



# **TI-86** GRAPHING CALCULATOR **GUIDEBOOK**



TI-GRAPH LINK, Calculator-Based Laboratory, CBL, CBL 2, Calculator-Based Ranger, CBR, Constant Memory, Automatic Power Down, APD, and EOS are trademarks of Texas Instruments Incorporated.

Windows is a registered trademark of Microsoft Corporation.

IBM is a registered trademark of International Business Machines Corporation

Macintosh is a registered trademark of Apple Computer, Inc.

Copyright © 1997, 2001 by Texas Instruments Incorporated

## **Important**

Texas Instruments makes no warranty, either expressed or implied, including but not limited to any implied warranties of merchantability and fitness for a particular purpose, regarding any programs or book materials and makes such materials available solely on an “as-is” basis.

In no event shall Texas Instruments be liable to anyone for special, collateral, incidental, or consequential damages in connection with or arising out of the purchase or use of these materials, and the sole and exclusive liability of Texas Instruments, regardless of the form of action, shall not exceed the purchase price of this equipment. Moreover, Texas Instruments shall not be liable for any claim of any kind whatsoever against the use of these materials by any other party.

## **US FCC Information Concerning Radio Frequency Interference**

This equipment has been tested and found to comply with the limits for a Class B digital device, pursuant to Part 15 of the FCC rules. These limits are designed to provide reasonable protection against harmful interference in a residential installation. This equipment generates, uses, and can radiate radio frequency energy and, if not installed and used in accordance with the instructions, may cause harmful interference with radio communications. However, there is no guarantee that interference will not occur in a particular installation.

If this equipment does cause harmful interference to radio or television reception, which can be determined by turning the equipment off and on, you can try to correct the interference by one or more of the following measures:

- ◆ Reorient or relocate the receiving antenna.
- ◆ Increase the separation between the equipment and receiver.
- ◆ Connect the equipment into an outlet on a circuit different from that to which the receiver is connected.
- ◆ Consult the dealer or an experienced radio/television technician for help.

# Table of Contents

<b>TI-86 Quick Start</b>	<b>1</b>
Preparing to Use Your New TI-86 .....	2
Installing the AAA Batteries .....	2
Turning On and Turning Off the TI-86.....	2
Adjusting the Contrast .....	2
Resetting All Memory and Defaults.....	3
Calculating on the Home Screen.....	3
Calculating the Sine of a Number.....	3
Storing the Last Answer to a Variable.....	3
Using a Variable in an Expression .....	4
Editing an Expression.....	4
Displaying a Complex Number as a Result.....	5
Using a List with a Function .....	5
Displaying the Integer Part of Real Numbers in a List .....	6
Removing (Exiting) a Menu.....	6
Finding the Square Root.....	7
Calculating Derivatives.....	7
Retrieving, Editing, and Re-evaluating the Previous Entry .....	8
Converting Degrees Fahrenheit to Degrees Celsius.....	8
Storing an Unevaluated Expression to an Equation Variable .....	9
Plotting Functions on the Graph Screen.....	9

Displaying and Entering Functions in the Equation Editor.....	9
Changing the Graph Style of a Function.....	10
Plotting a Function on the Graph Screen.....	11
Tracing a Function.....	11
Evaluating y for a Specific x Value (During a Trace) .....	12
Changing a Window Variable Value.....	12
Deselecting a Function .....	13
Zooming In on a Portion of the Graph Screen .....	14

<b>Chapter 1: Operating the TI-86</b>	<b>15</b>
Installing or Replacing Batteries .....	16
When to Replace Batteries .....	16
Turning On and Turning Off the TI-86.....	17
Adjusting the Display Contrast.....	17
The Home Screen .....	18
Displaying Entries and Answers.....	18
Entering Numbers .....	19
Entering Negative Numbers .....	19
Using Scientific or Engineering Notation.....	20
Entering Complex Numbers.....	20
Entering Other Characters.....	21
The 2nd Key.....	21

