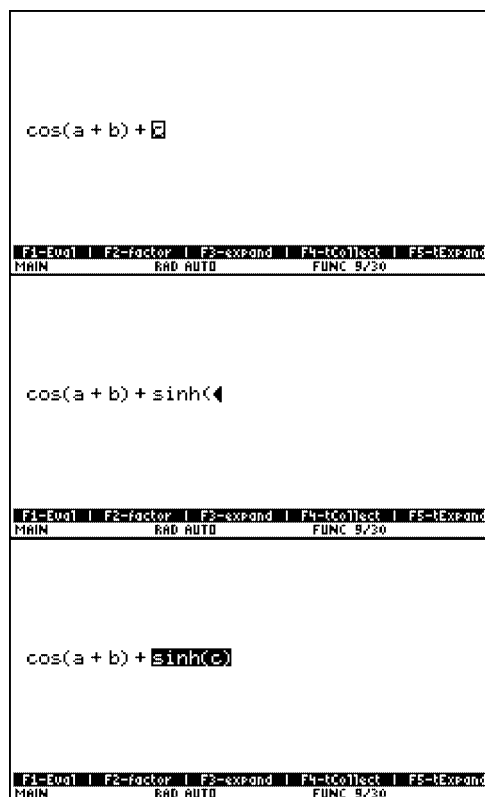
As above, we have pressed [DEL] to delete the  $\sin()$  function.

Type [S] [I] [N] [H] [ ( ] to enter the  $\sinh()$  function.

Note that the argument  $c$  temporarily disappears.

Press [RIGHT] or [UP] to finish the function name entry.

The expression is complete.



You can also use [CUT] to delete marked expressions and elements.

## Using the built-in menus

You can use the built-in menus to insert function names and characters in expressions in EQW. The menus are:

[MATH]	<i>built-in math functions and programs</i>
[VAR-LINK]	<i>user functions, programs and variables</i>
[CATALOG]	<i>both built-in and user programs and functions</i>
[UNITS]	<i>units of measure</i>
[CHAR]	<i>special characters</i>

To use the built-in menus, move the cursor to the point at which you want to insert the object, then press the menu key. Select the object, and press [ENTER] to insert it. This example shows some typical built-in menu usage.

Start EQW.

Press [CHAR] [1] [1] to insert the  $\alpha$  character.

Press [+].

Press [CHAR] [1] [2] to insert the  $\beta$  character.

Press [UP] to mark the entire expression.

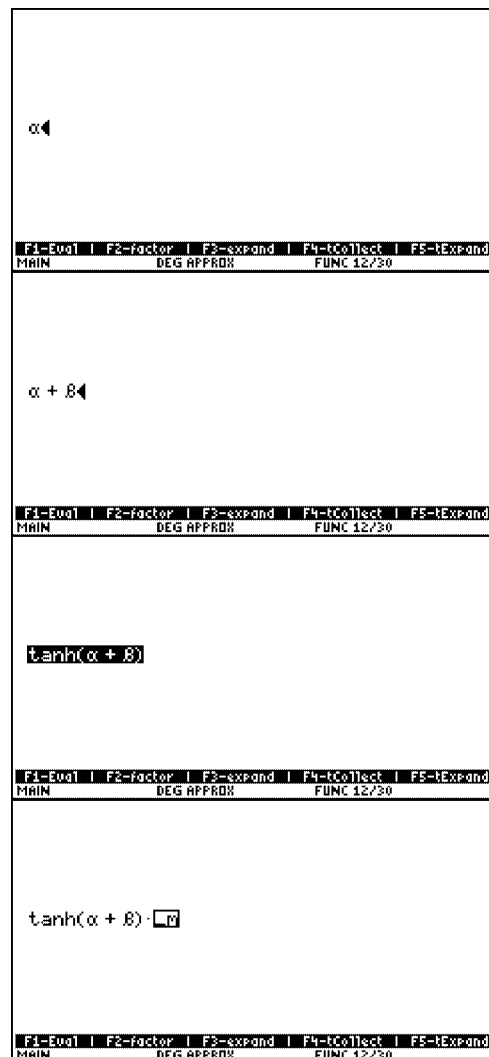
Press [MATH] [B] [3] to insert the  $\tanh()$  function.

Press [UNITS].

Press [DOWN] [RIGHT] to open the Length units menu.

Press [DOWN] to scroll down to the  $_m$  unit.

Press [ENTER] to add the unit to the expression. The expression is complete.





### ***Using cut, copy and paste***

EQW implements the standard *cut*, *copy* and *paste* editing functions. The built-in clipboard is used. This means that

- You can delete expressions or elements by marking them, then using [CUT].
- You can copy expressions, subexpressions or elements, and paste them elsewhere in the expression.
- You can copy or cut expressions and paste them into other built-in calculator applications, after exiting EQW.

The cut, copy and paste functions are not labeled on the TI-92+ keyboard, but they are available:

cut: [DIAMOND] [X]  
copy: [DIAMOND] [C]  
paste: [DIAMOND] [V]

It is not possible to cut or paste matrix rows into a matrix, although you may copy them. If you paste a copied matrix row, it is pasted as a list. It is possible to cut, copy and paste entire matrices.

### Using the UNDO feature

EQW has an undo feature which reverses the effects of previous operations. You can undo up to 10 operations. Press [INS] to undo an operation. This example shows some operations with UNDO.

We want to enter  $x \cdot y + z$ . Start EQW and press [x] [\*]  
[y] [UP] [-]

$x \cdot y -$

By mistake we pressed [-] instead of [+]. Push [INS]  
to undo pressing [-]. The original expression is  
shown.

$x \cdot y$

Press [+] [z] to complete the expression. Now  
suppose that we want to divide the expression by  
 $z^y$ . Press [2nd] [UP] to mark the complete  
expression, Push [z] [^] [y].

$\frac{x \cdot y + z}{z^y}$

Now suppose we want to mark this expression and  
copy it. Push [2nd] [UP] to mark the expression. But  
instead of pressing [COPY], we press  
[BACKSPACE] by mistake.

$\frac{x \cdot y + z}{z^y}$

We can recover from this mistake by pressing [INS]  
to undo the deletion, and the original expression is  
restored.

$\frac{x \cdot y + z}{z^y}$

UNDO will reverse the effects of any operation except deleting characters when the arrow cursor is shown. This is not a significant effect: UNDO usually does just what you expect. But this example shows what can happen when UNDO is used while entering and deleting characters in an expression.

Start EQW and press [2] [-] [5] [\*] [8]. Note that the triangle cursor is shown after '8'.

2 - 5 · 8◀

MAIN RAD AUTO FUNC 5/30

Press [CLEAR]. As expected, the last character entered is deleted and replaced with a placeholder.

2 - 5 · □

MAIN RAD AUTO FUNC 5/30

Press [INS] to attempt the UNDO operation. The expression is not changed, but just marked. In effect, pressing [CLEAR] has "undone" entering the '8' character, so UNDO has no effect.

2 - 5 · □

MAIN RAD AUTO FUNC 5/30

Press [CLEAR] to clear the entire expression and start over. Again, press [2] [-] [5] [\*] [8]. to enter the same expression. Press [UP], and note that 5\*8 is marked.

2 - 5 · 8

MAIN RAD AUTO FUNC 5/30

Press [CLEAR], and the expression 5\*8 is replaced by a place-holder.

2 - □

MAIN RAD AUTO FUNC 5/30

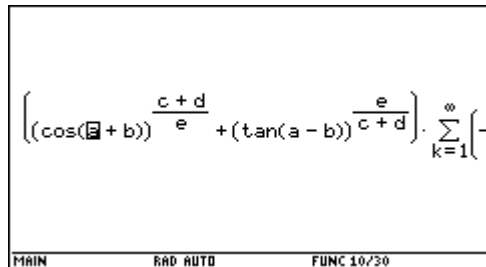
Press [INS] to perform the undo. In this case, the effect of the last [CLEAR] is undone, and the expression is restored.

2 - 5 · 8

MAIN RAD AUTO FUNC 5/30

### Scrolling large expressions in the display

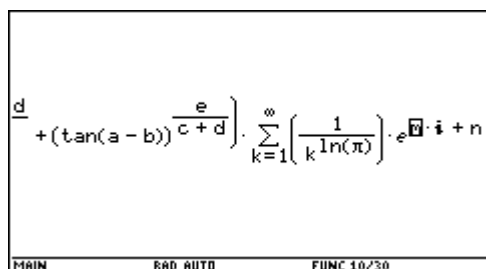
You can create expressions which are larger than the TI-89 or TI-92+ display. To show part of the expression that is off the screen, use [SHIFT] in combination with [LEFT], [RIGHT], [UP] and [DOWN] to move the expression. For example, this expression is too large to display on the TI92+ screen:



$$\left[ (\cos(\boxed{a} + b)) \frac{c+d}{e} + (\tan(a-b)) \frac{e}{c+d} \right] \cdot \sum_{k=1}^{\infty} \left( \frac{1}{k} \right)$$

MAIN RAD AUTO FUNC 10/30

Press [SHIFT] [RIGHT] a few times to see the end of the expression:



$$\frac{d}{e} + (\tan(a-b)) \frac{e}{c+d} \cdot \sum_{k=1}^{\infty} \left( \frac{1}{k \ln(\pi)} \right) \cdot e^{\boxed{a} \cdot i + n}$$

MAIN RAD AUTO FUNC 10/30

[SHIFT] [UP] and [SHIFT] [DOWN] work in a similar way to scroll the expression up and down in the display.

The expression will not scroll such that the cursor is moved off of the screen. You may need to move the cursor, then scroll the screen to show the end of an expression. You can also use [2nd] [LEFT] and [2nd] [RIGHT] to move to the beginning and end of the expression.

This scrolling feature is particularly useful on the TI-89 since it has a smaller screen than the TI-92+.

### Entering program and function names

You can enter the names of built-in functions, programs and commands, as well as your own user functions and programs. You can either type in the function name, or choose the function from the CATALOG, MATH or VAR-LINK menus.

*Note that function and program names are case-sensitive!* If you type in a built-in function or program name, it is only recognized if you use the correct upper- and lower-case characters: the recognition is case-sensitive. Built-in function names always begin with a lower-case letter, and each subsequent word has the first letter capitalized: *solve()*, *nSolve()* and *comDenom()*, for example. Built-in command names have the first letter capitalized as well, for example, *LinReg*, *SortA* and *DelVar*. User function and program names have all characters in lower-case.

EQW will automatically show place-holders for *required* arguments for functions. To add *optional* arguments, press [.] to add additional place-holders, then type the arguments.

There is only *one* exception to entering function names: you cannot enter *augment()*, with the semicolon separator. For example, you cannot enter *augment(m1;m2)*. Refer to the *Examples* section for more details on using *augment()*.

These examples show the two methods to enter built-in function names..

#### Entering a built-in function from a catalog

We want to enter the <i>remain()</i> function with arguments <i>a</i> and <i>b</i> . Start EQW, then press [CATALOG], select the <i>remain()</i> function, and press [ENTER]. <i>remain()</i> has two arguments, so two place-holders are shown.	<code>remain(□, □)</code>
Press [a] to enter the first argument. Press [RIGHT] to move the cursor to the next place-holder. Press [b]. The expression is complete.	<code>remain(a, b)</code>

#### Entering a built-in function by typing the function name

We want to enter the <i>rowAdd()</i> function, with arguments <i>mat1</i> , 2 and 3. Start EQW and type the function name; note that 'A' is capitalized. Press [R], then press [RIGHT]. The <i>rowAdd()</i> function is recognized, and the place-holders are shown.	<code>rowAdd(□, □, □)</code>
Type <i>mat1</i> then press [RIGHT], [2], [RIGHT], [3]	<code>rowAdd(mat1, 2, 3)</code>

The same methods are used to enter built-in programs. The method to enter built-in *commands* is slightly different, since commands do not use parentheses. You can choose the command from the CATALOG menu, and the place-holders are shown as above. If you would rather type in the command name, you press [SPACE], instead of [R], to display the place-holders for the arguments. This example shows how to enter the *LinReg* command and its arguments, as well as the optional arguments.

### Entering a built-in command by typing the name

Type the <i>LinReg</i> command name, with the correct capital letters. Push [SPACE], [RIGHT]. The command is recognized and the two argument place-holders are shown.	LinReg <input type="text"/> , <input type="text"/>
Type the first argument name, <i>xlist</i> , push [RIGHT], and type the second argument name, <i>ylist</i> . The expression is complete if only these arguments are needed.	LinReg xlist,ylist
To add optional arguments, press [,] and another place-holder is shown.	LinReg xlist,ylist, <input type="text"/>
Type the third argument <i>freq</i> . Push [,] to add another placeholder, and type the fourth argument <i>cat1</i> . Push [,] one more time, and type the fifth argument <i>cat2</i> . The expression is complete.	LinReg xlist,ylist,freq,cat1,cat2

User functions and programs can also be entered in EQW with the [CATALOG] function: press [CATALOG], then press [F4] to display your program and function names. Choose the function, then press [ENTER]. The function name and parentheses are inserted, and the place-holders are automatically shown. You can also type in the names of user functions, as in this example.

### Entering a user function by typing the name

We want to enter a user function called <i>erf()</i> . Start by typing the name.	erf
Push [ ( ], [ ) ] to enter the parentheses, then push [,] to put a place-holder between the parentheses.	erf( <input type="text"/> )
Type the argument name <i>x</i> . The expression is complete.	erf(x)

If a function has more than one argument, you add place-holders to enter those arguments with the [,] key. This example shows how to enter a user function *myfunc(1,2,3)*.

Start EQW and type the function name

myfunc

---

MAIN RAD AUTO FUNC 1/20

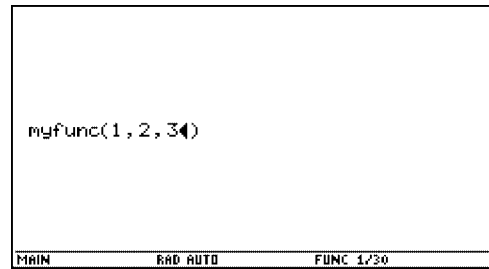
myfunc(

---

MAIN RAD AUTO FUNC 1/20

Press [(] [)] to add the parentheses, then press [,] to put a place-holder between the parentheses.

Add the arguments by pressing [1] [,] [2] [,] [3]  
The expression is complete



These hints may be helpful when entering functions:

- After you enter the parentheses for a user function or program, you can insert the first place-holder by pressing [,], or by marking the function expression and pressing [,].
- To delete function arguments, select the argument with the outline box, press [BACKSPACE] so that the arrow cursor is shown, then press [BACKSPACE] until a place-holder is shown, then press [BACKSPACE] once again, to delete the place-holder.
- Another way to delete program and function arguments is to select the argument with an outline box, then press [CLEAR].
- To add more place-holders to a function that is already completely entered, select the entire function and press [,]. You can also mark any function argument (except the last one) and press [,]. You can only add new place-holders after the existing arguments.

## Evaluating expressions with [F1] and [DIAMOND] [ENTER]

You can use [F1] to evaluate expressions within EQW. Simply mark the expression and press [F1].

Start with this expression	$\cos\left(\frac{\pi}{3}\right)$
Push [UP] twice to mark the expression	$\boxed{\cos\left(\frac{\pi}{3}\right)}$
Push [F1] to evaluate the expression. The evaluated result is shown.	$\boxed{0.5}$

You can also use [F1] with *ans()* and *entry()* to paste expressions from the history display. For example, suppose that the history display is

■ sin(x·y·z)	sin(x·y·z)
■ cos(a)	cos(a)
MAIN RAD AUTO FUNC 2/30	

then we can use these results in EQW:

Start EQW	$\boxed{\phantom{000}}$
Push [ANS] [1], then push [UP] to mark the expression.	$\boxed{\text{ans}(1)}$
Push [F1] to evaluate ans(1). The contents of the history display at level 1 are pasted.	$\boxed{\cos(a)}$
Push [+] [ENTRY] [2] [UP] to enter and mark the entry() function	$\cos(a) + \boxed{\text{entry}(2)}$
Push [F1] to evaluate entry(2). The contents of the history display at level 2 are pasted.	$\cos(a) + \boxed{\sin(x \cdot y \cdot z)}$

You can also use the key combination [DIAMOND] [ENTER] to evaluate expressions in Approximate mode. This is useful when the calculator is in Auto or Exact modes, and you want the approximate result. If the calculator is in Approximate mode, then [F1] evaluates the expression in Approximate mode. This example shows some variations on using [F1] and [DIAMOND] [ENTER].

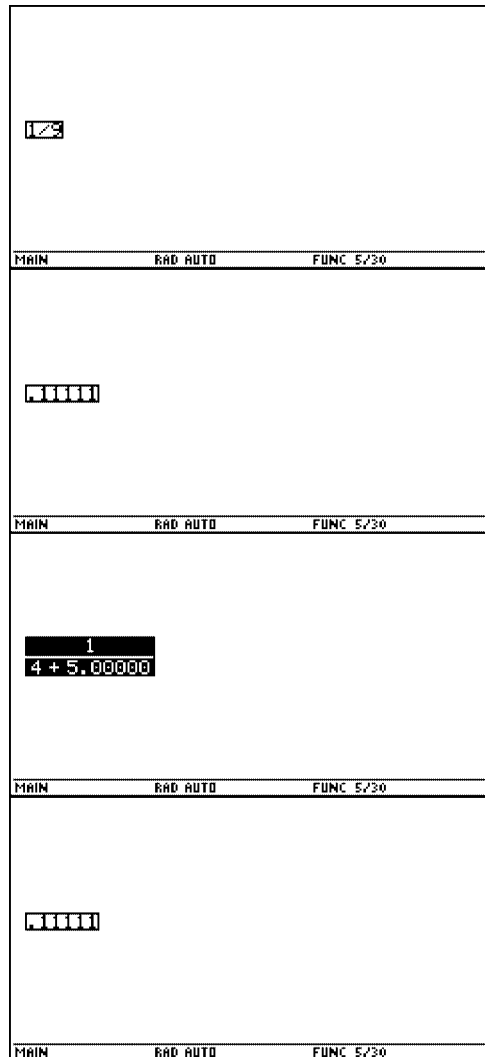
Use the [MODE] key to set the mode to Auto, then start EQW. Press [1] [/] [4] [+] [5] to enter the expression, then press [2nd] [UP] to mark the expression.

$\boxed{\frac{1}{4+5}}$
MAIN RAD AUTO FUNC 5/30



Press [F1] to evaluate the expression.

The exact result is shown, because the calculator is set to Auto mode.



Press [INS] to undo the evaluation. The expression is marked. Press [DIAMOND] [ENTER] to evaluate the expression in Approximate mode. The approximate result is shown.

Press [DIAMOND] [CLEAR] to clear the EQW screen. To enter the expression, press [1] [/] [4] [+] [5] [.] . Press [2nd] [UP] to mark the expression.

Note that the '5' in the expression is shown in the current numeric display mode, since we included the decimal point.

Press [F1] to evaluate the expression. The approximate result is shown, since the expression included a numeric constant with a decimal point.