The ALPHA Key.....	21
ALPHA-lock and alpha-lock.....	22
Common Cursors.....	22
Cursor Direction Keys.....	23
Inserting, Deleting, and Clearing Characters.....	23
Entering Expressions and Instructions.....	24
Entering an Expression.....	24
Using Functions in Expressions.....	25
Using an Instruction.....	25
Entering Functions, Instructions, and Operators.....	25
Entering Consecutive Entries.....	26
The Busy Indicator.....	26
Interrupting a Calculation or Graph.....	26
Diagnosing an Error.....	27
Correcting an Error.....	27
Reusing Previous Entries and the Last Answer.....	28
Retrieving the Last Entry.....	28
Retrieving and Editing the Last Entry.....	28
Retrieving Previous Entries.....	28
Retrieving Multiple Entries.....	29
Clearing the ENTRY Storage Area.....	29
Retrieving the Last Answer.....	29
Using Ans Preceding a Function.....	30
Storing Results to a Variable.....	30
Using TI-86 Menus.....	31

Displaying a Menu.....	31
The Menu Keys.....	32
Selecting a Menu Item.....	32
Exiting (Removing) a Menu.....	33
Viewing and Changing Modes.....	34
Changing a Mode Setting.....	34

## **Chapter 2:** **The CATALOG, Variables, and Characters**      **37**

The CATALOG.....	38
Storing Data to Variables.....	39
Creating a Variable Name.....	39
Storing a Value to a Variable Name.....	40
Storing an Unevaluated Expression.....	40
Storing an Answer.....	41
Copying a Variable Value.....	41
Displaying a Variable Value.....	41
Recalling a Variable Value.....	42
Classifying Variables as Data Types.....	42
The CATLG-VARS (CATALOG-Variables) Menu.....	43
Selecting a Variable Name.....	44
The CUSTOM Menu.....	44
Entering CUSTOM Menu Items.....	44
Clearing CUSTOM Menu Items.....	45
Deleting a Variable from Memory.....	45

The CHAR (Character) Menu.....	45
The CHAR MISC (Miscellaneous) Menu.....	46
The CHAR GREEK Menu.....	46
The CHAR INTL (International) Menu.....	46
Adding a Modifier to a Vowel.....	46

### Chapter 3:

#### Math, Calculus, and Test Operations 47

Keyboard Mathematical Functions.....	48
The MATH Menu.....	49
The MATH NUM (Number) Menu.....	49
The MATH PROB (Probability) Menu.....	50
The MATH ANGLE Menu.....	51
The MATH HYP (Hyperbolic) Menu.....	51
The MATH MISC (Miscellaneous) Menu.....	52
The Interpolate/Extrapolate Editor.....	53
The CALC (Calculus) Menu.....	54
The TEST (Relational) Menu.....	55
Using Tests in Expressions and Instructions.....	56

### Chapter 4: Constants,

#### Conversions, Bases, and Complex Numbers 57

Using Built-In and User-Created Constants.....	58
The CONS (Constants) Menu.....	58
The CONS BLTIN (Built-In Constants) Menu.....	58

Creating or Redefining a User-Created Constant.....	60
The Constant Editor Menu.....	60
Entering a Constant Name in an Expression.....	61
Converting Units of Measure.....	61
Converting a Unit of Measure.....	61
The CONV (Conversions) Menu.....	62
The CONV LENGH (Length) Menu.....	63
The CONV AREA Menu.....	63
The CONV VOL (Volume) Menu.....	63
The CONV TIME Menu.....	63
The CONV TEMP (Temperature) Menu.....	63
The CONV MASS Menu.....	64
The CONV FORCE Menu.....	64
The CONV PRESS (Pressure) Menu.....	64
The CONV ENRGY (Energy) Menu.....	64
The CONV POWER Menu.....	64
The CONV SPEED Menu.....	64
Converting a Value Expressed as a Rate.....	65
Number Bases.....	65
Number Base Ranges.....	66
One's and Two's Complements.....	66
The (Number) BASE Menu.....	66
The BASE A-F (Hexadecimal Characters) Menu.....	67
Entering Hexadecimal Digits.....	67
The BASE TYPE Menu.....	67

The BASE CONV (Conversion) Menu .....	68
Converting Number Bases .....	68
The BASE BOOL (Boolean) Menu .....	68
Results of Boolean Operations .....	69
The BASE BIT Menu.....	69
Using Complex Numbers.....	70
Complex Results.....	70
Using a Complex Number in an Expression.....	71
The CPLX (Complex Number) Menu .....	71

**Chapter 5: Function Graphing 73**

Defining a Graph.....	74
Setting the Graph Mode .....	74
The GRAPH Menu .....	75
Using the Equation Editor .....	76
The Equation Editor (GRAPH $y(x)=$ ) Menu .....	76
Defining a Function in the Equation Editor .....	77
Notes about Defining Function Equations.....	78
Selecting Graph Styles.....	79
Setting the Graph Style in the Equation Editor.....	80
Using Shading Patterns to Differentiate Functions.....	80
Viewing and Changing On/Off Status of Stat Plots .....	81
Setting the Window Variables.....	81
Displaying the Window Editor.....	82
Changing a Window Variable Value.....	82

Setting Graphing Accuracy with $\Delta x$ and $\Delta y$ .....	83
Setting the Graph Format.....	83
Displaying a Graph.....	85
Pausing or Stopping a Graph in Progress .....	85
Modifying a Drawn Graph.....	85
Graphing a Family of Curves .....	86
Smart Graph .....	86

**Chapter 6: Graph Tools 87**

Graph Tools on the TI-86 .....	88
The GRAPH Menu.....	88
Using the Free-Moving Cursor.....	89
Graphing Accuracy .....	89
Tracing a Graph.....	90
Stopping and Resuming a Trace .....	91
Resizing the Graph Screen with ZOOM Operations.....	91
The GRAPH ZOOM Menu .....	91
Defining a Custom Zoom In.....	93
Setting Zoom Factors.....	93
Zooming In and Zooming Out on a Graph.....	93
Storing and Recalling Zoom Window Variable Values.....	95
Using Interactive Math Functions .....	95
The GRAPH MATH Menu.....	95
Settings That Affect GRAPH MATH Operations .....	96
Using ROOT, FMIN, FMAX, or INFLC .....	97

Using $f(x)$ , DIST, or ARC .....	98
Using $dy/dx$ or TANLN .....	99
Using ISECT .....	100
Using YICPT .....	100
Evaluating a Function for a Specified $x$ .....	101
Drawing on a Graph .....	101
Before Drawing on a Graph .....	102
Saving and Recalling Drawn Pictures .....	102
Clearing Drawn Pictures .....	103
The GRAPH DRAW Menu .....	103
Shading Areas of a Graph .....	104
Drawing a Line Segment .....	105
Drawing a Vertical or Horizontal Line .....	106
Drawing a Circle .....	106
Drawing a Function, Tangent Line, or Inverse Function .....	107
Drawing Freehand Points, Lines, and Curves .....	107
Placing Text on a Graph .....	108
Turning On or Turning Off Points .....	108

## Chapter 7: Tables 109

Displaying the Table .....	110
TABLE Menu .....	110
The Table .....	110
Independent and Dependent Variables in the Table .....	111
Navigating the Table .....	111

The Table Menus .....	112
Setting Up the Table .....	113
Viewing and Editing Dependent Variable Equations .....	114
Clearing the Table .....	114

## Chapter 8: Polar Graphing 115

Preview: Polar Graphing .....	116
Defining a Polar Graph .....	117
Setting Polar Graphing Mode .....	117
The GRAPH Menu .....	117
Displaying the Polar Equation Editor .....	118
Setting the Graph Screen Window Variables .....	118
Setting the Graph Format .....	119
Displaying the Graph .....	119
Using Graph Tools in Pol Graphing Mode .....	119
The Free-Moving Cursor .....	119
Tracing a Polar Equation .....	120
Moving the Trace Cursor to a $\theta$ Value .....	121
Using Zoom Operations .....	121
The GRAPH MATH Menu .....	122
Evaluating an Equation for a Specified $\theta$ .....	122
Drawing on a Polar Graph .....	122

**Chapter 9: Parametric Graphing 123**

Preview: Parametric Graphing .....	124
Defining a Parametric Graph.....	125
Setting Parametric Graphing Mode.....	126
The GRAPH Menu.....	126
Displaying the Parametric Equation Editor.....	126
Selecting and Deselecting a Parametric Equation .....	127
Deleting a Parametric Equation.....	127
Setting the Graph Screen Window Variables.....	127
Setting the Graph Format.....	128
Displaying the Graph.....	128
Using Graph Tools in Param Graphing Mode.....	128
The Free-Moving Cursor.....	128
Tracing a Parametric Function.....	128
Moving the Trace Cursor to a t Value.....	129
Using Zoom Operations.....	129
The GRAPH MATH Menu.....	130
Evaluating an Equation for a Specified t .....	130
Drawing on a Parametric Graph.....	130