### Applying functions with [F2] - [F7]

You can apply some built-in TI-89/92+ functions without typing in the function name. The functions are applied with the function keys [F2] through [F7], and they are:

[F2]	<i>factor()</i>
[F3]	<i>expand()</i>
[F4]	<i>tCollect()</i>
[F5]	<i>tExpand()</i>
[F6]	<i>comDenom()</i>
[F7]	<i>propFrac()</i>

Refer to the *TI-89/TI-92+ User's Guide* for descriptions of these functions. The functions are applied by marking the expression, then pressing the function key to apply the function to the marked expression.

This example shows how to use *factor()* and *expand()* with the function keys.

Start EQW. Press [x] [+] [1] [UP] [\*] [x] [+] [2] to enter the expression shown.

$$(x + 1) \cdot (x + 2)$$

MAIN RAD EXACT FUNC 4/30

Press [UP] [UP] to mark the expression

$$(x + 1) \cdot (x + 2)$$

MAIN RAD EXACT FUNC 4/30

Press [F3] to apply *expand()* to the marked expression. Note that the original expression has been expanded.

$$x^2 + 3 \cdot x + 2$$

MAIN RAD EXACT FUNC 4/30

Press [F2] to factor the expression. The expression has been factored.

$$(x + 1) \cdot (x + 2)$$

MAIN RAD EXACT FUNC 4/30

Note that the [MODE] key displays the key functions for [F1] -[F8].

## Using forms

A form is an expression with place-holders instead of elements. Forms can save time if you frequently use an expression in which some of the values change. You can save the form, then recall it later into EQW and fill in the place-holders. The steps to create and use a form in EQW are:

1. Create the expression in EQW, but do not put elements in the place-holders.
2. Mark the entire expression (you can use [2nd] [UP]).
3. Press ["] to convert the expression to a string.
4. Exit EQW with [ENTER]
5. Store the string in the entry line to a variable.
6. When you next want to edit the form equation, start EQW with the form variable name.
7. Fill in the place-holders as needed.

This example shows how to make a form to create a sum, then use the form.

Push [Σ] [1] [/] [k] [^] [2] to create the sum argument	$\sum_{k=1}^{\infty} \left( \frac{1}{k^2} \right)$
Push [RIGHT] to move the cursor, then press [k]	$\sum_{k=1}^{\infty} \left( \frac{1}{k^2} \right)$
Push [2nd] [UP] to mark the entire expression, then press ["] to convert it to a string	"Σ(1/k^2,k,1,∞)"
Push [ENTER] to quit EQW and return the expression to the command line. Push [STO] to save the form to the variable <i>sumform</i>	<b>"Σ(1/k^2,k,1,∞)"→sumform</b> MAIN RAD EXACT FUNC 0/30
We want to create a sum with limits of 4 and 6. First, start EQW with the <i>sumform</i> variable	<b>eqw(sumform)</b> MAIN RAD EXACT FUNC 0/30
EQW starts with the saved form. Note the place-holders are present.	$\sum_{k=1}^{\infty} \left( \frac{1}{k^2} \right)$
Push [RIGHT] twice, to move the cursor to the first place-holder, then push [4]	$\sum_{k=4}^{\infty} \left( \frac{1}{k^2} \right)$
Push [RIGHT] once, to move the cursor to the next place-holder, then press [6]. The desired sum is complete.	$\sum_{k=4}^6 \left( \frac{1}{k^2} \right)$

You may also save the form to a variable within EQW. After converting the form to a string with ["], press [STO], enter a variable name in the placeholder, mark the expression, then press [F1] to evaluate the expression, which stores the form to the variable name.

With this simple program you can recall forms within EQW.

```
eqwprgmr()  
Prgm  
@Recall form  
Local varname  
Dialog  
Title "Recall Form"
```

```

Request  "Enter variable name",varname
EndDialog
If  ok=1:#varname→main\eqwcom
EndPrgm

```

Because this program is named *eqwprgmr()* and is saved in the *main\* folder, it is executed when [RCL] is pressed. This dialog box is shown:



The screenshot above shows that I have already entered the *sumform* variable name. When I press [ENTER], the sum form is inserted at the current placeholder.

Since forms are text strings, you can create them in the entry line or in a program. The placeholder character code is 255, which is displayed as  $\tilde{y}$ . This character can be created with *char(255)*. For example, the string to create a simple form for the *sin()* function is

```
"sin("&char(255)&")"
```

Like any other character, the placeholder character can be saved as a variable. If the variable name is prefixed with the underscore character "\_", the variable can be accessed from any folder. To save the placeholder character to a variable *\_ph*, use

```
char(255)→_ph
```

To create the sine form from the last example, use

```
"sin("&_ph&")"
```

As another example, this expression creates a form with two placeholders. It is awkward to create the *augment(;*) function with the ";" in EQW, but a form is easily created:

```
"augment("&_ph&";"&_ph&")"
```

### ***Other features and tips***

- Use [DIAMOND] [ENTER] to insert the *approx()* function at the cursor. If an expression is marked, [DIAMOND] [ENTER] does not insert *approx()*, but instead evaluates the expression in Approx mode.
- If the expression includes place-holders, you can use [ENTER] to move the cursor to the next place-holder.
- Use ["] to convert a marked expression to a string.
- Use [&] to concatenate two or more strings.
- You can enter any characters in a string; there are no restrictions. For example, pressing [ENTER] during string character entry will insert the 'left-hook' ENTER character.
- You can create an empty string in EQW. Press ["] to create a string, then press [BACKSPACE] to delete the place-holder. The expression "" is an empty string. The vertical bar is just the cursor, which shows where the next string character would be inserted.
- Use the Greek menu in the [CHAR] menu to type Greek characters.
- Use the Math menu in the [CHAR] menu to insert the natural logarithm base *e*, without any exponent.
- You can use [STO] to store expressions to variables. Use [2nd] [UP] to mark the entire expression, if necessary, then press [STO] to insert the → operator, and enter the variable name. If you want to actually perform the 'store' operation in EQW, press [2nd] [UP] [F1] to mark and evaluate the expression.
- EQW will only use parenthesis when they are mathematically necessary. When the expression is returned to the command line, the CAS may insert additional parentheses. For example, the expression  $(1/a)*(1/b)*(1/c)$  will be entered and shown in EQW without parentheses. However, the expression is returned to the command line as  $1/a*(1/b)*(1/c)$

---

## Calling EQW from TI Basic programs

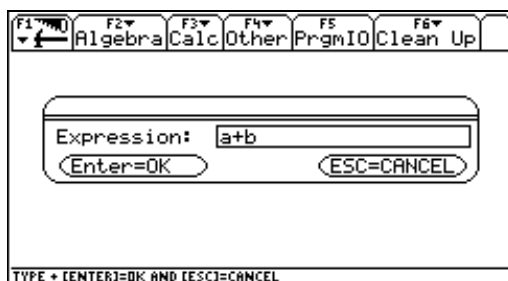
---

EQW may be called from a TI Basic program. You may also use the expression returned from EQW in your program. You can also just use EQW to display an expression in 'pretty-print' format while your program is running. This sample program *eqwtest()* shows how.

```
eqwtest()  
Prgm  
©Sample program for EQW and TI Basic  
©22feb01/dburkett@infinet.com  
  
local x,y  
  
© Example 1  
© Start EQW with a clean screen  
eqww\eqw()  
request "Expression",y  
  
© Example 2  
© Start EQW with an expression and  
© save result as a string in 'y'  
eqww\eqw(sin(p^q)-cos(r^s))  
request "Expression",y  
  
© Example 3  
© Start EQW with an expression as a string in x  
© and save the result as an expression in y  
"√(b^2-4*a*c)"→x  
eqww\eqw(x)  
request "Expression",y  
expr(y)→y  
  
© Example 4  
© Start EQW with a clean screen, and  
© return the expression to the entry line  
© when the program exits  
eqww\eqw()  
  
EndPrgm
```

In these examples, EQW is installed in a folder *\eqww*, so all the *eqw()* calls include this folder specification. This ensures that *eqwtest()* runs without error from any folder.

In example 1, we start EQW with a clean screen, that is, with no expression. Suppose we enter the expression  $a+b$  in EQW, and press [ENTER]. The *request* command 'catches' the output expression from EQW and saves it as a string in variable *y*. The *request* command displays a dialog box, as usual:



You must press [ENTER] twice to accept the value and resume program execution. If you exit EQW by pressing [ESC] instead of [ENTER], then no expression is returned, but the dialog box is still shown. You can also use the *Dialog ... EndDialog* structure for a more elaborate dialog box. When the dialog box is closed with [ENTER], the variable *y* contains the EQW result as a string, that is, "a+b".

Example 2 shows that you can call EQW with an expression. As usual, EQW displays the expression. Note that the variables used in the expression must either be global variable, or initialized local variables, or an *Undefined Variable* error dialog box will be displayed.

Example 3 shows two features. EQW is started with a string, to avoid automatic simplification, and the result in *y* is converted to an expression with *expr()*.

Example 4 shows that EQW will return its expression to the entry line, if you do not 'catch' its results with *Request*.

It is also possible to get the expression returned by EQW without using *Request* and displaying the dialog box. You use the functions *rcldclip()* and *stoclip()*, which E.W. has written. *rcldclip()* recalls the current clipboard expression as a string and saves it to the variable name, specified as a string in the function argument. *stoclip()* stores its string argument to the clipboard. For example,

<code>rcldclip("varname")</code>	stores the clipboard contents to variable <i>varname</i>
<code>stoclip("a+b")</code>	stores the string "a+b" to the clipboard

Note that the arguments to both functions are strings. The variable name argument for *rcldclip()* must be passed as a string.

There are different versions of *stoclip()* and *rcldclip()* for the TI-89 and the TI-92+; make sure you use the right one for your calculator.

This demonstration program shows how to use *stoclip()* and *rcldclip()* to get EQW results in your programs. *eqw()*, *rcldclip()*, and *stoclip()* are all installed in folder *leqww*, so the function calls specify the folder.

```
eqwtest2()
Prgm
©Demonstration program for stoclip() and rcldclip() by E.W.
©24feb00/dburkett@infinet.com

Local oldclip,eqwclip,eqwexp

©Save current clipboard contents, only
©if you will need to restore them later
eqww\rcldclip("oldclip")

©Clear clipboard and start EQW
eqww\stoclip("")
eqww\eqw()

©Retrieve and save expression from EQW
eqww\rcldclip("eqwclip")

©Test for value returned by EQW
If eqwclip≠"" Then
  expr(eqwclip)→eqwexp
  dialog
```

```

        text "EQW expression saved"
        text eqwclip
    enddlog
else
    dialog
        text "No expression returned by EQW"
    enddlog
EndIf

@Restore clipboard if needed
eqww\stoclip(oldclip)

EndPrgm

```

The demonstration program does not really do much with the expression returned by EQW. The *If ... EndIf* test shows that you can take different actions depending on whether or not an expression is returned by EQW. If EQW is exited by pressing [ESC], then an empty string "" is returned.



---

## EQW Extensions

---

TI Basic programs can be executed while EQW is running. With this feature you can extend and customize EQW, without exceeding the 24K ASM limit imposed by Texas Instruments.

Eleven programs can be called from EQW. They have specific names and must all be installed in the *main\* folder. I call these *EQW extensions*. One EQW extension is named *eqwuser()*, and it is executed when [APPS] is pressed in EQW. Nine more extensions are named *eqwprgm1()* through *eqwprgm9()*, and are executed with [DIAMOND] [1] through [DIAMOND] [9].

Finally, the extension named *eqwprgmr()* is executed when [RCL] is pressed. This extension and key assignment is useful for recalling forms, but may perform any other function. See *Using forms* for an example.

Several sample EQW extension programs are included in the distribution file. Do not open these in GraphLink before sending them, otherwise they may not work properly. Instead, just use the Link, Send menu items to send the programs to the calculator. The TI GraphLink software is the source of the problem, not EQW or the EQW extensions. When the files are opened, GraphLink does not recognize some characters and converts them to other characters. For example, the filled square symbol (character code 16) will be converted to the question mark symbol "?". In addition, strings involving repeated double-quote characters (character code 34) may cause syntax errors.

EQW communicates with the EQW extensions through two variables, *main\eqwrun* and *main\eqwcom*. *eqwrun* is a boolean variable which is set to *True* when EQW is running and can accept input through *eqwcom*. EQW takes the expression from *eqwcom* and inserts it at the placeholder in EQW.

You may want to test and debug your EQW extensions from the entry line, instead of calling them from EQW. If your extension has a syntax error or other bug and you run it from EQW, the error message dialog box is displayed, but you will not be able to open the program editor at the error location.

### E.W.'s sample *eqwuser()* extension

Some typical extensions which take good advantage of EQW programs are custom toolbar menus and pop-up menus. E.W. has written a sample *eqwuser()* extension which creates several useful toolbar menus. This program is distributed with the EQW package, and is described below. E.W. has also written a sample pop-up menu extension, *eqwprgm2()*, which is described in the *More Examples* section under *Use a pop-up menu of history display answers in EQW*.

The code on the next page shows E.W.'s sample *eqwuser()* extension. I have reformatted the code to fit a single page. The version distributed with EQW looks different but is functionally identical. Note:

1. The program sets up four toolbar menus. The menus are constructed with the usual *ToolBar ... EndTBar* structure.
2. The program *ins()*, which is defined within *eqwuser()*, does the work of sending the chosen text string to EQW if EQW can accept input. Otherwise, *ins()* uses *sendtext()* to send the menu item to the home screen entry line. Note that *sendtext()* must be installed in the *\main* folder.
3. *eqwuser()* can perform tasks beyond just sending text to EQW. For example, some of the Other menu items clear the Graph screen and show the Program I/O screen.
4. The toolbar menus can be used to streamline EQW procedures which are awkward or require a lot of keystrokes in EQW. For example, the Vector menu lets you enter templates for any type of non-rectangular vector.

### **Demonstration eqwuser() extension**

```

eqwuser()
Prgm
Local ins

© Define program to send string to EQW
Define ins(txt)=Prgm
If when(main\eqwrun,true,false,false)
Then
    txt→main\eqwcom
Else
    main\sendtext(txt)
EndIf
EndPrgm

© Display custom menu toolbar
Toolbar
Title "Vector"
Item "[□,△□] polar",v1
Item "[□;△□] polar",v2
Item "[□,△□,□] cylind",v3
Item "[□;△□;□] cylind",v4
Item "[□,△□,△□] sphere",v5
Item "[□;△□;△□] sphere",v6

Title "Algebra"
Item "solve(",a1
Item "factor(",a2
Item "expand(",a3
Item "zeros(",a4
Item "comDenom(",a5
Item "propFrac(",a6
Item "nSolve(",a7

Title "Calc"
Item "d( differentiate",cc1
Item "f( integrate",cc2
Item "limit(",cc3
Item "Σ(",cc4
Item "Π(",cc5
Item "fMin(",cc6
Item "fMax(",cc7
Item "arcLen(",cc8
Item "taylor(",cc9
Item "nDeriv(",cc10
Item "nInt(",cc11
Item "deSolve(",cc12

Title "Other"
Item "Graph",o1
Item "Table",o2
Item "Clr Graph",o3
Item "Clr I0",o4
Item "Clr Table",o5
Item "Trace",o6
Item "Show PrgI0",o7
Item "Show Graph",o8

EndTBar
Return

© Algebra menu items
Lb1 a1:ins("solve("):Return
Lb1 a2:ins("factor("):Return
Lb1 a3:ins("expand("):Return
Lb1 a4:ins("zeros("):Return
Lb1 a5:ins("comDenom("):Return
Lb1 a6:ins("propFrac("):Return
Lb1 a7:ins("nSolve("):Return

© Calc(ulus) menu items
Lb1 cc1:ins("d("):Return
Lb1 cc2:ins("f("):Return
Lb1 cc3:ins("limit("):Return
Lb1 cc4:ins("Σ("):Return
Lb1 cc5:ins("Π("):Return
Lb1 cc6:ins("fMin("):Return
Lb1 cc7:ins("fMax("):Return
Lb1 cc8:ins("arcLen("):Return
Lb1 cc9:ins("taylor("):Return
Lb1 cc10:ins("nDeriv("):Return
Lb1 cc11:ins("nInt("):Return
Lb1 cc12:ins("deSolve("):Return

© Vector menu items
Lb1 k1:DispG:Trace:DispHome:Return
Lb1 v1:ins("[ȳ,△ȳ]"):Return
Lb1 v2:ins("[ȳ;△ȳ]"):Return
Lb1 v3:ins("[ȳ,△ȳ,ȳ]"):Return
Lb1 v4:ins("[ȳ;△ȳ;ȳ]"):Return
Lb1 v5:ins("[ȳ,△ȳ,△ȳ]"):Return
Lb1 v6:ins("[ȳ;△ȳ;△ȳ]"):Return

© Other menu items
Lb1 o1:ins("Graph "):Return
Lb1 o2:ins("Table "):Return
Lb1 o3:ClrGraph:Return
Lb1 o4:ClrI0:Return
Lb1 o5:ClrTable:Return
Lb1 o6
    Trace
    ins(["&string(xc)&","&string(yc)&"])
    DispHome
    Return
Lb1 o7:Disp:Pause:DispHome:Return
Lb1 o8:DispG:Pause:DispHome:Return

EndPrgm

```

This screen shot shows a typical menu from *eqwuser()*.



### Using [DIAMOND] [F1] ... [DIAMOND] [F5] with *eqwuser()*

EQW can send keycodes for five function keys to *eqwuser()*. The keys are [DIAMOND] [F1] through [DIAMOND] [F5]. In other words, pressing one of these function keys will start *eqwuser()* and send the keycode. This feature means that you can execute up to five distinct features in *eqwuser()*, without first pressing [APPS]. Note, though, that the keycode which is sent is the code for the *function* key, *not* the code for the function key modified by the [DIAMOND] key. For example, if [DIAMOND] [F1] is pressed, then the keycode 268 is sent to *eqwuser()*, not 8460. For these five keys, the keycodes are the same for the TI-89 and the TI-92+, so the same program may be used for both calculators. For reference, these are the decimal keycodes for the five function keys:

[F1] 268                      [F1] 269                      [F2] 270                      [F3] 271                      [F4] 272

This a simple *eqwuser()* extension which toggles the Angle mode with [DIAMOND] [F2], and toggles the Exact/Approx/Auto mode with [DIAMOND] [F1].

```

eqwuser()
Prgm
@Toggle exact/approx/auto with [DIAMOND][F1], and rad/deg with [DIAMOND][F2]
@15oct01/dburkett@infinet.com

local key

@Get the keycode passed by EQW
getkey()→key

if key=268 then

@Toggle Exact/Approx/Auto mode w/[F1]
setmode("14",string(exact(mod(expr(getmode("14")),3)+1)))

elseif key=269 then

@Toggle DEG/RAD angle mode w/[F2]
setmode("3",string(exact(mod(expr(getmode("3")),2)+1)))

endif

EndPrgm

```

If the current Exact/Approx/Auto mode is Exact, then repeatedly pressing [DIAMOND] [F1] toggles the mode through each setting:

Exact -> Approx -> Auto -> Exact -> ...

Similarly, pressing [DIAMOND] [F2] toggles the angle mode:

DEG -> RAD -> DEG -> ...

No output display is necessary, since the current modes are shown in the status line at the bottom of the calculator display.

This program uses the numeric string codes for *getMode()* and *setMode()*, for two reasons. First, the program will work correctly regardless of the language localization of the calculator. Second, the numerical code string returned by *getMode()* can be converted to a number with *expr()*, which means that we can use the *mod()* function to switch to the next setting.

The expressions which change the modes are a bit convoluted, but the basic idea is to get the current mode as a number, use the *mod()* (modulo) function to limit the mode number, then add 1 to find the next mode number. The *exact()* function removes decimal point and the base-ten exponent characters, if any, from the calculated mode number. The calculated mode number contains these characters if the mode is Approx, but *setMode()* will not accept these characters in the argument. Finally *string()* converts the mode number to a string argument of *setMode()*. For example, when the Auto/Exact/Approx mode is toggled, we want the mode numbers to cycle through 1, 2, 3, 1, ... This table shows how the next mode number is found for each mode.

Mode	Mode Number	mod(mode,3)	mod(mode,3)+1
Auto	1	1	2
Exact	2	2	3
Approx	3	0	1

### Pop-up menus in *eqwuser()*

We now have extensions which can change the modes for angle and Auto/Exact/Approx, and it would also be useful to change some other mode settings within EQW. In particular, the Exponent, Complex and Vector format settings will affect calculations and displays. The following extension, called *eqwprgm3()*, uses a pop-up menu to display and change these mode settings.

```
eqwprgm3()
Prgm
@Toggle exponent, complex & vector modes
©11oct01/dburkett@infinet.com
local v,i,l,k

Ø→v   ©Initialize the pop-up menu variable

©Initialize the pop-up menu list
{"NORM exp","SCI","ENG","REAL cmplx","RECT","POL","RECT vect","CYL","SPH "}→l

©Change menu list elements to show the mode status

exact(expr(getmode("4")))->i           ©Exponent mode items
for k,1,3                               ©... loop through each item
  when(i=k,char(18)," ")&l[k]→l[k]     ©... prepend " ", or "•" if item set
endfor

exact(expr(getmode("5")))->i           ©Complex mode items
for k,4,6
  when(i+3=k,char(18)," ")&l[k]→l[k]
endfor
```

```

exact(expr(getmode("6")))->i          @Vector mode items
for k,7,9
  when(i+6=k,char(18)," ")&l[k]->l[k]
endfor

popup l,v          @Display the menu

if v≠0 then          @Change mode if menu item selected, else exit
setmode(string(exact(intdiv(v-1,3)+4)),string(exact(mod(v-1,3)+1)))
endif

EndPrgm

```

When this program is executed in EQW by pressing [DIAMOND] [3], the following pop-up menu is shown:



The pop-up menu shows all three possible settings for each of the three modes. The ► character indicates the current mode setting. For the example above, the Exponential format is Normal, the Complex format is Rectangular, and the Vector format is Cylindrical.

To return to EQW without changing the settings, press [ESC]. To change a setting, use [UP] and [DOWN] to highlight the setting and press [ENTER]. Another way to change a setting is to press the number key of the setting. For example, to change the Complex mode to Polar, press [6]. After the selection is made, the mode setting is changed and the pop-up menu disappears.

Although I have shown the keyboard programs to change the Angle and other mode settings as individual 'hotkey' programs, you can also incorporate them into *eqwuser()*, and activate them with the [DIAMOND] [Fx] hotkeys, where [Fx] is [F1] through [F5].

## Define EQW extensions as program name strings

There is still another method to execute your extensions from EQW. Simply store the program folder and name, as a string, to one of the the extension programs. For example, if your program is called *myprgm* and is located in the folder *main\*, and you want to execute the program with [DIAMOND] [4], then use

```
"main\myprgm()"→main\eqwprgm4
```

## A comprehensive *eqwuser()* example

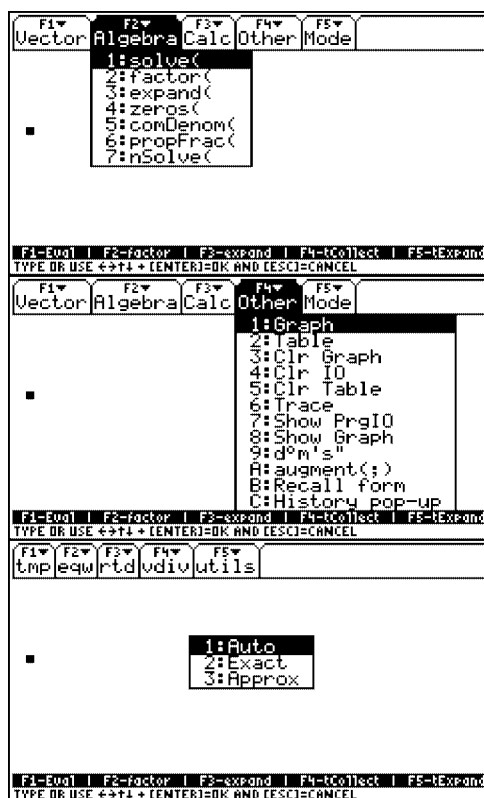
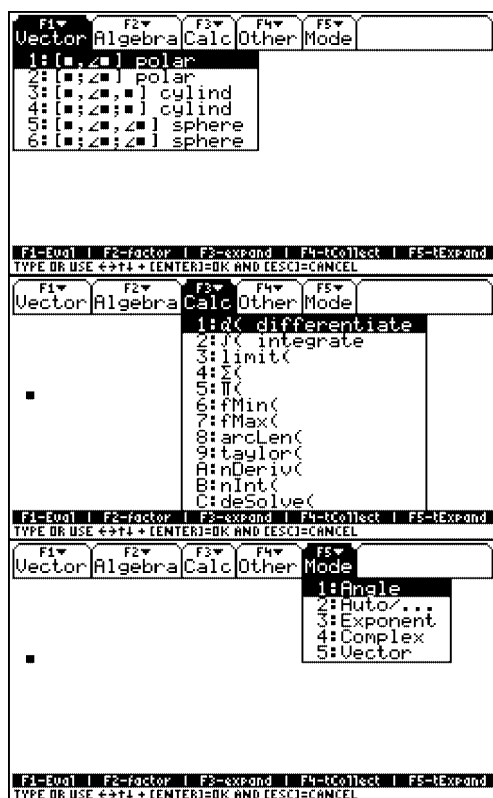
As a final example, the following *eqwuser()* extension uses drop-down and pop-up menus to implement these features:

- All of the original menu items from E.W.'s *eqwuser()* extension.

- d°m's" form
- *augment(;)* form
- Recall form feature
- History pop-up menu
- Mode settings for Angle, Auto/Exact, Exponent format, Complex format and Vector format.

This version of *eqwuser()* is distributed as *eqwuserb.9xp* in the distribution package.

These screen shots show the menu items. The last screen shot shows the pop-up menu to change the Auto/Exact mode setting.



This version of *eqwuser()* does not use the [DIAMOND] [F1] ... [DIAMOND] [F5] keys, so you can use those keys to add more of your own features.

```

eqwuser()
Prgm
@EQW toolbars w/mode toggles
@Original version by E.W.
@Modified 24oct01
@by dburkett@infinet.com
@This version of eqwuser() includes the
original E.W. toolbar menus, dms and
augment(;) forms, recall form, popup history
and mode settings

Local ins,i,var,lst,n,mnu,fs

Define ins(txt)=Prgm
  If when(main\eqwrun,true,false,false) Then
    txt→main\eqwcom
  Else
    main\sendtext(txt)
  EndIf
EndPrgm

char(16)→fs @Define filled-square character

Toolbar

Title "Vector"
Item "["&fs&"",∠"&fs&"] polar",v1
Item "["&fs&";∠"&fs&"] polar",v2
Item "["&fs&"",∠"&fs&"", "&fs&"] cylind",v3
Item "["&fs&";∠"&fs&"; "&fs&"] cylind",v4
Item "["&fs&"",∠"&fs&"",∠"&fs&"] sphere",v5
Item "["&fs&";∠"&fs&";∠"&fs&"] sphere",v6

Title "Algebra"
Item "solve(",a1
Item "factor(",a2
Item "expand(",a3
Item "zeros(",a4
Item "comDenom(",a5
Item "propFrac(",a6
Item "nSolve(",a7

Title "Calc"
Item "d( differentiate",cc1
Item "f( integrate",cc2
Item "limit(",cc3
Item "Σ(",cc4
Item "Π(",cc5
Item "fMin(",cc6
Item "fMax(",cc7
Item "arcLen(",cc8
Item "taylor(",cc9
Item "nDeriv(",cc10
Item "nInt(",cc11
Item "deSolve(",cc12

Title "Other"
Item "Graph",o1
Item "Table",o2
Item "Clr Graph",o3
Item "Clr IO",o4
Item "Clr Table",o5
Item "Trace",o6
Item "Show PrgIO",o7
Item "Show Graph",o8
Item "d°m's""",o9
Item "augment(;)",o10
Item "Recall form",o11
Item "History pop-up",o12

Title "Mode"
Item "Angle",m1
Item "Auto/...",m2
Item "Exponent",m3
Item "Complex",m4
Item "Vector",m5

EndTBar

Return

@Algebra menu items
Lb1 a1:ins("solve("):Return
Lb1 a2:ins("factor("):Return
Lb1 a3:ins("expand("):Return
Lb1 a4:ins("zeros("):Return
Lb1 a5:ins("comDenom("):Return
Lb1 a6:ins("propFrac("):Return
Lb1 a7:ins("nSolve("):Return

@Calculus menu items
Lb1 cc1:ins("d("):Return
Lb1 cc2:ins("f("):Return
Lb1 cc3:ins("limit("):Return
Lb1 cc4:ins("Σ("):Return
Lb1 cc5:ins("Π("):Return
Lb1 cc6:ins("fMin("):Return
Lb1 cc7:ins("fMax("):Return
Lb1 cc8:ins("arcLen("):Return
Lb1 cc9:ins("taylor("):Return
Lb1 cc10:ins("nDeriv("):Return
Lb1 cc11:ins("nInt("):Return
Lb1 cc12:ins("deSolve("):Return

@Vector menu items
Lb1 v1:ins("[ÿ,∠ÿ"):Return
Lb1 v2:ins("[ÿ;∠ÿ"):Return
Lb1 v3:ins("[ÿ,∠ÿ,ÿ"):Return
Lb1 v4:ins("[ÿ;∠ÿ;ÿ"):Return
Lb1 v5:ins("[ÿ,∠ÿ,∠ÿ"):Return
Lb1 v6:ins("[ÿ;∠ÿ;∠ÿ"):Return

@Other menu items
Lb1 o1:ins("Graph "):Return
Lb1 o2:ins("Table "):Return
Lb1 o3:ClrGraph:Return
Lb1 o4:ClrIO:Return
Lb1 o5:ClrTable:Return

Lb1 o6
Trace
ins(["&string(xc)&","&string(yc)&"]")
DispHome
Return

Lb1 o7:Disp:Pause:DispHome:Return
Lb1 o8:DispG:Pause:DispHome:Return
Lb1 o9:ins("ÿ°ÿ'ÿ"""):Return
Lb1 o10:ins("augment(ÿ;ÿ)"):Return

Lb1 o11
Dialog
  Title "Recall Form"
  Request "Enter variable name",var
EndDlog
If ok=1:#var→main\eqwcom
return

```

```

Lb1 o12
{}->lst
{}->mnu
Ø->var
1->n
for i,1,2Ø
  Try
  string(expr("ans("&string(exact(i))&")"))->lst[
n]
  left(lst[n],15)->mnu[n]
  n+1->n
  else
  clrerr
  endtry
endfor

if dim(mnu)>Ø then
popup mnu,var
if var>Ø
  lst[var]->main\eqwcom
else
  dialog
  title "HISTORY POP-UP"
  text "No history elements"
  enddlog
endif
return

Lb1 m1
Ø->var
popup {"RAD", "DEG"},var
if var=1 then
  setmode("3", "1")
elseif var=2 then
  setmode("3", "2")
endif
return

Lb1 m2
Ø->var
popup {"Auto", "Exact", "Approx"},var
if var=1 then
  setmode("14", "1")
elseif var=2 then
  setmode("14", "2")
elseif var=3 then
  setmode("14", "3")
endif
return

```

```

Lb1 m3
Ø->var
popup {"Normal", "Sci", "Eng"},var
if var=1 then
  setmode("4", "1")
elseif var=2 then
  setmode("4", "2")
elseif var=3 then
  setmode("4", "3")
endif
return

Lb1 m4
Ø->var
popup {"Real", "Rect", "Polar"},var
if var=1 then
  setmode("5", "1")
elseif var=2 then
  setmode("5", "2")
elseif var=3 then
  setmode("5", "3")
endif
return

Lb1 m5
Ø->var
popup {"Rect", "Cylind", "Sphere"},var
if var=1 then
  setmode("6", "1")
elseif var=2 then
  setmode("6", "2")
elseif var=3 then
  setmode("6", "3")
endif
return

EndPrgm

```



### ***TI Menus and Equation Library***

Steve Chism (SteveChism@hotmail.com) has written a Windows application, *TI Menus*, which automates the process of creating toolbar extensions for EQW. You can download *TI Menus* at

<http://www.warp2k.com/eqlib/tbmaker.htm>

*TI Menus* automatically creates a TI Basic program which implements the toolbar menus you specify, and can also create toolbars for use in the native TI-89/92+ operating environment. Steve gives these features for *TI Menus*:

Types of toolbars supported by *TI Menus*:

- Standard TI Toolbars
- EQW Toolbars - too be used directly within EQW
- EQLib Toolbars - too be used within EQLib

Features:

- Toolbar maker for the TI-89 and TI-92 Calculator
- User-friendly Windows environment
- Compiles automatically
- Make, save, load and trade dozens of toolbars
- Edit, Move and Delete menus and items
- Master Equation List which enables the user to move 100's of commands and equations between menus
- Compiles toolbars for use with EQLib and EQW enhanced toolbars
- Verifies lengths of title, item and description fields to prevent error
- Just copy and paste in TI Graph Link

Nevin McChesney (nevin@mcchesney.com) has written a sophisticated equation library program, *Equation Library*, which works with EQW. For more information, see

<http://www.warp2k.com/eqlib/description.htm>

Nevin says that

*"Equation Library was written in order to organize and make equations used on a regular basis more available. Equation library is a viewer for libraries that you create or download."*

Libraries for *Equation Library* can be created with *TI Menus*.

---

## More examples

---

The examples in this section demonstrate creating and editing various expressions in EQW. Some examples also show that most home screen calculations can also be performed within EQW.

### ***Limitations on using ans() and entry()***

*ans()* and *entry()* are used to retrieve results and entries from the home screen. You may freely use these expressions in EQW, but they are not true built-in functions, so they cannot be evaluated (with [F1] or [DIAMOND] [ENTER]) if they are combined with other subexpressions. This is not a limitation of EQW, it is a limitation of the calculator operating system. You may, however, evaluate *ans()* or *entry()* expressions by selecting only the expression, or you may return the entire expression to the entry line, with [ENTER], then press [ENTER] again to evaluate it.

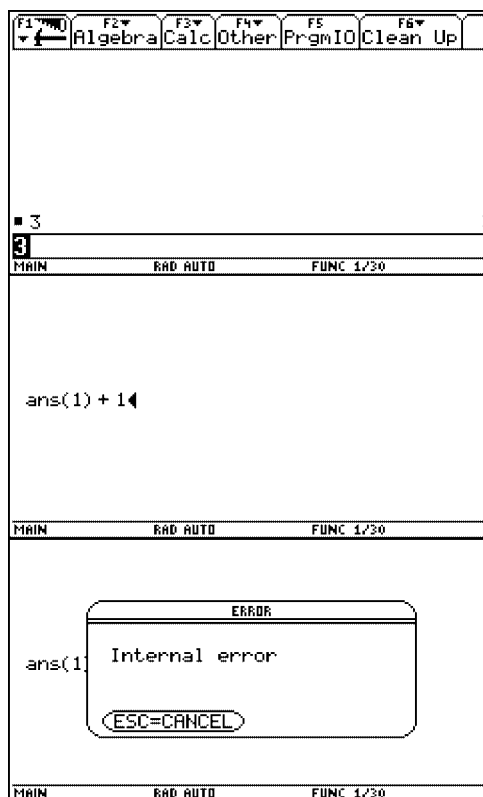
If you try to evaluate a compound expression including *ans()* or *entry()*, a dialog box is shown with the message *Internal Error*.

This example shows some operations with *ans()* and *enter()*.

At the home screen entry line, press [3] [ENTER]. 3 is entered on the first history level.

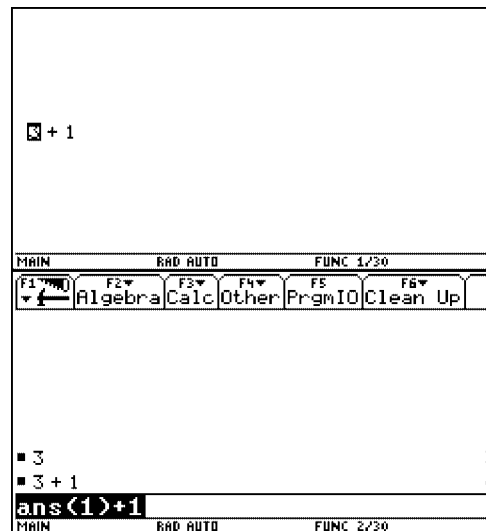
Start EQW. Press [ANS] [1] [UP] [+] [1] to enter the expression.

Press [2nd] [UP] to mark the expression, then press [F1] to evaluate it. The error message is shown, because we cannot evaluate a compound expression with *ans()*.



Press [ESC] to clear the error message. Press [DOWN] to mark the *ans()* expression, then press [F1]. The expression is correctly evaluated.

Press [INS] to undo the evaluation. Press [ENTER] to return the expression to the home screen entry line. Press [UP] [BACKSPACE] to delete the history entry with which you started EQW. Press [DOWN] to move the cursor back to the entry line. Press [ENTER] to evaluate the returned expression. The expression is evaluated correctly.



## Entering and editing matrices

The TI-89/TI-92+ have a built-in matrix editor, but you can use also EQW to create and edit a matrix. There is no advantage to using EQW for simple or numeric matrices, but it is useful if you create matrices with complicated symbolic elements.

You can cut and copy complete matrices. You can cut or copy a matrix row, but you cannot paste it as a row: it will be pasted as a string.

These keys are used to create and edit matrices:

### Keys for matrix editing operations

Create a new matrix at the cursor location	[ [ ] ]
Insert a new column, when a matrix element or row is marked. New columns can only be added at the right of the matrix.	[ , ]
Insert a new row, when the entire matrix is marked. New rows may only be added at the bottom of the matrix.	[ , ]
Insert a new row, when a matrix element or row is selected or marked. New rows may only be added at the bottom of the matrix.	[ ; ]
Move the outline cursor to the next or previous element	[LEFT] or [RIGHT]
Mark a row, when an element is selected with the outline cursor.	[UP]
Mark the preceding row, when a row is marked. If the first row is already marked, [LEFT] marks the entire matrix.	[LEFT]
Mark the next row, when a row is marked. If the last row is already marked, [RIGHT] puts the outline cursor at the first element of the row.	[RIGHT]
Select an element if a row is marked.	[DOWN]
Mark the entire matrix if an element is selected	[UP] [UP]
Mark the entire matrix if a row is marked	[UP]
Move the cursor to the first matrix element	Press [UP] repeatedly, until the entire matrix is marked, then press [DOWN]
Delete a row	Mark the row with [UP], then press [BACKSPACE]
Delete a column	Move the outline cursor to one of the column elements, push [BACKSPACE] to delete the element, then push [BACKSPACE] to delete the column.
Edit a matrix element	Move the cursor to the element. If the element is marked, push [DOWN] so that the outline cursor appears.