**Chapter 10: Differential Equation Graphing 131**

Defining a Differential Equation Graph.....	132
Setting Differential Equation Graphing Mode .....	132
The GRAPH Menu.....	133
Setting the Graph Format.....	133

Displaying the Differential Equation Editor .....	134
Setting the Graph Screen Window Variables.....	135
Setting the Initial Conditions .....	136
Setting the Axes .....	137
Differential Equation Graphing Tips .....	137
The Built-In Variable fldPic .....	138
Displaying the Graph.....	138
Entering and Solving Differential Equations.....	139
Graphing in SlpFld Format.....	139
Transforming an Equation into a First-Order System.....	140
Graphing in DirFld Format.....	141
Graphing a System of Equations in FldOff Format.....	142
Solving a Differential Equation for a Specified Value .....	144
Using Graph Tools in DifEq Graphing Mode .....	144
The Free-Moving Cursor .....	144
Tracing a Differential Equation.....	144
Moving the Trace Cursor to a t Value.....	145
Drawing on a Differential Equation Graph .....	145
Drawing an Equation and Storing Solutions to Lists.....	145
Using ZOOM Operations.....	147
Drawing Solutions Interactively with EXPLR.....	148
Evaluating Differential Equations for a Specified t .....	150



**Chapter 11: Lists 151**

Lists on the TI-86 .....	152
The LIST Menu .....	152
The LIST NAMES Menu .....	153
Creating, Storing, and Displaying Lists .....	153
Entering a List Directly in an Expression .....	153
Creating a List Name by Storing a List .....	154
Displaying List Elements Stored to a List Name .....	154
Displaying or Using a Single List Element .....	155
Storing a New Value to a List Element .....	155
Complex List Elements .....	156
The List Editor .....	156
The List Editor Menu .....	156
Creating a List Name in the Unnamed Column .....	157
Inserting a List Name into the List Editor .....	157
Displaying and Editing a List Element .....	158
Deleting Elements from a List .....	158
Removing a List from the List Editor .....	158
Using List Operations .....	159
The LIST OPS (Operations) Menu .....	159
Using Mathematical Functions with Lists .....	161
Attaching a Formula to a List Name .....	162
Comparing an Attached List with a Regular List .....	163
Using the List Editor to Attach a Formula .....	163
Using the List Editor With Attached-Formula Lists .....	164

Executing and Displaying Attached Formulas .....	164
Handling Errors Related to Attached Formulas .....	165
Detaching a Formula from a List Name .....	166
Editing an Element of a Attached Formula List .....	166

**Chapter 12: Vectors 167**

Vectors on the TI-86 .....	168
Creating, Storing, and Displaying Vectors .....	169
The VECTR (Vector) Menu .....	169
The VECTR NAMES Menu .....	169
Creating a Vector in the Vector Editor .....	169
The Vector Editor Menu .....	170
Creating a Vector on the Home Screen .....	170
Creating a Complex Vector .....	171
Displaying a Vector .....	171
Using a Vector in an Expression .....	172
Editing Vector Dimension and Elements .....	172
The VECTR MATH Menu .....	173
The VECTR OPS (Operations) Menu .....	173
The VECTR CPLX (Complex) Menu .....	175
Using Mathematical Functions with Vectors .....	176

**Chapter 13: Matrices 177**

Matrices on the TI-86 .....	178
Creating, Storing, and Displaying Matrices .....	178

The MATRX (Matrix) Menu .....	178
The MATRX NAMES Menu .....	178
Creating a Matrix in the Matrix Editor .....	178
The Matrix Editor Menu .....	179
Creating a Matrix on the Home Screen .....	180
Creating a Complex Matrix .....	180
Displaying Matrix Elements, Rows, and Submatrices .....	181
Using a Matrix in an Expression .....	181
Editing Matrices in the Matrix Editor .....	182
Editing Matrices on the Home Screen .....	182
The MATRX MATH Menu .....	183
The MATRX OPS (Operations) Menu .....	184
The MATRX CPLX (Complex ) Menu .....	185
Using Mathematical Functions with Matrices .....	185