The following example shows how to create this matrix:

$$\begin{bmatrix} a & b & c \\ d & e & f \\ g & h & i \end{bmatrix}$$

### Example: Create a matrix

Start EQW.

Press [ ] to create a matrix

Press [,] [,] to create two more columns.

To add another row, press [2nd] [UP] to mark the entire matrix, then press [,]. To add the last row, press [2nd] [UP] [,] again.

The matrix has all the placeholders, so we can start entering the elements.

Press [2nd] [UP] to mark the entire matrix, then press [DOWN] [DOWN] to select the first placeholder.

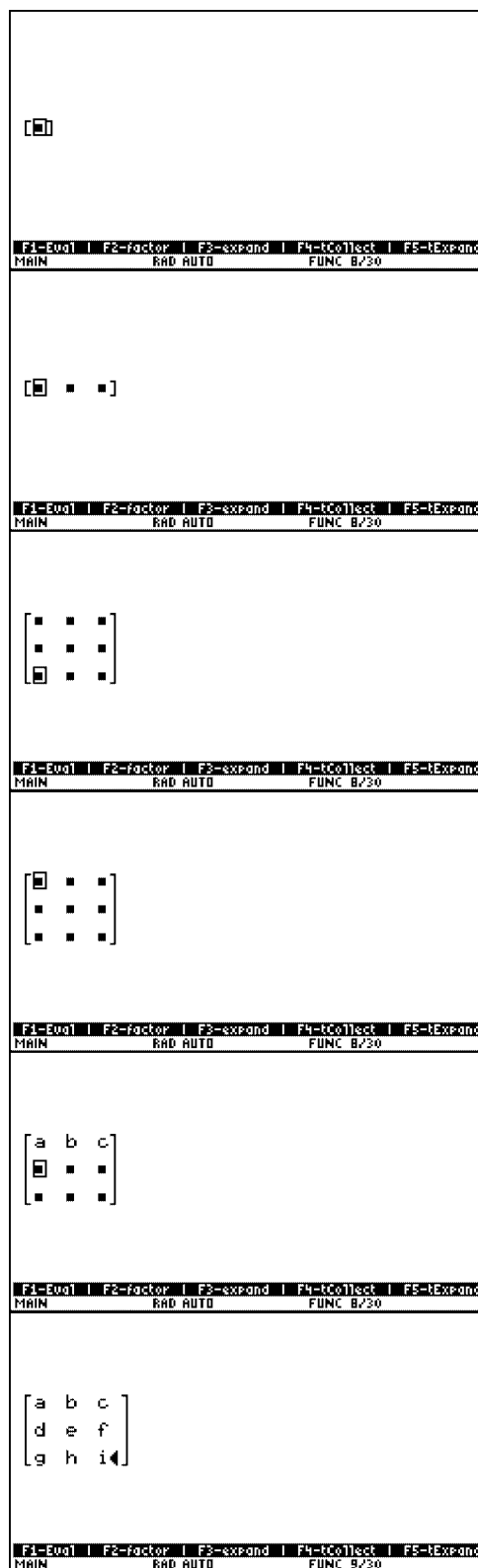
Enter the first row of matrix elements.

Press [A] [ENTER]  
[B] [ENTER]  
[C] [ENTER]

Enter the remaining row elements by pressing

[D] [ENTER] [E] [ENTER] [F] [ENTER]  
[G] [ENTER] [H] [ENTER] [I]

The matrix is complete.



The next example shows how to select and edit matrix elements.

### Example: Editing a matrix element

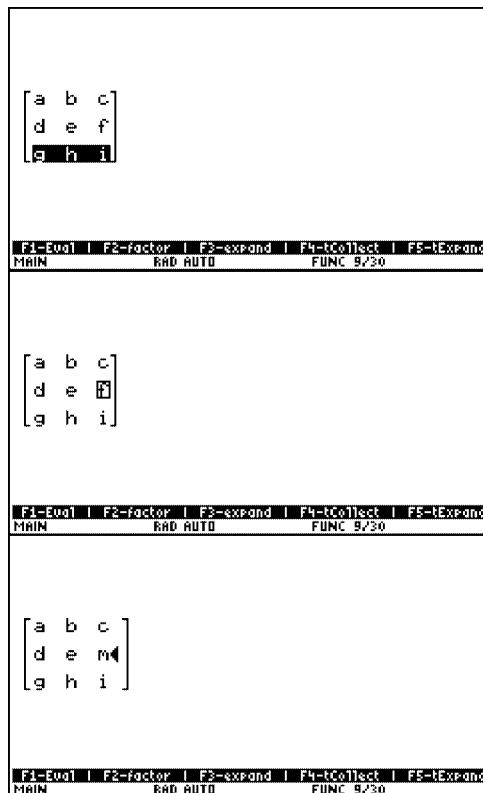
We want to change the element 'f' to 'm'. Continuing from the previous example, press [UP] to mark the last row.

Press [LEFT] to mark the second row.  
Press [DOWN] to select the 'd' element.  
Press [RIGHT] [RIGHT] to select the 'f' element.

Press [BACKSPACE] [BACKSPACE] to delete the 'f' element.

Press [M] to enter the new element.

Done!



Matrix elements can be cut, copied and pasted, as this example shows.

### Example: Cut, copy and paste

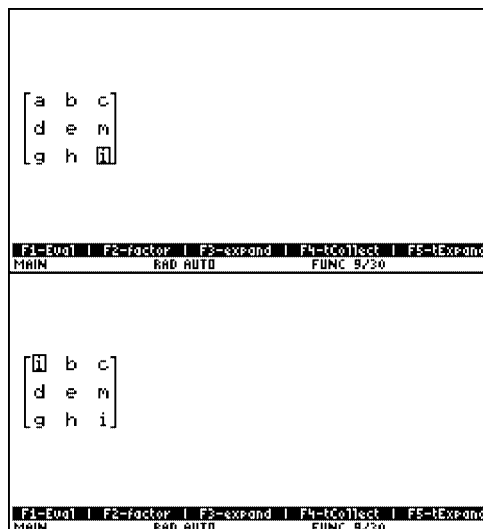
We want to replace element 'a' with element 'i'. We will do this by copying 'i', then pasting it to the 'a' element location.

Continuing from the previous example, press [UP] [RIGHT] [DOWN] [RIGHT] [RIGHT] to select 'i'.

Press [COPY] to copy 'i'.

Press [UP] [LEFT] [LEFT] [DOWN] to select 'a'.

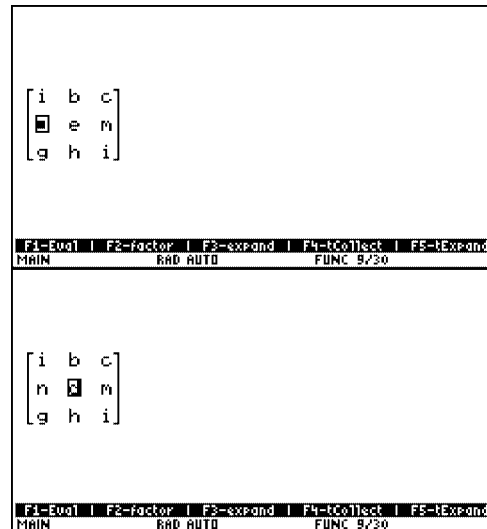
Press [PASTE], and the change is made.



Next, we want to replace element 'e' with 'd', and change 'd' to 'n'. We will do this by cutting 'd'.

Press [RIGHT] [RIGHT] [RIGHT] to mark 'd'. Press [CUT] to cut 'd'. The placeholder is shown.

Press [N] to enter the new element.  
Press [RIGHT] to select 'e'.  
Press [PASTE] to replace 'e' with 'd'.

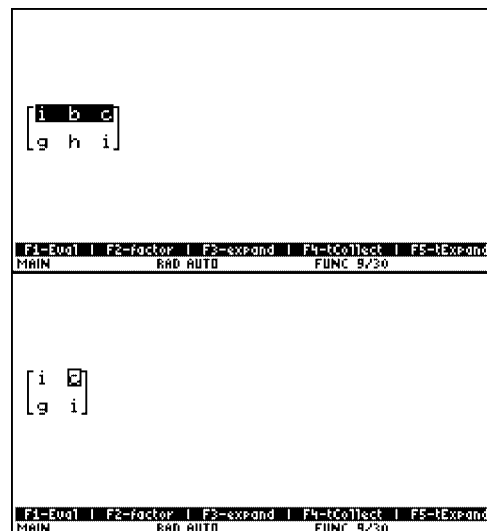


### ***Example: Deleting rows and columns***

We want to delete the second column and row of the matrix. We will delete the second row first.

Continuing from the previous example, press [UP] to mark the second row. Press [BACKSPACE] to delete the row.

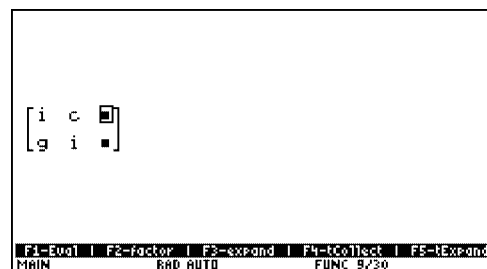
A column is deleted by deleting a placeholder in the column. Press [DOWN] [RIGHT] to mark 'b'. Press [BACKSPACE] [BACKSPACE] to delete the element, leaving a placeholder. Press [BACKSPACE] to delete the column.



### ***Example: Adding rows and columns***

We want to add a new row and column to the matrix. Rows and columns can only be added at the bottom and right of the matrix.

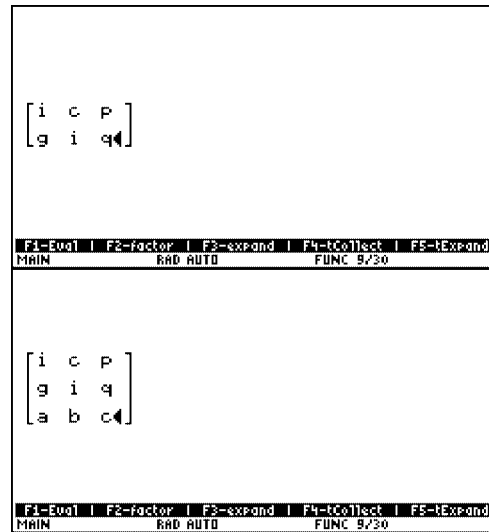
Continuing from the previous example, element 'c' is marked, so press [,] to add a new column. The placeholders are shown.



To enter the new elements 'p' and 'q', press  
[P] [ENTER] [Q]

We are ready to add the new row, but first an element or row must be marked. Press [UP] to mark the second row. Press [;] to add the row. The row element placeholders are shown.

Press [A] [ENTER] [B] [ENTER] [C] to enter the new elements.



Congratulations! You have worked through a long exercise. You should now be familiar with the techniques of creating and editing a matrix. Of course, the more you use them, the better you will get.



### Using indexed variables: lists, vectors and matrices

The elements of vectors, matrices and lists are accessed with an index, so these are called indexed variables. For example  $mylist[3]$  refers to the third element of the list  $mylist$ , so the index is 3.  $m[i,j]$  refers to the element in row  $i$  and column  $j$  of matrix  $m$ , so the indices are  $i$  and  $j$ . The key sequence [DIAMOND] [,] is used to insert the index. This example shows how to create the expression  $mylist[3]$ .

Start EQW and press [m] [y] [l] [i] [s] [t] to enter the list name. Press [DIAMOND] [,] to insert the brackets and a place-holder for the list index.

mylist[ ]		
MAIN	RAD AUTO	FUNC 6/30
mylist[3 ]		
MAIN	RAD AUTO	FUNC 6/30

Press [3] to enter the list index. The expression is complete.

This example shows how to create the matrix expression  $m[i,j]$ .

Start EQW and press [m] to enter the matrix name. Press [DIAMOND] [,] to insert the brackets and a place-holder for the matrix indices.

m[ ]		
MAIN	RAD AUTO	FUNC 6/30
m[i, j ]		
MAIN	RAD AUTO	FUNC 6/30

Press [,] to insert another place-holder, since we need two of them. Press [i] to enter the first index variable. Press [ENTER] to move to the next place-holder, and press [j]. The expression is complete.

## Working with strings

EQW is not designed to manipulate strings, but it can perform some string operations. The string concatenation operator '&' is supported by EQW.

This example shows how to enter two strings and use '&' to concatenate them.

### Example: Enter and concatenate two strings

Start EQW and press ["] to insert a pair of double quote marks and a place-holder. Press [a] [b] [c] to enter the first string.

"abc"
MAIN RAD AUTO FUNC 5/30
"abc" & "
MAIN RAD AUTO FUNC 5/30
"abc" & "def"
MAIN RAD AUTO FUNC 5/30
"abcdef"
MAIN RAD AUTO FUNC 5/30

Press [2nd] [UP] to mark the string. Press [&] to insert the '&' operator and a place-holder. [&] is [2nd] [H] on the TI-92+, and [2nd] [\*] on the TI-89.

Press ["] to insert another string.

Press [d] [e] [f] to enter the next string.

Press [2nd] [UP] to mark the complete expression, then press [F1] to evaluate the expression. The '&' operator is evaluated and the concatenated strings are shown.

Entering some characters requires special techniques. Symbols which are used for built-in functions, for example,  $\Pi$ ,  $\Sigma$ ,  $d$ , and  $\int$ , cannot be entered directly from the keyboard. There are several methods to enter these characters:

- Use the CHAR menu, if the symbol is on that built-in menu.
- Use the CATALOG menu to insert the symbol.
- Multiply the string by the function, then cut and copy the character into the string.

- Use an EQW extension to create a menu which inserts the character.

This example shows that string characters can be entered with the CHAR and CATALOG menus.

**Example: Enter special characters with the CHAR and CATALOG menus**

Start EQW and pres ["] to create a string.

" "

Press [CHAR] to display the character menu.  
Press [2] to choose the Math menu.  
Press [UP] repeatedly to select the square root symbol.  
Press [ENTER] to enter the symbol.

"√"

Press [CATALOG] .  
Press [A] to display entries beginning with 'A'.  
Press [UP] to display the first page of symbols.  
Press [UP] repeatedly to select the integral symbol.  
Press [ENTER], then press [BACKSPACE] to delete the parenthesis.

"√∫"

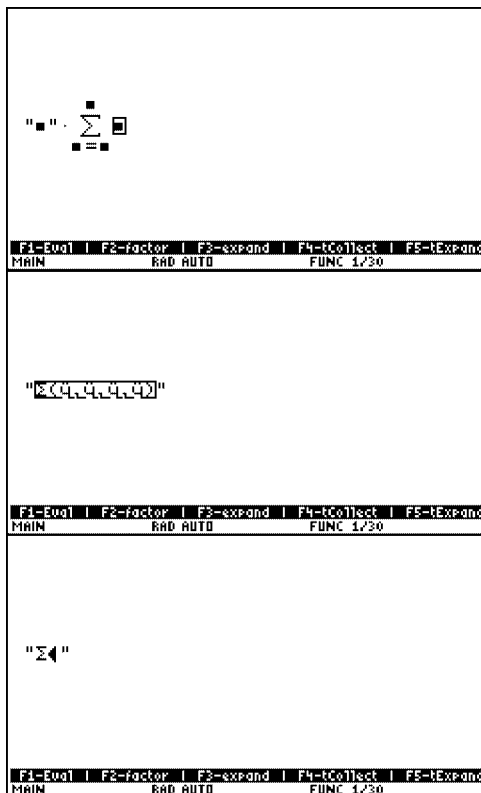
The next example shows that strings of single special characters can be created by multiplying an empty string by the function.

**Example: Enter special characters with string 'multiplication'**

Start EQW and press ["] [UP] to create and mark an empty string.

" "

Press [\*] [Σ] to create a summation form.



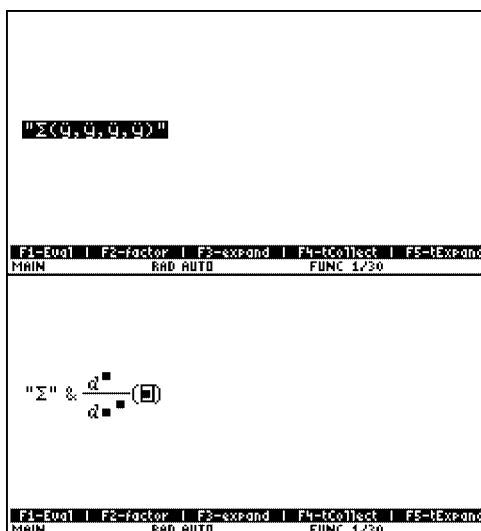
Press [UP] to mark the summation form.  
 Press [CUT] to cut the integral form.  
 Press [BACKSPACE] to delete the placeholder.  
 Press [DOWN] to select the string placeholder.  
 Press [PASTE] to paste the summation form.

Press [BACKSPACE] ten times to delete the placeholder characters and parentheses. A single sigma character remains. Remaining characters can be added if needed.

A method very similar to the last example is to use the concatenation operator "&" and the ["] key to build strings, with forms, character by character. This example shows how to create a two-character string.

### Example: Enter special characters with forms and "&"

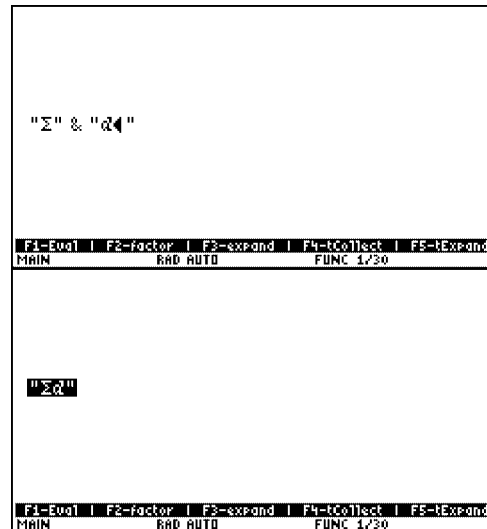
Start EQW and press [Σ] to create a summation form.  
 Press [2nd] [UP] to mark the entire form.  
 Press ["] to convert the form to a string.



Press [DOWN] to select the string contents.  
 Press [BACKSPACE] ten times to delete the placeholders and parentheses.  
 Press [2nd] [UP] to mark the string.  
 Press [&] to apply the concatenation operator.  
 Press [d] to insert a derivative form at the placeholder.

Press [UP] to mark the derivative form.  
 Press [=] to convert the form to a string.  
 Press [DOWN] to select the string contents.  
 Press [BACKSPACE] eight times to delete the placeholders and parentheses.