## **Chapter 14: Statistics** **187**

Statistical Analysis on the TI-86 .....	188
Setting Up a Statistical Analysis .....	188
The STAT (Statistics) Menu .....	188
Entering Statistical Data .....	189
The LIST NAMES Menu .....	189
The STAT CALC (Calculations) Menu .....	189
Automatic Regression Equation Storage .....	191
Results of a Statistical Analysis .....	192
The STAT VARS (Statistical Variables) Menu .....	192

Plotting Statistical Data .....	194
The STAT PLOT Status Screen .....	194
The STAT PLOT Menu .....	195
Setting Up a Stat Plot .....	195
Turning On and Turning Off a Stat Plot .....	195
The PLOT TYPE Menu (Selecting a Plot Type) .....	196
Plot Type Characteristics .....	196
The STAT DRAW Menu .....	199
Forecasting a Statistical Data Value .....	199

## **Chapter 15: Equation Solving** **201**

Preview: The Equation Solver .....	202
Entering an Equation in the Equation-Entry Editor .....	203
Setting Up the Interactive-Solver Editor .....	204
Entering Variable Values .....	204
Controlling the Solution with Bounds and a Guess .....	204
Editing the Equation .....	205
The Solver Menu .....	206
Solving for the Unknown Variable .....	206
Graphing the Solution .....	207
Solver Graph Tools .....	207
The Solver ZOOM Menu .....	208
The Simultaneous Equation Solver .....	208
Entering Equations to Solve Simultaneously .....	208
Storing Equation Coefficients and Results to Variables .....	210
The Polynomial Root-Finder .....	211

Entering and Solving a Polynomial.....	211
Storing a Polynomial Coefficient or Root to a Variable .....	212
<b>Chapter 16: Programming</b>	<b>213</b>
Writing a Program on the TI-86 .....	214
The PRGM Menu .....	214
Creating a Program in the Program Editor .....	214
The Program Editor Menu .....	215
The PRGM I/O (Input/Output) Menu .....	215
The TI-86 Key Code Diagram .....	217
The PRGM CTL Menu .....	218
Entering a Command Line .....	220
Menus and Screens in the Program Editor .....	220
Running a Program .....	221
Breaking (Interrupting) a Program .....	222
Working with Programs .....	223
Managing Memory and Deleting a Program .....	223
Editing a Program.....	223
Calling a Program from Another Program.....	224
Copying a Program to Another Program Name.....	225
Using and Deleting Variables within a Single Program .....	225
Running an Assembly Language Program .....	225
Entering and Storing a String .....	226
The STRNG (String) Menu .....	227
Creating a String .....	227

<b>Chapter 17: Memory Management</b>	<b>229</b>
Checking Available Memory .....	230
The MEM (Memory) Menu .....	230
Checking Memory Usage.....	230
Deleting Items from Memory .....	231
The MEM DELET (Delete) Menu .....	231
Resetting the TI-86 .....	232
The MEM RESET (Reset) Menu.....	232
ClrEnt (Clear Entry).....	232
<b>Chapter 18: The TI-86 Communication Link</b>	<b>233</b>
TI-86 Linking Options.....	234
Linking Two TI-86s.....	234
Linking a TI-86 and a TI-85 .....	234
Linking a TI-86 and a CBL 2/CBL or CBR System.....	234
Linking a TI-86 and a PC or Macintosh .....	235
Downloading Programs from the Internet.....	235
Connecting the TI-86 to Another Device.....	235
The LINK Menu.....	236
Selecting Data to Send.....	236
The LINK SEND Menu .....	236
Initiating a Memory Backup .....	237
Selecting Variables to Send .....	238
The SEND WIND (Window Variables) Screen.....	238
Sending Variables to a TI-85 .....	239

The LINK SND85 (Send Data to TI-85) Menu .....	239
Preparing the Receiving Device.....	240
Transmitting Data .....	240
Receiving Transmitted Data .....	241
Repeating Transmission to Several Devices .....	242
Error Conditions .....	242
Insufficient Memory in Receiving Unit.....	242

**Chapter 19: Applications 243**

Using Math Operations with Matrices .....	244
Finding the Area between Curves .....	245
The Fundamental Theorem of Calculus.....	246
Electrical Circuits.....	248
Program: Taylor Series .....	250
Characteristic Polynomial and Eigenvalues.....	252
Convergence of the Power Series .....	254
Reservoir Problem.....	256
Predator-Prey Model.....	258
Program: Sierpinski Triangle .....	260

**Chapter 20:**

**A to Z Function and Instruction Reference 261**

Quick-Find Locator .....	262
Alphabetical Listing of Operations .....	266

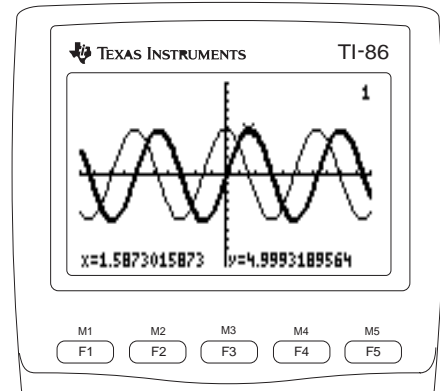
**Appendix 379**

TI-86 Menu Map .....	380
Handling a Difficulty.....	392
Error Conditions .....	393
Equation Operating System (EOS™).....	397
Implied Multiplication .....	397
Parentheses .....	397
TOL (The Tolerance Editor).....	398
Computational Accuracy .....	399
Support and Service Information.....	400
Product Support.....	400
Product Service.....	401
Other TI Products and Services.....	401
Warranty Information.....	402
Customers in the U.S. and Canada Only.....	402
Australia & New Zealand Customers Only.....	403
All Customers outside the U.S. and Canada .....	404

**Index**

# TI-86 Quick Start

Preparing to Use Your New TI-86 .....2  
Calculating on the Home Screen.....3  
Plotting Functions on the Graph Screen ..... 9



## Preparing to Use Your New TI-86

The brief examples in the TI-86 Quick Start demonstrate some common TI-86 features. Before you begin, you must install the batteries, turn on the calculator, adjust the contrast, and reset the memory and the defaults. Chapter 1 has more details on these topics.

### Installing the AAA Batteries

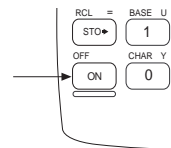
Four AAA batteries are included in the TI-86 retail package. Remove the batteries from the package and install them in the battery compartment on the back of the calculator. Arrange the batteries according to the polarity (+ and -) diagram in the battery compartment.

### Turning On and Turning Off the TI-86

*After about four minutes of inactivity, the TI-86 turns off automatically.*

To turn on the TI-86, press **[ON]**, which is in the bottom-left corner of the keyboard. You should see the entry cursor (■) blinking in the top-left corner of the screen. If you do not see it, adjust the contrast (see below).

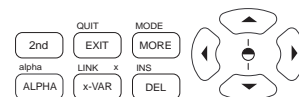
To turn off the calculator, press **[2nd]**, and then the key under OFF, which is **[ON]**. This guidebook uses brackets ( [ and ] ) to express **[2nd]** and **[ALPHA]** keystroke combinations. For example, to turn off the TI-86, press **[2nd]** **[OFF]**.



### Adjusting the Contrast

*If you release **[▲]** or **[▼]** while adjusting the contrast, you must press **[2nd]** again to continue the adjustment.*

- ❶ Press and release the yellow **[2nd]** key.
- ❷ Press and hold **[▲]** or **[▼]** (above or below the half-shaded circle).
  - ◆ To darken the screen contrast, press and hold **[▲]**.
  - ◆ To lighten the screen contrast, press and hold **[▼]**.



## Resetting All Memory and Defaults

To reset all memory and defaults, press  $\boxed{2\text{nd}} \boxed{[\text{MEM}]} \boxed{[\text{F3}]} \boxed{[\text{F1}]} \boxed{[\text{F4}]}$ . The messages **Mem cleared** and **Defaults set** are displayed on the home screen, confirming that all memory and defaults are reset. You may need to adjust the contrast after memory and default reset.

## Calculating on the Home Screen

To replicate the screens shown in the Quick Start activities, reset all memory and defaults once before you begin. Before doing an activity, press  $\boxed{[\text{CLEAR}]}$  to clear the screen (except before the entry retrieval and integer-part examples). Otherwise, the screens your TI-86 shows may differ from the screens pictured next to the activities.