Press [2nd] [UP] to mark the entire expression.  
 Press [F1] to evaluate the expression.  
 The string is complete.



If you need to enter many special characters, the most efficient method is probably to create an EQW extension with the characters you use. The following program shows an example in which the characters are selected from a pop-up menu.

```
eqwprgm4()
Prgm
@EQW extension; insert special characters
@29oct01/dburkett@infinet.com

Local ins,k,m

Define ins(txt)=Prgm
  If when(main\eqwrun,true,false,false) Then
    txt→main\eqwcom
  Else
    main\sendtext(txt)
  EndIf
EndPrgm

{"√","d","f","Σ","Π","±","!"}→m

popup m,k

if k≠0:ins(m[k])

EndPrgm
```

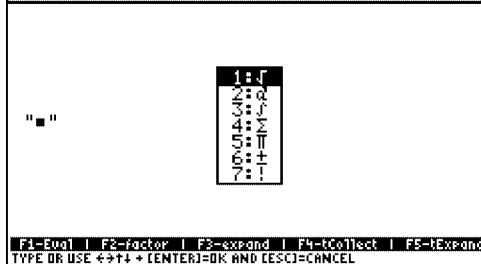
Execute the extension by pressing [DIAMOND] [4] while EQW is running. The list of special characters is stored in the *m* variable. This example shows how to use the extension to create a string of special characters.

**Example: Enter special characters with an EQW extension**

Start EQW and press ["] to create a string.



Press [DIAMOND] [4] to display the special character menu.



Press [2] to insert the derivative symbol.



To add more characters to the string, press [BACKSPACE] so the cursor is shown.



Press [DIAMOND] [4] to display the character menu.  
Press [3] to insert an integral symbol.  
Press [DIAMOND] [4] to display the character menu.  
Press [5] to insert a product symbol.

The string is complete.



### Define a function in EQW

You can use the *Define* operator in EQW to define functions. The [F8] key inserts a *Define* form, and you fill in the place-holders as needed. Suppose we want to define a simple function which is the sum of two input variables:

Start EQW and press [F8] to insert the *Define* form.

The first place-holder contains the function name and arguments.

Press [s] [u] [m] [x] [y] [( )] [,] [x] [,] [y] to enter the function name and input arguments.

Press [RIGHT] or [ENTER] to move the cursor to the place-holder at the right of the '=' sign. Press [x] [+] [y] to enter the function definition. The function definition is complete.

You can press [ENTER] to return the expression to the home screen command line. You can also evaluate the definition in EQW; to do so, press [2nd] [UP] to mark the expression, then press [F1]. EQW shows *Done* to indicate that the definition was successful.

Define $\square = \square$		
EQW	RAD AUTO	FUNC 5/30
Define sumxy(x, y) = $\square$		
EQW	RAD AUTO	FUNC 5/30
Define sumxy(x, y) = x + y		
EQW	RAD AUTO	FUNC 5/30
Done		
EQW	RAD AUTO	FUNC 5/30

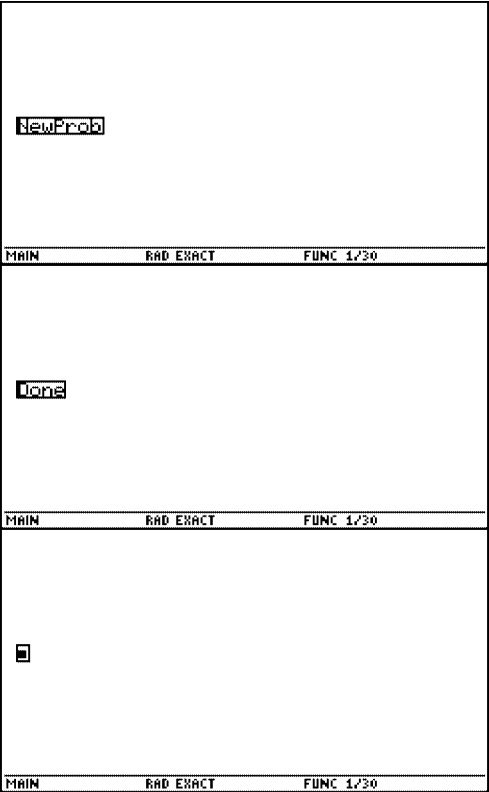
**Using NewProb in EQW**

This example shows that the built-in *NewProb* command can be executed in EQW.

Start EQW and press [CAT] [n], then scroll the cursor down to *NewProb*. Press [ENTER]. The *NewProb* command is inserted.

Push [F1] to evaluate the *NewProb* command. The display shows *Done*, indicating that *NewProb* is finished. All variables named with single letters have been deleted, so those variables can be used in symbolic calculations. Note that all functions and statistics plots are turned off, too.

Push [CLEAR] to delete the *Done* expression. You can now enter a new expression.





## Symbolic manipulation and solving

Suppose we want to manually solve

$$(x+3)(x-4) = x+1$$

for  $x$ . Start EQW, and press

[x] [+] [3] [UP] [\*] [x] [-] [4] [UP] [UP] [=] [x] [+] [1]  
to enter the equation.

Press [UP] [LEFT] to mark the left expression, then  
press [F3] to expand it.

Subtract  $(x+1)$  from both sides of the equation by  
pressing [UP] to mark the entire equation. Press [-] [x]  
[+] [1].

Press [2nd] [UP] to mark the entire equation, then  
press [F1] to evaluate it.

Since the equation is a quadratic, try factoring the  
expression on the left by pressing [DOWN] to mark the  
left side, then press [F2]. Since the expression did not  
change, there are no simple factors.

Use  $cSolve()$  to find the solution.  $cSolve()$  is used in  
case the equation has complex solutions. Press [UP]  
[MATH] [9] [a] [1] [ENTER], to insert the  $cSolve()$   
function and a place-holder. Press [x], then [2nd] [UP].  
Now, press [F1] to evaluate the expression. The  
solutions are shown.

$$(x+3)(x-4) = x+1$$

MAIN RAD EXACT FUNC 1/30

$$x^2 - x - 12 = x + 1$$

MAIN RAD EXACT FUNC 1/30

$$(x^2 - x - 12 = x + 1) - (x + 1)$$

MAIN RAD EXACT FUNC 1/30

$$x^2 - 2x - 13 = 0$$

MAIN RAD EXACT FUNC 1/30

$$x^2 - 2x - 13 = 0$$

MAIN RAD EXACT FUNC 1/30

$$x = -(\sqrt{14} - 1) \text{ or } x = \sqrt{14} + 1$$

MAIN RAD EXACT FUNC 1/30

## Solving simultaneous equations with matrices

We want to solve the following set of equations for  $x$ ,  $y$ ,  $z$  and  $t$ :

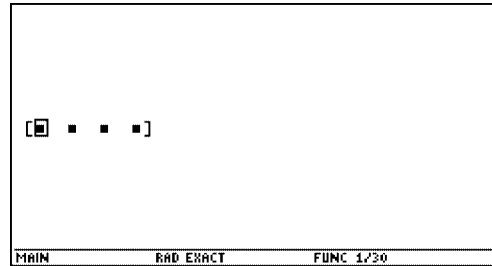
$$6x + 3\frac{3}{4}y - 3z + 4t = 0$$

$$\frac{28}{52}x - z + 6t = 1$$

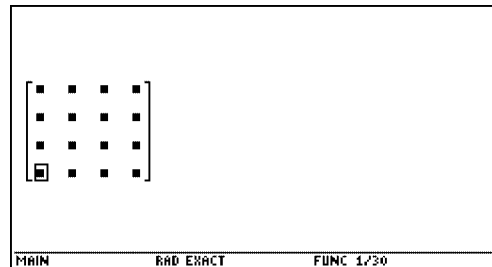
$$-2\frac{2}{7}x - 3y + \sqrt{5}z - 2t = -\frac{1}{3}$$

$$x + 9y + 4\sqrt{2}z + \frac{77}{16}t = 6\frac{1}{2}$$

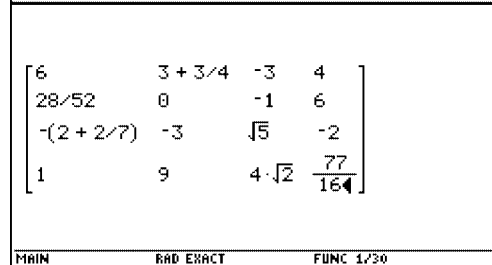
Start EQW and press [ ] to create the coefficients matrix. Press [,] [,] [,] to insert place-holders for 3 more columns



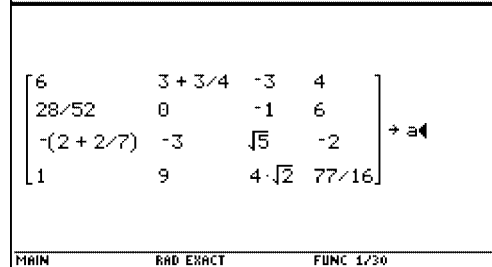
Press [,], [,], [,] to create three new rows.



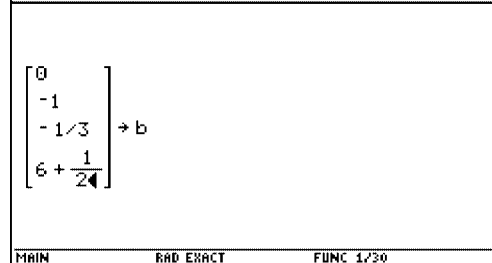
Push [2nd] [UP] [DOWN] [DOWN] to move the cursor to the first element. To enter the first row, press [6] [ENTER] [3] + [3] [/] [4] [ENTER] [3] [-] [ENTER] [4] [ENTER]. Note that [ENTER] is used to move to the next place-holder after each element is entered. When [ENTER] is pressed after the last element in a row is entered, the cursor moves to the next row. Enter the coefficients in the remaining rows.



Store the matrix to a variable called  $a$ . Press [2nd] [UP] to mark the matrix, then press [STO] [a]. Press [2nd] [UP] [F1] to mark and evaluate the expression, which stores the matrix to  $a$ . Press [CLEAR] to clear the marked matrix. The screen shot at right shows the screen before the expression is marked and evaluated.

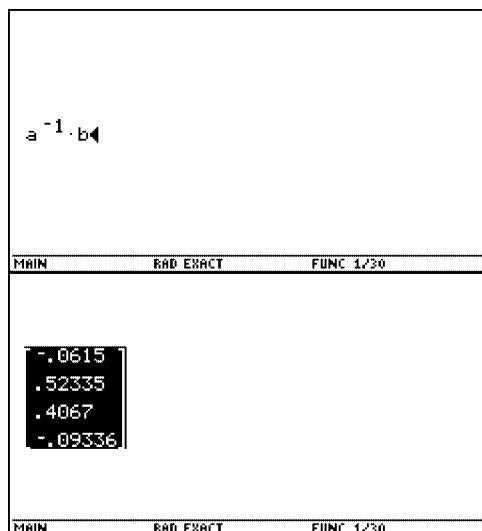


Press [ ] to create a new matrix. Press [,], [,], [,] to add three rows. Enter the constant coefficients. Press [2nd] [UP] [STO] [b] [2nd] [UP] [F1] to store the coefficient matrix to a variable  $b$ . Press [CLEAR] to delete the coefficient matrix. The screen shot shows the expression before [CLEAR] is pressed.



We will solve the system of equations using the matrix inverse. Press [a] [^] [(-)] [1] [UP] [\*] [b]

Press [2nd] [UP] to mark the expression. Press [DIAMOND] [ENTER] to find the approximate solution. Alternatively, you could have pressed [F1] to find the exact solution, which takes longer.



### Use a form to evaluate an expression

This example shows how to enter an expression, save it as a form, then later evaluate the expression in EQW with different input arguments. As an example, we'll use the expression to calculate the payment in a compound interest loan, which is

$$PMT = \frac{PV \cdot i}{100c \left( 1 - \left( \frac{c}{c + .01i} \right)^{c \cdot n} \right)}$$

where PMT is the payment, PV is the present value (the amount of the loan),  $c$  is the number of payments per year,  $i$  is the annual interest rate, and  $n$  is the number of years to pay back the loan.

Start EQW and enter the expression:

[p] [v] [\*] [i] [UP] [/]  
[1] [0] [0] [\*] [c] [\*] [1] [-] [c] [/] [c] [+] [-] [0] [1] [\*] [i]  
[UP] [UP] [UP] [^] [c] [\*] [n]

Mark the expression with [2nd] [UP].

Press [I] to add the 'with' operator.

To add three *and* operators and four place-holders,

press [a] [n] [d] [ENTER] [a] [n] [d] [ENTER]

[a] [n] [d] [ENTER]

Start filling in the place-holders:

[p] [v] [=] [RIGHT]

[n] [=] [RIGHT]

[c] [=] [RIGHT]

[i] [=]

Now the expression is complete, with place-holders to fill in values later. Press [2nd] [UP] to mark the expression, then press ["] to convert it to a string. Press [STO] [p] to save the string to a variable called  $p$ . Press [ENTER] to return the expression to the home screen, and press [ENTER] again to evaluate it.

The screenshots show the following steps:

- Entering the formula: 
$$\frac{PV \cdot i}{100 \cdot c \cdot \left( 1 - \left( \frac{c}{c + .01000 \cdot i} \right)^{c \cdot n} \right)}$$
- Marking the expression with [2nd] [UP] and adding the 'with' operator [I].
- Adding three *and* operators and four placeholders: 
$$\frac{PV \cdot i}{100 \cdot c \cdot \left( 1 - \left( \frac{c}{c + .01000 \cdot i} \right)^{c \cdot n} \right)} \text{ and } \blacksquare \text{ and } \blacksquare \text{ and } \blacksquare \text{ and } \blacksquare$$
- Filling in the placeholders: 
$$\frac{PV \cdot i}{100 \cdot c \cdot \left( 1 - \left( \frac{c}{c + .01000 \cdot i} \right)^{c \cdot n} \right)} \text{ and } PV = \blacksquare \text{ and } n = \blacksquare \text{ and } c = \blacksquare \text{ and } i = \blacksquare$$
- Converting the expression to a string and saving it to variable  $p$ : 
$$\blacksquare \text{ "PV*i/(100*c*(1-(c/(c+.01*i))^(c*n)))" } \blacksquare$$
  
$$\blacksquare \text{ "PV*i/(100*c*(1-(c/(c+.01*i))^(c*n)))" } \blacksquare$$
  
$$\blacksquare \text{ and n=y and c=y and i=y" } \rightarrow p$$

To evaluate the expression, start EQW with the  $p$  variable, by entering  $eqw(p)$  at the command line. EQW will start and display the expression. Press [ENTER] to move to the first placeholder.

Enter a present value of \$25000 with [2][5][0][0][0] [ENTER].  
Enter the five year period with [5] [ENTER].  
Enter 12 payments/year with [1][2] [ENTER].  
Enter the percentage rate of 12 with [1][2].

Press [2nd] [UP] to mark the expression.  
Press [F1] to evaluate it. The payment of \$556.11 is shown.

$\frac{PV \cdot i}{100 \cdot c \cdot \left( 1 - \left( \frac{c}{c + .01000 \cdot i} \right)^{C \cdot n} \right)} \quad   \quad PV = \blacksquare \text{ and } r$		
EQW	RAD AUTO	FUNC 2/30
$  \quad PV = 25000 \text{ and } n = 5 \text{ and } c = 12 \text{ and } i = 12$		
EQW	RAD AUTO	FUNC 2/30
$556.1119$		
EQW	RAD AUTO	FUNC 2/30

### Creating a form with conditional (test) operators

EQW automatically creates an input form if you press the key for a conditional operators such as [=], [<], [>], etc. Suppose we want to enter the expression  $2x+y < 4x+t$ :

Start EQW and press [<] to create the form. EQW inserts the '<' operator, and creates two place-holders for the expressions.

Press [2] [\*] [x] [UP] [+] [y] to enter the left expression.

Push [RIGHT] to move the cursor to the right place-holder. Press [4] [\*] [x] [UP] [+] [t]. The expression is complete.

<div>□ &lt; □</div>		
EQW	RAD EXACT	FUNC 2/30
<div><math>2 \cdot x + y &lt; \square</math></div>		
EQW	RAD EXACT	FUNC 2/30
<div><math>2 \cdot x + y &lt; 4 \cdot x + t</math></div>		
EQW	RAD EXACT	FUNC 2/30

### ***Nested multiplication***

Suppose we want to enter the following equation:

$$y = a \cdot (x + b \cdot (x + c \cdot (x + d)))$$

When you enter this equation in the normal calculator entry line, you simply type it as shown, with the parentheses. However, EQW supplies its own matching parentheses. The example shows that EQW automatically applies the parentheses correctly to enter the equation.

Start EQW

Press [y] [=] [a] [\*] [x]

Press [+]. Note that EQW automatically applies the parentheses and a place-holder.

Press [b] [\*] [x] [+]. Again, EQW properly applies the parentheses.

Press [c] [\*] [x] [+ [d]. The expression is complete.



y = a · x

y = a · (x + )

y = a · (x + b · (x + ) )

y = a · (x + b · (x + c · (x + d)))

Now suppose that we really wanted

$$y = ax + bx + cx + d$$

We cannot just type the same keystrokes as in the previous example, because that would apply parentheses where we do not want them. Instead, we use [UP] to mark each product, before adding the next, as shown:

Start EQW

Press [y] [=] [a] [\*] [x] [UP]. The product is marked.

Press [+] [b] [\*] [x] [UP]. The bx product is marked.

Press [+] [c] [\*] [x] [UP]. The cx product is marked.

Press [+] [d]. The expression is complete.



y = a · x




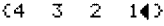
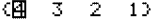

y = a · x + b · x

y = a · x + b · x + c · x

y = a · x + b · x + c · x + d

## Creating and editing a list

Start EQW and press [{} to create a list. We want to enter the list {4,3,2,1}

		
MAIN	RAD EXACT	FUNC 4/30
		
MAIN	RAD EXACT	FUNC 4/30
		
MAIN	RAD EXACT	FUNC 4/30
		
MAIN	RAD EXACT	FUNC 4/30
		
MAIN	RAD EXACT	FUNC 4/30
		
MAIN	RAD EXACT	FUNC 4/30

Push [,] [,] [,] to add three place-holders

Push [LEFT] [LEFT] [LEFT] to move the cursor to the first element.

Push [4] [RIGHT]  
[3] [RIGHT]  
[2] [RIGHT]  
[1]

The list we wanted is entered.

Suppose we really wanted the list {5, 3, 2, 1}, so we need to change the first element from 4 to 5. Push [LEFT] [LEFT] [LEFT] to move the cursor to the first element.

Push [BACKSPACE] two times, then press [5]. The correction is made.



### Entering angles in degrees, minutes and seconds

This example shows how to enter 22 degrees, 33 minutes and 44.4 seconds.

Start EQW and press [2] [2] [°]  
Note that 22° is marked.

22°		
EQW	RAD EXACT	FUNC 5/30
22° 33'		
EQW	RAD EXACT	FUNC 5/30
22° 33' 44.4"		
EQW	RAD EXACT	FUNC 5/30
22° 55' 44.4"		
EQW	RAD EXACT	FUNC 5/30

Press [,] to make a placeholder for the minutes,  
then press [3] [3] to enter the minutes

Press [,] to make a place-holder for the seconds,  
then press [4] [4] [,] [4] to enter the minutes. The  
expression is complete.

To change the angle to 22°55'44.4", press [LEFT]  
so that an outline box surrounds 33, then press  
[BACKSPACE] to edit the element. Push  
[BACKSPACE] two times to delete 33, then enter  
[5] [5]. The expression is complete.

### **Entering expressions with logical operators (*and*, *or*, *not*, *xor*)**

Logical expressions contain the logical operators *and*, *or*, *xor* and *not*. These are usually called Boolean expressions, in honor of the 19th-century British mathematician George Boole. For a short biography of Boole, see

<http://www-groups.dcs.st-and.ac.uk/~history/Mathematicians/Boole.html>

Boolean expressions can be typed in directly, or entered from the MATH Test menu. The logical operators are needed to use *Solve()* with multiple equations, to apply constraints to *Solve()*, and to use *deSolve()* with boundary conditions.

Just as there is a hierarchy for arithmetic operators and functions, there is also a hierarchy for logical operators. The hierarchy specifies the order in which the operators are applied in an expression. The hierarchy is

1. *not()*
2. *and*
3. *or*, *xor* (exclusive or)

As with the arithmetic operators, parentheses can be used to control the order in which the logical operators are applied. Since *or* and *xor* have the same priority, they are evaluated from left to right in the order in which they occur in the expression.

You need to understand the logical operator hierarchy to understand the results when you enter logical expressions. EQW will not place parentheses where they are not needed.

You can either type the complete expression in directly, or let EQW recognize the logical operators and insert place-holders. As an example, suppose we want to enter

$x \text{ and } y \text{ or } z$

To enter this directly, just type it in as shown. Note that this expression is really the same as

$(x \text{ and } y) \text{ or } z$

because *and* has a higher priority than *or*. You can use EQW's ability to recognize the built-in logic operators and automatically insert place-holders. To enter the expression this way, press

[a] [n] [d] [RIGHT] [UP] [o] [r] [RIGHT]

Notice that the *and* expression temporarily disappears when you type [o], but the complete expression reappears when you press the final [RIGHT]. To enter the variable names in the place-holders, press

[LEFT] [LEFT] [x] [RIGHT] [y] [RIGHT] [z]

Now suppose that we want to enter

$x \text{ and } (y \text{ or } z)$

We want to use parentheses to evaluate *y or z* before the *and*. We could enter this directly by typing each character, or we can use

[a] [n] [d] [RIGHT] [RIGHT] [o] [r] [RIGHT]

to create the expression with place-holders, then use [LEFT] and [RIGHT] to move the cursor and fill in the place-holders with x, y and z. To enter this expression by using the Math menu, instead of typing the operators, use these keystrokes:

[MATH] [8] [8] [RIGHT] [MATH] [8] [9]

which results in the expression, with place-holders.

Some more examples:

(x or t) and (y or z)

With the Math menu: [MATH] [8] [9] [UP] [MATH] [8] [8] [MATH] [8] [9]  
[LEFT] [LEFT] [DOWN] [x] [ENTER]  
[t] [ENTER] [y] [ENTER] [z]

By typing the operators: [o] [r] [RIGHT] [UP] [a] [n] [d] [RIGHT] [o] [r] [RIGHT]  
[LEFT] [LEFT] [x] [ENTER]  
[t] [ENTER] [y] [ENTER] [z]

((x or t) and y) and z

With the Math menu: [MATH] [8] [9] [UP] [MATH] [8] [8] [UP] [MATH] [8] [8]  
[2nd] [LEFT] [x] [ENTER] [t] [ENTER]  
[y] [ENTER] [z] [ENTER]

By typing the operators: [o] [r] [RIGHT] [UP] [a] [n] [d] [RIGHT] [UP] [a] [n] [d] [RIGHT]  
[2nd] [LEFT] [x] [ENTER] [t] [ENTER]  
[y] [ENTER] [z] [ENTER]

Note that this expression is displayed as (x or t) and y and z. Only one set of parentheses is needed, because of the left-to-right evaluation of the two *and* operators.

Typically we often need to combine logical expressions with test expressions. These examples show three ways to enter the expression

$x < 1$  and  $y > 2$

By typing in the expression: [x] [<] [1] [UP] [a] [n] [d] [RIGHT] [a] [n] [d] [y] [>] [2]

With place-holders: [a] [n] [d] [RIGHT] [x] [<] [1] [ENTER] [y] [>] [2]

With the Math menu: [MATH] [8] [8] [x] [<] [1] [ENTER] [y] [>] [2]

This example shows a more complicated expression, with parentheses needed:

$(x < 1 \text{ or } y > 2) \text{ and } z = 3$

By typing in the expression: [x] [<] [1] [UP] [o] [r] [RIGHT] [y] [>] [2] [UP] [UP]  
[a] [n] [d] [RIGHT] [z] [=] [3]

With place-holders:           [o] [r] [RIGHT] [UP] [a] [n] [d] [RIGHT] [ENTER]  
                                      [x] [<] [1] [ENTER] [y] [>] [2] [ENTER] [z] [=] [3]

With the Math menu:           [MATH] [8] [9] [UP] [MATH] [8] [8] [ENTER]  
                                      [x] [<] [1] [ENTER] [y] [>] [2] [ENTER] [z] [=] [3]

In summary:

1. There are three ways to enter logical expressions: by simply typing them in, by typing them in and letting EQW supply place-holders, and by using the Math menu.
2. Use [UP] to mark a subexpression, to apply additional operators.
3. Use [RIGHT] after typing in an operator, to apply the place-holders.
4. You may use either [RIGHT] or [ENTER] to move the cursor to the next place-holder.
5. EQW will automatically remove parentheses from your expression if they are not necessary.

## Entering and evaluating a definite integral

This example shows how to evaluate this definite integral:

$$\int_t^{2t} (x^2 + 3x) dx$$

Start EQW and press [f] to insert the integral form.

$$\int \square d\square$$

Enter the integrand:

[x] [^] [2] [UP] [+] [3] [\*] [x]

$$\int \square (x^2 + 3 \cdot x) d\square$$

Press [RIGHT] to move the cursor to the place-holder for the variable of integration. Press [x] to enter the variable. Press [RIGHT] to move the cursor to the place-holder for the lower integration limit.

$$\int \square (x^2 + 3 \cdot x) dx$$

Press [t] to enter the lower integration limit. Press [RIGHT], to move the cursor to the place-holder for the upper integration limit. Press [2] [\*] [t] to enter the limit.

$$\int_t^{2 \cdot t} (x^2 + 3 \cdot x) dx$$

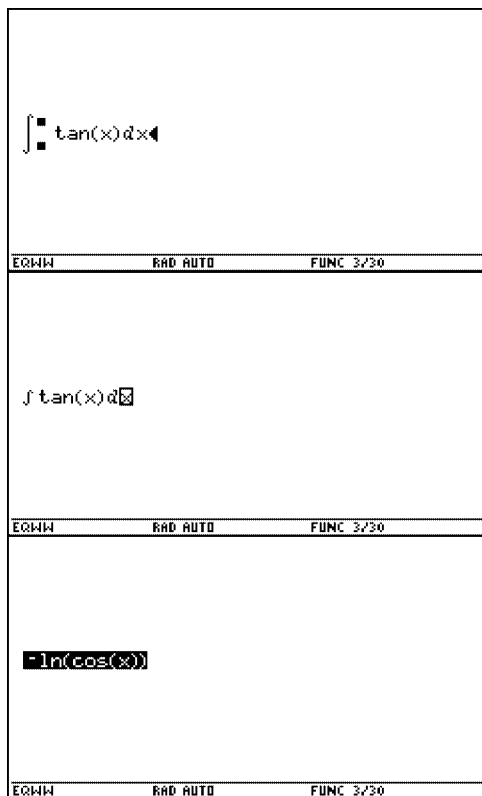
Press [2nd] [UP] to mark the complete expression. Press [F1] to evaluate the integral.

$$\frac{t^2 \cdot (14 \cdot t + 27)}{6}$$

### Entering an indefinite integral (anti-derivative)

The procedure to enter an indefinite integral is similar to entering a definite integral, except that the integration limits must be removed because EQW automatically inserts the limits. To remove the limits, move the cursor to the place-holder for the lower limit and press [BACKSPACE].

Start EQW and press [f] to insert the integral form. Press [tan] [x] to enter the integrand, then press [RIGHT] to move the cursor to the place-holder for the variable of integration. Press [x] to enter the variable.

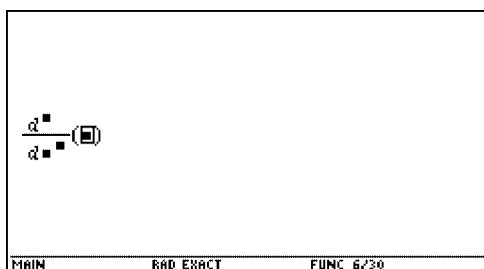


Press [ENTER] to move the cursor to the place-holder for the lower integration limit. Press [BACKSPACE] to delete the limits.

Press [2nd] [UP] to mark the complete expression. Press [F1] to evaluate the integral. The integral is shown.

## Entering derivatives

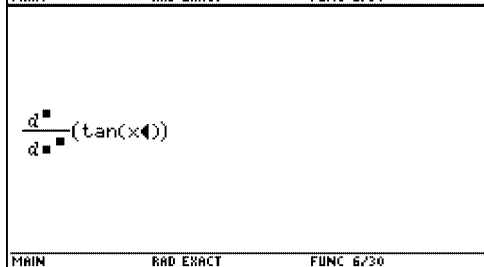
Start EQW and press [d], the derivative operator. EQW inserts a derivative template. The cursor is at the derivative function argument.



$$\frac{d}{dx} ( )$$

MAIN RAD EXACT FUNC 6/30

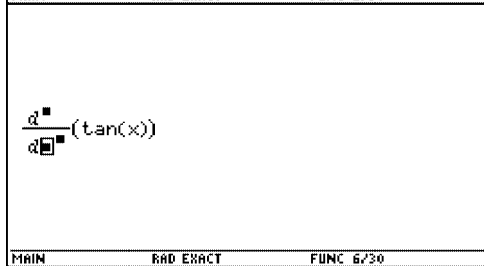
Push [TAN] [x] to enter the argument function.



$$\frac{d}{dx} (\tan(x))$$

MAIN RAD EXACT FUNC 6/30

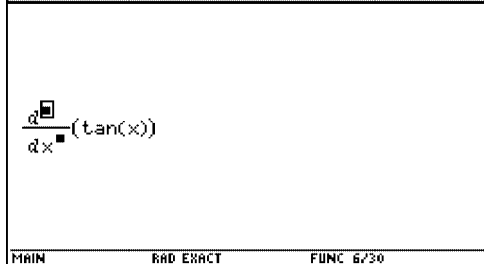
Push [RIGHT] to move the cursor to the first placeholder, which is the dependent variable.



$$\frac{d}{dx} (\tan(x))$$

MAIN RAD EXACT FUNC 6/30

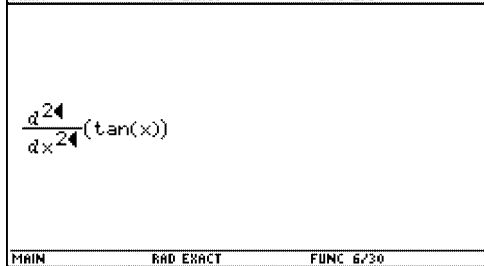
Push [x] to enter the independent variable, and press [RIGHT] to move to the next place-holder. The next place-holder is the order of the derivative.



$$\frac{d}{dx} (\tan(x))$$

MAIN RAD EXACT FUNC 6/30

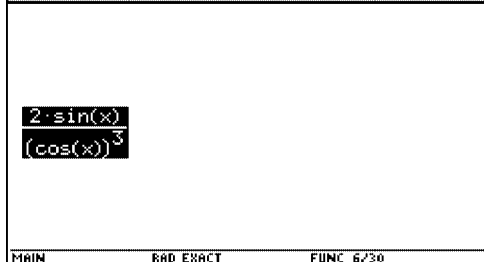
Push [2] to enter the order of the derivative. Note that the derivative order is automatically entered in both the numerator and denominator. The expression is complete. If you had wanted the first derivative instead of the second derivative, enter '1' instead of '2'.



$$\frac{d^2}{dx^2} (\tan(x))$$

MAIN RAD EXACT FUNC 6/30

If you want to evaluate the derivative, press [2nd] [UP] to select the entire expression, then press [F1] to evaluate the derivative.



$$2 \cdot \sin(x) / (\cos(x))^3$$

MAIN RAD EXACT FUNC 6/30

Slightly different forms of the derivative operator will be shown, depending on whether you press the  $[d]$  key, or select  $d/$  *differentiate* from the MATH menu. If you press the  $[d]$  key, a place-holder for the order of the derivative is shown. To remove the place-holder, move the cursor to the place-holder and press [BACKSPACE].

If you select  $d/$  *differentiate* from the MATH or CATALOG menu, the place-holder for the derivative order is not shown. To add it, move the cursor to the place-holder for the variable of differentiation and press [,].



## Entering a sum

This example shows how to enter and calculate this sum:

$$\sum_{i=1}^3 \left( \frac{i^2 - \sin(i)}{(2 \cdot i)! + i + 1} \right)$$

Set the mode to Auto, and start EQW. Press  $\Sigma$  to insert a sum form.

Enter the summand:

Press  $[i] [\wedge] [2] [\text{UP}] [/] [2] [*] [i] [\text{UP}]$

Press  $[2\text{nd}] [w]$  to enter the ! symbol on the 92+, or press  $[\text{DIAMOND}] [/]$  on the 89.

Press  $[\text{UP}] [-] [\text{SIN}] [i] [\text{UP}] [\text{UP}] [/] [i] [+] [1]$

Press  $[\text{ENTER}]$  to move the cursor to the place-holder for the sum variable. Press  $[i]$  to enter the variable. Press  $[\text{ENTER}]$  to move the cursor to the next place-holder, and press  $[1]$ . Press  $[\text{ENTER}]$  to move the cursor to the last place-holder, and press  $[3]$  to enter the upper sum limit.

Press  $[2\text{nd}] [\text{UP}]$  to mark the entire expression.

Press  $[F1]$  to evaluate the sum.

Press  $[\text{DIAMOND}] [\text{ENTER}]$  to find the approximate value for the sum.

The image displays a series of five calculator screen captures, each showing a different stage of entering and evaluating the sum  $\sum_{i=1}^3 \left( \frac{i^2 - \sin(i)}{(2 \cdot i)! + i + 1} \right)$ .

- Screen 1:** Shows the sum template  $\sum_{i=1}^{\square} \square$  with the EQW mode indicator.
- Screen 2:** Shows the summand  $\left( \frac{i^2 - \sin(i)}{(2 \cdot i)! + i + 1} \right)$  being entered into the template.
- Screen 3:** Shows the sum limits  $i=1$  and upper limit  $3$  being entered.
- Screen 4:** Shows the entire expression marked for evaluation with a black background.
- Screen 5:** Shows the final approximate result  $\approx 4.5043$ .

## Entering and evaluating a product

This example shows how to use EQW to enter and evaluate this product:

$$\prod_{i=1}^2 \left( \int (x^i \cdot e^{i \cdot x}) dx \right)$$

Start EQW. Insert a product form:

press [DIAMOND] [∏] [SHIFT] [p] on the 89, or

press [DIAMOND] [g] [SHIFT] [p] on the 92+

Press [ ( ], [RIGHT]. EQW recognizes the product symbol and inserts the product form.

$$\prod_{i=1}^{\square}$$

Enter the product function:

[∫] [x] [^] [i] [UP] [\*] [e^] [i] [\*] [x]

Press [RIGHT] or [ENTER] to move the cursor to the integration variable place-holder, and press [x]

$$\prod_{i=1}^{\square} \int_{\square}^{\square} (x^i \cdot e^{i \cdot x}) dx$$

The integral is indefinite, so we need to remove the integration limits. Press [ENTER] to move the cursor to the lower integration limit, then press [BACKSPACE] to remove the integration limits.

$$\prod_{i=1}^{\square} \int (x^i \cdot e^{i \cdot x}) dx$$

Press [RIGHT] or [ENTER] to move the cursor to the product index variable place-holder, then press [i]. Press [ENTER] to move the cursor to the lower limit and press [1]. Press [ENTER] to move the cursor to the upper limit place-holder and press [2]. The expression is complete.

$$\prod_{i=1}^2 \int (x^i \cdot e^{i \cdot x}) dx$$

Press [2nd] [UP] to mark the complete expression, then press [F1] to evaluate it. The computed product is shown.

$$\frac{(x-1) \cdot (2 \cdot x^2 - 2 \cdot x + 1) \cdot e^{3 \cdot x}}{4}$$

## Entering a limit

This example shows how to enter and evaluate this limit in EQW:

$$\lim_{x \rightarrow \frac{1}{r+1}} \left( \frac{x \cdot r \cdot (e^{x \cdot r + x - 1} - 1)}{x \cdot r + x - 1} \right)$$

This simple-looking limit gave quite a few computer algebra systems trouble in the 1980's.

Start EQW and type [I] [i] [m] [i] [t] [(] [.,]. EQW recognizes the built-in *limit()* function and inserts the limit template.

Enter the limit argument with these keystrokes:

[x] [\*] [r] [UP] [\*] [e^x]  
 [x] [\*] [r] [UP] [+] [x] [-] [1] [UP] [UP] [UP] [-] [1]  
 [UP] [UP] [/]  
 [x] [\*] [r] [UP] [+] [x] [-] [1]

The limit argument has been entered. Now press [RIGHT] to move the cursor to the first place-holder.

Press [x], then press [RIGHT] to move to the next placeholder. Press [1] [/] [r] [+] [1] to enter the limit expression.

To evaluate the limit, press [2nd] [UP] to mark the entire expression, then press [F1]. The result is shown.

lim

EQW44RAD EXACTFUNC 5/30

lim

x

\*

r

\*

(

e

\*

x

\*

r

+

x

-

1

-

1

)

-

1

)

x

\*

r

+

x

-

1

EQW44RAD EXACTFUNC 5/30

lim

x

\*

r

\*

(

e

\*

x

\*

r

+

x

-

1

-

1

)

-

1

)

x

\*

r

+

x

-

1

EQW44RAD EXACTFUNC 5/30

x

\*

r

\*

(

e

\*

x

\*

r

+

x

-

1

-

1

)

-

1

)

1

/

r

+

1

EQW44RAD EXACTFUNC 5/30

r

/

r

+

1

EQW44RAD EXACTFUNC 5/30

With the *limit()* function you can specify the direction from which the limit is taken: either from  $-\infty$  ('from the left') or from  $+\infty$  ('from the right'). In EQW, you change the direction limit by inserting a direction place-holder with [.,] then changing it to '+' or '-' with the [(-)] key. Suppose we want to enter this limit expression:

$$\lim_{x \rightarrow 0^-} \frac{\sin(x)}{x}$$

Start EQW and type 'limit', then press [(] [.] to create a limit() form.

The EQW screen displays the initial limit function template. The top line shows 'lim' followed by a square box containing a right-pointing arrow. Below this, the status bar shows 'EQW', 'RAD EXACT', and 'FUNC 2/30'.

Press [x] [sin] [/] [x] to enter the limit argument. Press [RIGHT] to move the cursor to the limit variable place-holder, and press [x]. Press [RIGHT] to move the cursor to the limit value place-holder.

The EQW screen shows the limit argument  $\frac{\sin(x)}{x}$  entered inside the square box. The cursor is now positioned at the limit variable place-holder (x →). The status bar remains 'EQW', 'RAD EXACT', and 'FUNC 2/30'.

Press [0] to enter the limit value, then press [.,] to enter a place-holder for the limit direction. The '+' sign indicates that the limit will be taken from the right.

The EQW screen shows the limit value '0' entered at the limit variable place-holder. The cursor is now at the limit direction place-holder (.,). The status bar remains 'EQW', 'RAD EXACT', and 'FUNC 2/30'.

Press [(-)] to change the limit sign from '+' to '-'. The expression is complete.

The EQW screen shows the final limit expression  $\lim_{x \rightarrow 0^-} \frac{\sin(x)}{x}$  complete. The status bar remains 'EQW', 'RAD EXACT', and 'FUNC 2/30'.

## Entering a differential equation

This example shows how to enter and solve this differential equation:

$$\frac{d^2y}{dx^2} + 2\frac{dy}{dx} + y = x^2$$

Start EQW and type

[d] [e] [S] [o] [I] [v] [e] [I] [ENTER]

to insert a *deSolve()* form. Note that the 'S' must be capitalized so EQW can recognize the function. Alternatively, you can enter *deSolve()* from the MATH menu.

Enter the differential equation at the first place-holder:

[y] ['] ['] [+] [2] [\*] [y] ['] [UP] [UP] [+] [y] [UP] [=]  
[x] [^] [2]

Press [ENTER] to move the cursor to the next place-holder. Press [x] to enter the independent variable. Press [ENTER] to move the cursor to the next place-holder. Press [y] to enter the dependent variable. The expression is complete.

Press [2nd] [UP] to mark the complete expression. Press [F1] to evaluate the expression. The solution is shown.

deSolve(□, □, □)		
EQW	RAD AUTO	FUNC 3/30
deSolve(y'' + 2·y' + y = x <sup>2</sup> , □, □)		
EQW	RAD AUTO	FUNC 3/30
deSolve(y'' + 2·y' + y = x <sup>2</sup> , x, y)		
EQW	RAD AUTO	FUNC 3/30
y = (01·x + 02)·e <sup>-x</sup> + x <sup>2</sup> - 4·x + 6		
EQW	RAD AUTO	FUNC 3/30

To use deSolve() to find a *particular* solution of a second-order differential equation, you need to enter two initial conditions. This example shows that process to solve this differential equation

$$\frac{d^2y}{dx^2} + \frac{dy}{dx} = x \quad \text{where} \quad \frac{dy}{dx}(0) = 0 \quad \text{and} \quad y(0) = 0$$

Start EQW and type [d] [e] [S] [o] [I] [v] [e] [I] [ENTER] to create the deSolve() form. With the cursor at the first place-holder, type [a] [n] [d] [ENTER] [a] [n] [d] [ENTER] to enter two *and* operators, and three place-holders

Enter the differential equation at the first place-holder: [y] ['] ['] [+] [y] ['] [UP] [=] [x]  
Press [ENTER] to move to the next place-holder, and type in the first initial condition:  
[y] ['] [I] [0] [I] [=] [0]  
Press [ENTER] to move the cursor to the next place-holder, and type in the second condition:  
[y] [I] [0] [I] [=] [0]

Push [ENTER] to move the cursor to the first place-holder, and press [x] for the independent variable. Press [ENTER] to move the cursor to the next place-holder, and press [y] for the dependent variable. Press [2nd] [UP] [F1] to mark and evaluate the expression. The particular solution is shown.

deSolve( and and , , )		
EQW	RAD AUTO	FUNC 3/30
[ve(y' ' + y' = x and y '(0) = 0 and y(0) = 0],		
EQW	RAD AUTO	FUNC 5/30
$y = -e^{-x} + \frac{x^2}{2} - x + 1$		
EQW	RAD AUTO	FUNC 5/30

### Using *augment()*

*augment()* is a built-in function that augments two lists or matrices, so *augment()* must have two input arguments. *augment()* is unique in that either a comma or a semicolon may be used to separate two matrix arguments. For example,

*augment(m1,m2)* appends matrix *m2* as new columns to matrix *m1*

*augment(m1;m2)* appends matrix *m2* as new rows to matrix *m1*

*augment(;)* is the only function that cannot be entered in EQW. This restriction exists because using the semicolon as the argument delimiter would require special handling, and then EQW would exceed the 24K ASM limit.

Note that you can easily use *augment(,)*, with the comma delimiter, in EQW.

However, there are three ways that you can accomplish the *augment(;)* function in EQW. The first is to pass *augment(;)* as an input argument to EQW, the second method is to use the transpose function, and a third method is to create a form.

For example, if you start EQW with this expression:

```
eqw("augment(m1;m2)")
```

then EQW will start with the augment expression. You may edit the *m1* and *m2* arguments, apply additional functions or operators, or perform any other operation you want. When you pass *augment(;)* as an EQW input argument, it behaves like any other function. Of course, you cannot delete the semicolon.

The second method is to use the built-in matrix transpose operator  $^T$  with *augment(,)*. This method works because *augment(,)* adds the second matrix argument as columns. So, if we transpose each of the input arguments, then transpose the result, *m2* will be augmented to *m1* as rows, which is what *augment(;)* does. The expression to simulate *augment(m1;m2)* in EQW looks like this:

```
(augment(m1T,m2T))T
```

The third method is to create a form with the *augment(;)* function and placeholders. This example saves the form to a variable named *aug1*:

```
"augment("&char(255("&";"&char(255)&")")->aug1
```

You may start EQW with this form:

```
eqw(aug1)
```

and edit the placeholders for the two matrices as needed. You can also use a small program to recall this form within EQW; see *Using Forms* in the *Basic operations with EQW* section.

## Entering vectors in polar, spherical and cylindrical coordinates

A vector is a one-dimensional matrix of elements. A two-dimensional vector has two elements, and a three-dimensional vector has three elements. Vectors of either dimension may be row or column matrices. In addition, vectors can be specified in terms of rectangular, polar, spherical or cylindrical coordinates. These are all valid vectors:

Two-dimensional rectangular row vector:  $\begin{bmatrix} 3 & 4 \end{bmatrix}$

Two-dimensional rectangular column vector:  $\begin{bmatrix} 3 \\ 4 \end{bmatrix}$

Three-dimensional cylindrical row vector:  $\begin{bmatrix} 1 & \Delta 2 & 3 \end{bmatrix}$

Three-dimensional spherical column vector:  $\begin{bmatrix} 1 \\ \Delta 2 \\ \Delta 3 \end{bmatrix}$

EQW can create vectors of any dimension, in any coordinate system, but polar, spherical and cylindrical vectors require special handling.

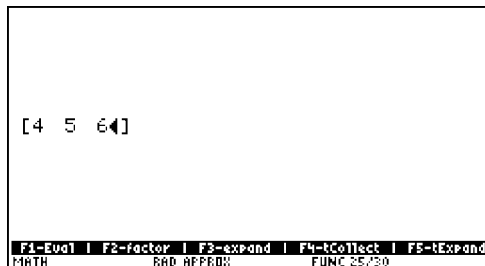
Entering a vector in rectangular coordinates is straightforward: simply use the  $\begin{bmatrix} \phantom{x} \end{bmatrix}$  key to create the vector as a matrix. For example, to create the three-dimensional rectangular vector [4,5,6]:

### Create a 3-dimensional rectangular vector

Start EQW and press  $\begin{bmatrix} \phantom{x} \end{bmatrix}$  to create a matrix.



Press [4] [,] [5] [,] [6] to create the vector.





We must use a different method to enter vectors with angular coordinates, because EQW will create a complex number form when the angle key [  $\angle$  ] is pressed to enter the angle. For example, to enter the vector [4,  $\angle$ 5] we might try the keystrokes [ [ ], [4], [  $\angle$  ], but this gives us



which is not what we want. EQW creates a complex number form when the angle key is pressed because EQW disallows the creation of invalid expressions, and the angle symbol, by itself, is not a valid expression.

In general, matrix entry in EQW is independent of the current vector format as defined in the [MODE] settings.

There are three methods which can be used to enter vectors with angular coordinates:

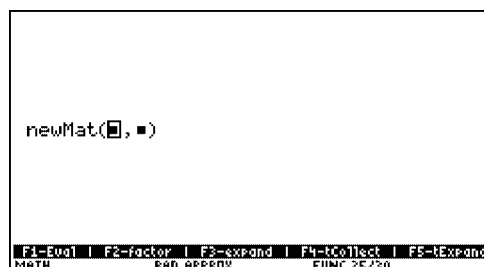
1. Use the *NewMat()* function, delete the defaults, then enter the elements. With this method, the vectors can be created directly, within EQW. However, the behaviour of the TI-89/92+ CAS requires several extra keypresses and deletions to enter the desired vector elements.
2. Use an input form. This method requires that the form be created and saved, but it need only be done once. This method is suitable for creating a single vector when starting EQW.
3. Define the vector template as a custom menu item defined in the *eqwuser()* program. This is the most efficient and effective method to enter any kind of vector. However, the *eqwuser()* program (distributed with EQW) must be installed in the \main folder.

The following examples show how to create the spherical vector [4, $\angle$ 5, $\angle$ 6] with each method.

### Create a vector with newMat()

Before starting EQW, press [MODE]. Set the Angle format to RADIAN; set the Vector Format to SPHERICAL

Start EQW. Press [CATALOG] [N]  
Use the [DOWN] key to select newMat(  
Press [ENTER]



Press [1] [ENTER] [3] to create a matrix with 1 row and 3 columns.  
 Press [2nd] [UP] to select the complete function.  
 Press [F1] to evaluate the function.

The TI-89/92+ CAS (not EQW!) creates this expression. We will delete the angle arguments, then enter the arguments we want.

The screen shows the initial vector expression:  $[ 0 < R * P0(0, 0) < \frac{\pi}{2} - \frac{\sin(w) \cdot \pi}{2} + \text{undef} \cdot i ]$ . The bottom status bar indicates: F1-Eval F2-factor F3-expand F4-tCollect F5-tExpand, MATH, RAD AUTO, FUNC 0/20.

Press [DOWN] [DOWN] to select the 0 element.  
 Press [BACKSPACE] [BACKSPACE] [4] to enter the first element.

The screen shows the first element replaced:  $[ 4 < R * P0(0, 0) < \frac{\pi}{2} - \frac{\sin(w) \cdot \pi}{2} + \text{undef} \cdot i ]$ . The bottom status bar indicates: F1-Eval F2-factor F3-expand F4-tCollect F5-tExpand, MATH, RAD AUTO, FUNC 0/20.

Press [RIGHT] [UP] [BACKSPACE] to mark and delete the first angle expression.  
 Press [5] to enter the angle argument at the placeholder.

The screen shows the first angle expression replaced:  $[ 4 < 5 < \frac{\pi}{2} - \frac{\sin(w) \cdot \pi}{2} + \text{undef} \cdot i ]$ . The bottom status bar indicates: F1-Eval F2-factor F3-expand F4-tCollect F5-tExpand, MATH, RAD AUTO, FUNC 0/20.

Press [RIGHT] [UP] [UP] [UP] [BACKSPACE] to mark and delete the second angle expression.  
 Press [6] to enter the angle argument at the placeholder.

The screen shows the second angle expression replaced:  $[ 4 < 5 < 6 < \frac{\pi}{2} - \frac{\sin(w) \cdot \pi}{2} + \text{undef} \cdot i ]$ . The bottom status bar indicates: F1-Eval F2-factor F3-expand F4-tCollect F5-tExpand, MATH, RAD AUTO, FUNC 0/20.

The vector is complete.

### Create a vector with an input form

We will use the method of the previous example, with newMat(), to create a vector with placeholders, and save it as an input form.

Before starting EQW, press [MODE]. Set the Angle format to RADIAN; set the Vector Format to SPHERICAL

Use newMat() to create a 1 x 3 matrix.

The screen shows the initial vector expression:  $[ 0 < R * P0(0, 0) < \frac{\pi}{2} - \frac{\sin(w) \cdot \pi}{2} + \text{undef} \cdot i ]$ . The bottom status bar indicates: F1-Eval F2-factor F3-expand F4-tCollect F5-tExpand, MAIN, RAD AUTO, FUNC 0/20.

Press [DOWN] [BACKSPACE] [BACKSPACE] to create the first placeholder.

The first screenshot shows the EQW interface with the first placeholder created. The second screenshot shows the second placeholder created. The third screenshot shows the third placeholder created. The final screenshot shows the completed vector form stored in a variable named *vecsph*.

Press [RIGHT] [UP] [BACKSPACE] to create the second placeholder.

Press [RIGHT] [UP] [UP] [UP] [BACKSPACE] to create the third placeholder.

Press [2nd] [UP] ["] to convert the matrix to an input form.

Press [STO>] [V] [E] [C] [S] [P] [H] [2nd] [UP] [F1] to store the form to a variable called *vecsph*.

The *vecsph* form can be used as any other form. Refer to *Using forms* in the section *Basic operations with EQW* for more details.

The following example shows how to use the input form to add two spherical vectors. The general idea is to start EQW with the vector form, then use the copy and paste features to create the second vector form. Before doing this example, press [MODE] and set Display Digits to FLOAT 3, set Angle to RADIAN, and set Vector Format to SPHERICAL.

### Add two vectors using input forms

Start EQW with the *vecsph* form by entering this at the command line:

EQW(*vecsph*)

(Note you may also need to specify the folder in which EQW is installed.)

The screenshot shows the EQW interface with the vector form being entered.

Copy the form before entering any elements, since we will want to enter another form later.

Press [2nd] [UP] [COPY]

Press [DOWN] [DOWN] to select the first element.  
To enter the elements of the first vector, press  
[1] [ENTER]  
[.] [7] [ENTER]  
[.] [2]

Press [2nd] [UP] to mark the vector.  
Press [+] to add the second vector.

Press [PASTE] to paste the input form.

Press [DOWN] [DOWN] to mark the first element.  
To enter the elements of the second vector, press  
[.] [9] [ENTER]  
[.] [5] [ENTER]  
[.] [6]

Press [2nd] [UP] [F1] to evaluate the expression.  
The result is shown.

Screen 1:  $[ ]$

Screen 2:  $[ 1 \ 7 \ 2 ]$

Screen 3:  $[ 1 \ 7 \ 2 ] + [ ]$

Screen 4:  $[ 1 \ 7 \ 2 ] + [ .9 \ 5 \ 6 ]$

Screen 5:  $[ 1.86 \ 7.556 \ 2.388 ]$

The next example shows that it is quite easy to create any type of vector, using a vector template in a custom menu.

## Create a vector with a custom menu

Before starting EQW, install the `eqwuser()` program in the `\main` folder. `eqwuser()` is distributed with EQW.

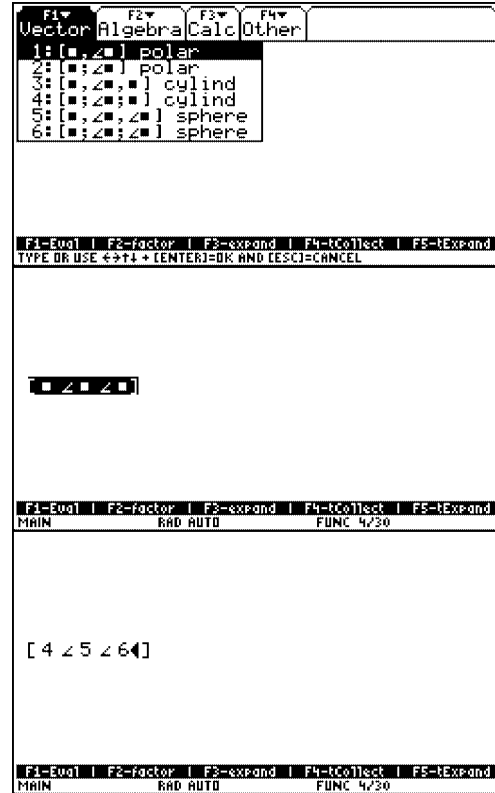
Press [APPS] to display the custom menu, then press [F1] to show the Vector menu.

Press [5] to choose the spherical vector form.

Press [DOWN] [DOWN] [4] to enter the first element.

Press [ENTER] [5] to enter the second element.  
Press [ENTER] [6] to enter the last element.

The vector is complete.



The fastest way to enter any type of vector is with the custom menu vector item. However, you may prefer one of the other two methods.

## Using EQW with the Y= editor

Even though EQW is not directly integrated into the Y= editor, it is easy to use EQW to enter equations into the editor. Since EQW returns the equation through the clipboard, use [PASTE] to insert the equation where you want it. This example shows the method.

### Use EQW with Y= editor

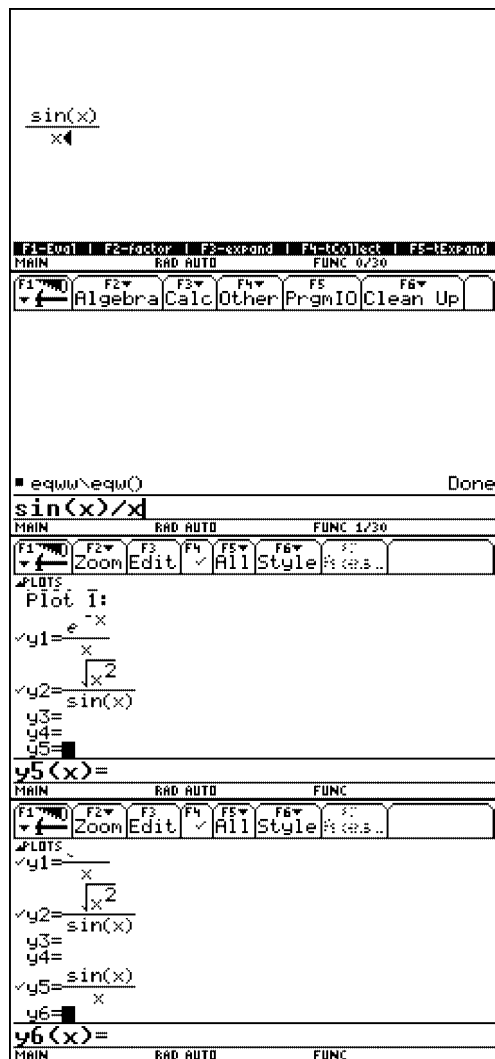
Start EQW.

Press [SIN] [x] [UP] [/] [x] to enter the expression to use in the Y= editor

Press [ENTER] to send the expression to the entry line. This also copies the expression to the clipboard.

Press [Y=] to start the Y= editor. Use [UP] and [DOWN] to select the desired Y= function. I use y5 in this example

Use [PASTE] to insert the expression from EQW into the entry line, then press [ENTER]. The expression is now defined as y5(x).



When you return to the home screen, the expression is still in the entry line. Press [CLEAR] to clear the expression.

### **Use a pop-up menu of history display answers in EQW**

You can use this simple program to paste answers from the home screen history display to EQW:

```
eqwprgm2()
Prgm
@Pop-up menu of history answers; by E.W.
@Changes by dburkett@infinet.com/7oct01

local var,lst,n,i,mnu

{}→lst           @list of history display answers
{}→mnu           @list of pop-up menu entries
0→var           @pop-up menu return variable

1→n             @list pointer to next valid history answer

for i,1,20       @Loop to try to get up to 20 history answers
  Try
    @Convert the next history element to a string
    string(expr("ans("&string(exact(i))&")"))→lst[n]
    @Truncate history element for menu display
    left(lst[n],15)→mnu[n]
    n+1→n        @Update pointer to next valid history element
  else
    clrerr       @Could not convert history element; just clear error code
  endtry
endfor

@ Display pop-up menu, if possible
if dim(mnu)>0 then @Display menu if there is at least one menu item
  popup mnu,var
  if var>0        @Send history element to EQW if user chose one
    lst[var]→main\eqwcom
  else           @Display error message if no valid history elements
    dialog
      title "HISTORY POP-UP"
      text "No history elements"
    enddlog
  endif
EndPrgm
```

This program was written by E.W. I have changed it so that it also works in Approx mode, and to display an error message if there are no valid history answers. This program is distributed with the EQW package, but the distributed version does not include the comments shown above. The program can be used with either the TI-89 or the TI-92+.

This program must be named *eqwprgm2*, and must be installed in the *\main* folder. You run the program within EQW by pressing [DIAMOND][2]. You can change which DIAMOND-key executes the program by changing the name. For example, if you want to run the program by pressing [DIAMOND] [4], change the name from *eqwprgm2* to *eqwprgm4*.

You can also call this program from a custom menu displayed by the *eqwuser()* program.

The history answer chosen from the pop-up menu will only be inserted if a placeholder (black square) is shown in EQW. The following example demonstrates using *eqwprgm2()* to build an expression.

### Use pop-up history menu with EQW

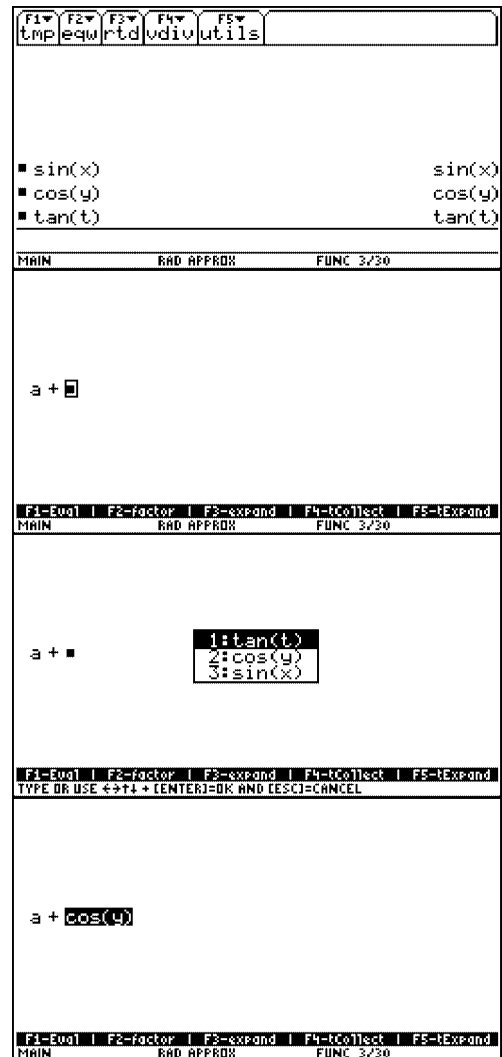
The figure at right shows the history display before EQW is started.

Start EQW. Press [a] [+] to start entering the expression. Note the black placeholder is selected.

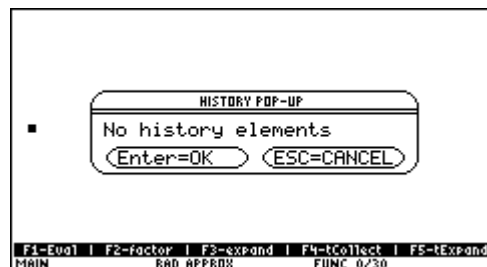
Press [DIAMOND] [2] to display the pop-up menu. Note that all three history expressions are shown in the menu. The expressions are short enough that they can be completely displayed in the menu. Longer expressions will be truncated to 15 characters.

Push [2] to choose  $\cos(y)$ , and that expression is inserted at the placeholder.

You can continue to enter the rest of the expression as needed. You can also press [DIAMOND] [2] again to insert more expressions from the history display.



There may be no valid expressions in the history display. If so, this dialog box is shown when [DIAMOND] [2] is pressed:



Press either [ENTER] or [ESC] to clear the dialog box and continue.



### **Create a form for d°m's" in an EQW program**

It is relatively easy to enter angles in degree, minute second format in EQW; refer to the example *Entering angles in degrees, minutes and seconds*. However, if you want to create a form for angles in this format, you must enter the form in the EQW program on the calculator, not in GraphLink. In addition, once you create the form in program, you *cannot* open the program in GraphLink, save the program, then send it to the calculator. Opening the program changes it so that a Syntax error occurs when you try to insert the form string.

As an example, consider this simple program which uses [DIAMOND] [8] to insert the angle form.

```
eqwprgm8()  
Prgm  
local v  
"ÿ°ÿ'ÿ""→v  
If when(main\eqwrun,true,false,false) Then  
  v→main\eqwcom  
Else  
  main\sendtext(v)  
EndIf  
EndPrgm
```

This program is created on the calculator in the *main\* folder. Press [APPS], [7] Program Editor, [3] New, to start entering the program. The form string will be stored in local variable v. The placeholder character ÿ can be created in the Home screen, like this:

char(255) [ENTER]	Create the placeholder string
[UP] [ENTER]	Copy the placeholder string to the entry line
[BACKSPACE] [LEFT] [BACKSPACE]	Delete the double quote characters
[SHIFT] [RIGHT]	Highlight the placeholder character
[COPY]	Copy the character to the clipboard (use [DIAMOND] [C] on the TI-92+)

After the placeholder character is copied to the clipboard, it is pasted three times in the program editor in the form string. Use [PASTE] on the TI-89, or [DIAMOND] [V] on the TI-92+. Note the three double quote characters at the end of the expression. Since we want to include a double quote in the string, we must use two characters.

This is a different way to create the form string, in *eqwprgm8()*, without creating and copying the placeholder character:

```
char(255)&"°"&char(255)&"' "&char(255)&"'"&""→v
```

You may use GraphLink to copy this program to a computer, but you must not use GraphLink to open the program file, or the Syntax Error will occur if you run the program after downloading it from the computer to the calculator.

This example shows how to use *eqwprgm8()* to enter an angle of 11°22'33".

Start EQW.

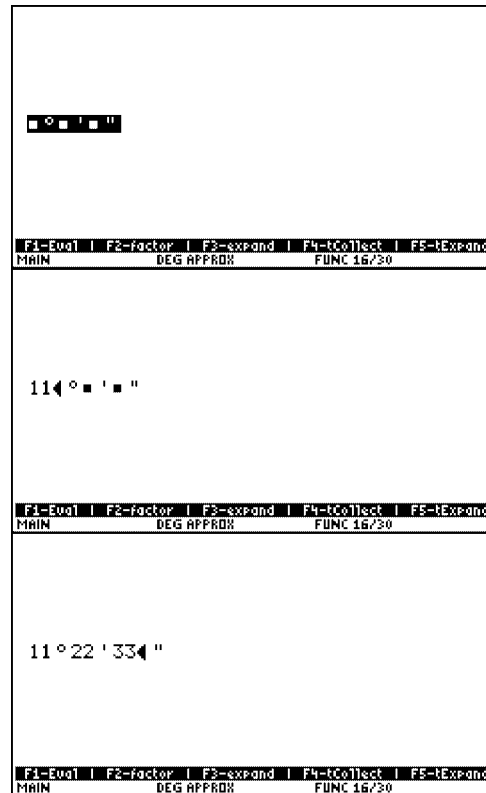
Press [DIAMOND] [8] to insert the form.

Press [DOWN] [1] [1] to select the first placeholder and enter 11 degrees.

Press [ENTER] [2] [2] to select the second placeholder and enter 22'

Press [ENTER] [3] [3] to select the third placeholder and enter 33".

The expression is complete.



---

## Fixing problems

---

This section describes solutions to some common problems that might occur while using EQW.

### "Syntax" error message

A *Syntax* error dialog box will be shown when you enter something that EQW doesn't understand. Since EQW uses the same built-in rules as the TI-89/92+ operating system, this means that the expression would not make sense outside of EQW, either. The solution to the syntax error is to clear the error and enter the correct expression. This solution doesn't help much, though, if you don't know what caused the error. This is difficult to describe because there are so many ways to enter an expression incorrectly. These operations can typically cause syntax errors:

1. Manually entering parenthesis where they are not needed. EQW almost always supplies its own matching parentheses, and if you try to add your own, the resulting expression is invalid. The solution is to let EQW supply the parentheses. You control *where* the parentheses are placed by appropriately marking the correct subexpression before applying the next function or operator.
2. Not using the correct capitalization for built-in functions and commands. EQW can automatically recognize built-in functions after you type them in and press [RIGHT] or [,], but the function name must be properly capitalized. For example, typing 'linreg' [SPACE] [RIGHT]] causes a syntax error, but typing 'LinReg' [SPACE] [RIGHT] creates the input form. To completely avoid this problem, use the CATALOG menu to enter the function names.

### "Too few arguments" error message

There are a variety of situations that can cause this error. For example, if you try to manually enter command arguments, without letting EQW recognize the command, EQW may decide that your command doesn't make any sense. The solution is to let EQW recognize the command or function, and insert the input form with place-holders.

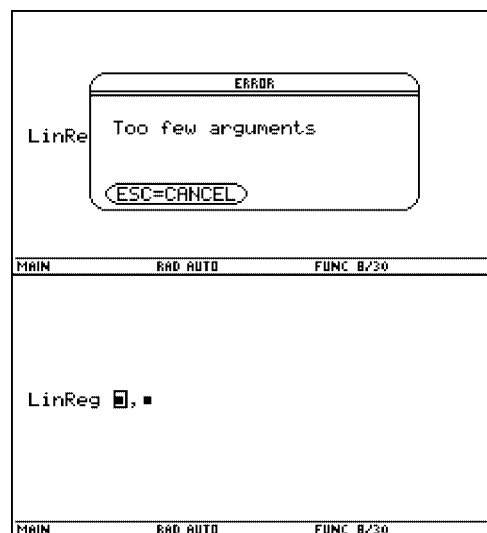
For example, suppose we want to enter the linear regression command *LinReg a,b*. This example shows one way that doesn't work, as well as the correct method.

The wrong way:

Press [L] [i] [n] [R] [e] [g] [SPACE] [a] [.,]  
The error message is shown.

The right way:

Press [L] [i] [n] [R] [e] [g] [SPACE] [RIGHT]  
EQW recognizes the command, and inserts a form with two place-holders, where you can enter *a* and *b*.



The key points are:

1. Press [RIGHT] after entering a built-in function or command name, so EQW can recognize it and insert an input form.
2. When typing in *commands*, you must supply the [SPACE] before pressing [RIGHT], just as you would if entering the command on the normal entry line.
3. When typing in *functions*, you must supply the ([) before pressing [RIGHT].

### "Missing (" error

This can happen if you type in a built-in function name, and you do not type in the parenthesis before pressing [RIGHT].

### EQW puts parentheses where I do not want them

EQW inserts parentheses as needed to create a sensible expression. EQW must operate this way, to prevent the creation of invalid expressions. However, you *do* have complete control over the parentheses: simply mark the expression before applying the next operation.

### I can't make the cursor go where I need to

This can be one of the most common, frustrating problems for new users. The dilemma is that the expression you see on the screen might not bear the slightest resemblance to the internal form of the expression. Variables and operators that are 'right next to each other' on the screen are not necessarily 'close' in the internal representation. Refer to the section *Basic operations in EQW* for more details and description.

Here is a method that will *always* work to move the cursor to the expression you want. Press [2nd] [UP] to mark the complete expression, then press [2nd] [DOWN]. This moves the outline box cursor to the first expression element. Now, press [RIGHT] until the cursor is where you want it. This is rarely the most efficient way to move the cursor, but it always works. For example:

I just entered this expression, but I realize that *c* should have been *x*. How do I get from *w* to *c*?

Press [2nd] [UP] to mark the complete expression.  
Press [2nd] [DOWN], and *a* is surrounded by the outline box.

The image shows two screenshots of the EQW calculator interface. The top screenshot displays a complex mathematical expression: 
$$\frac{(\sin(a) + \cos(b)) \cdot \frac{c+d}{e} \cdot \sqrt{f} + \tan\left(\frac{g \cdot h}{i+j}\right)}{\frac{\int_r \left( k \cdot \frac{1^m}{n-o \cdot p} \right) dq \cdot t}{u} \cdot (v+w)}$$
. The bottom screenshot shows the same expression after pressing [2nd] [UP] and [2nd] [DOWN]. The first element, 'a', is now enclosed in a small rectangular outline box, and the cursor is positioned at the start of 'a'. The status bar at the bottom of both screens reads 'MAIN RAD AUTO FUNC 7/30'.

Press [RIGHT] twice, so that c is selected with the outline box. Now press [BACKSPACE] [BACKSPACE] [x], and the correction is complete.

***I can't enter optional function arguments***

Many built-in functions can take optional arguments. For example, *det()* can be called as *det(m)* to find the determinant of matrix *m*, or it can be called as *det(m,tol)* where *tol* is the optional tolerance argument. When EQW creates a place-holder form for the function, it does not supply place-holders for the optional arguments. This is easily done with the [,] key. For example, to create the expression *det(m,.001)*:

Start EQW and press [d] [e] [t] [(] [RIGHT] to insert the form.

det(□)

MAIN RAD AUTO FUNC 7/30

Press [m] to enter the matrix name.

det(m□)

MAIN RAD AUTO FUNC 7/30

Press [,] to insert another place-holder

det(m, □)

MAIN RAD AUTO FUNC 7/30

Press [.] [0] [0] [1] to enter the tolerance. The expression is complete.

det(m, .001□)

MAIN RAD AUTO FUNC 7/30

Note that you could also have pressed [,] when the first place-holder was shown, instead of entering  $m$  first. In general, you can press [,] repeatedly to add as many place-holders as you need, then go back and fill them in with values. Or, you can enter the values one by one, and press [,] to add another place-holder.

### ***How do I change the direction in a limit?***

Refer to the section *Entering a limit* in the *Examples* section.

### ***EQW will not let me delete a place-holder***

EQW will not let you delete a place-holder if doing so would create an invalid expression. For example, the built-in command LinReg requires at least two input arguments. After you create the LinReg form, with its two place-holders, you cannot delete one of them.

### ***I cannot edit a fraction in EQW***

It is possible to create a numeric fraction in EQW which seems like it cannot be edited. This will happen if you evaluate the fraction, and is a result of the way the calculator internally represents numbers: a fraction is treated as a single number. However, you can still edit the fraction.

EQW actually handles two types of division, which I will call *common division* and *fraction division*. A number expressed with common division consists of a distinct numerator and denominator. A number expressed with fraction division is considered as a single number by the calculator AMS. If a fraction is expressed as common division, you can select the numerator and denominator, and edit them separately. If the number is expressed as a fraction division, the fraction is a single entity, and the numerator and denominator cannot be individually selected. However, it is possible in EQW to convert a fraction division to a common division, and then edit the numerator and denominator, by pressing [BACKSPACE]. The example below shows the different divisions.

Note that this distinction only arises when the numerator and denominator of the fraction are both integers (or integer expressions), and you have used [F1] to evaluate the expression.

Start EQW.

Press [3] [ / ] [7] to enter a fraction.

Press [UP] to select the entire fraction.

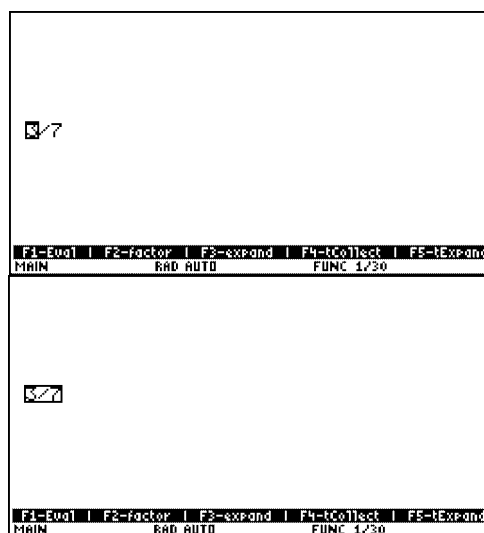
Note that you can use [LEFT] and [RIGHT] to select the numerator and denominator.

At this point, the fraction is a common division expression.

Press [2nd] [UP] to mark the entire fraction, then press [F1] to evaluate the expression.

The complete fraction is surrounded by the outline box. You cannot use [LEFT] or [RIGHT] to mark the numerator or denominator.

Evaluating the fraction has converted it to fraction division.



Press [BACKSPACE] to convert the fraction division to a common division. You can now use [LEFT] and [RIGHT] to select the numerator and denominator.



---

## Wishlist

---

This section lists changes suggested by the beta testers and EQW users. Initials in parentheses refer to the suggestor.

1. Evaluate functions [F2] - [F7] directly from the Help screen; (db). *This wish is less important now that [F1] - [F5] are shown as soft-menu functions at the bottom of the display.*
2. Display built-in TI-89/92+ help when a function is marked; (BB).
3. Allow cutting and pasting matrix rows; (BB).
4. Show source of syntax errors in expressions; (BB). *According to E.W., this is difficult or impossible.*
5. From within EQW, allow access to the built-in keys display, which is [DIAMOND] [k] on the TI-92+ and [DIAMOND] [EE] on the TI-89; (db).
6. Mark adjacent subexpressions, for example, mark  $(x+1)*(x+2)$  in  $(x+1)*(x+2)*(x+3)$ ; (BB). *According to E.W., this requires code that would exceed the 24K ASM limit.*
7. Start EQW from the history display, with the cursor in the history display; (PH). *According to E.W., this would require a hack into the OS which would reduce both reliability and compatibility with future AMS releases.*
8. Use a 'line-break' for expressions that are too long to fit on the screen, in one line, but can be shown completely with two or more lines; (db).
9. Display variable subscripts as true subscripts, for example,  $m[i,j]$  is displayed as  $m_{i,j}$ ; (AC).
10. Allow access to Mode screen, within EQW; (db). *This is less critical since mode settings can be made with EQW programs.*
11. Enable calling EQW from the graphing equation editor screen, that is, the Y= editor. This would make it easier to enter equations directly into the Y= editor. *This type of feature may be impossible with RAM programs, but may be possible with Flash applications.*