To express  $\boxed{2\text{nd}}$  and  $\boxed{[\text{ALPHA}]}$  keystroke combinations, this guidebook places brackets ( $[$  and  $]$ ) around the word above the key to press.

The TI-86 on-screen division symbol is a forward slash ( $/$ ), as in a fraction.

Following evaluation, the entry cursor automatically moves to the next line, ready for a new entry.

When the TI-86 evaluates an expression, it automatically stores the answer to the built-in variable **Ans**, replacing any previous value.

### Calculating the Sine of a Number

- 1 Enter the sine function.
- 2 Enter a value. You can enter an expression, which is evaluated when you press  $\boxed{[\text{ENTER}]}$ .
- 3 Evaluate the problem. The evaluation of the expression  $\text{sine}(\pi/4)$  is displayed.

$\boxed{[\text{CLEAR}]} \boxed{[\text{SIN}]}$

sin █

$\boxed{[\square]} \boxed{2\text{nd}} \boxed{[\pi]} \boxed{[\div]} \boxed{4} \boxed{[\square]}$

sin ( $\pi/4$ ) █

$\boxed{[\text{ENTER}]}$

sin ( $\pi/4$ )  
 .707106781187 █

### Storing the Last Answer to a Variable

- 1 Paste the store symbol ( $\rightarrow$ ) to the screen. Since a value must precede  $\rightarrow$ , but you did not enter a value, the TI-86 automatically pasted **Ans** before  $\rightarrow$ . (continued)

$\boxed{[\text{CLEAR}]}$

$\boxed{[\text{STO} \rightarrow]}$

Ans  $\rightarrow$  █

When ALPHA-lock is on and you press a key, the letters printed in blue above the keys are pasted to the screen. In the example, press  $\boxed{2}$  to enter a **V**.

- ② Enter the variable name to which you want to store the last answer. ALPHA-lock is on.
- ③ Store the last answer to the variable. The stored value is displayed on the next line.

[V]

Ans→V

[ENTER]

Ans→V  
.707106781187

### Using a Variable in an Expression

- ① Enter the variable, and then square it.
- ② Evaluate. The value stored to the variable **V** is squared and displayed.

[(CLEAR)]

V<sup>2</sup>[ALPHA] [V] [x<sup>2</sup>]

[ENTER]

V<sup>2</sup>  
.5

### Editing an Expression

- ① Enter the expression  $(25+14)(4-3.2)$ .
- ② Change **3.2** to **2.3**.
- ③ Move the cursor to the beginning of the expression and insert a value. The insert cursor blinks between **3** and **25**.
- ④ Evaluate. The result is displayed.

[(CLEAR)]

(25+14)(4-3.2)

[C] 25 [+ ] 14 ]

[C] 4 [- 3 . 2 ]

[←] [←] [←] [←] 2 [→] 3

(25+14)(4-2.3

[2nd] [←] [2nd] [INS] 3

3\_25+14)(4-2.3)

[ENTER]

3(25+14)(4-2.3)  
198.9

You need not move the cursor to the end of the line to evaluate the expression.



- $\square$  negates a value, as in  $-2$ .  
 $\square$  subtracts, as in  $5-2=3$ .

An ellipsis (...) indicates that the result continues beyond the screen.

## Displaying a Complex Number as a Result

- 1 Enter the natural log function.
- 2 Enter a negative number.
- 3 Evaluate. The result is displayed as a complex number.

(CLEAR) LN

In

$\square$   $\square$  2  $\square$

In (-2)

ENTER (press  $\square$  to display more)

In (-2)  
(.69314718056, 3.1415...

## Using a List with a Function

- 1 Enter the exponential function.
- 2 Display the LIST menu, and then select the open brace ( { ) from the LIST menu.  
On the TI-86, { specifies the beginning of a list.
- 3 Enter the list elements. Separate each element from the next with a comma.
- 4 Select the close brace ( } ) from the LIST menu to specify the end of the list.
- 5 Evaluate. The results of the constant  $e$  raised to the 5th, 10th, and 15th powers are displayed as list elements.

(CLEAR) 2nd  $[e^x]$

$e^{\square}$

2nd LIST

$e^{\{ \square$

F1

LIST menu  $\rightarrow$

$\square$   
< > NAMES EDIT OPS

5  $\square$  10  $\square$  15

$e^{\{5,10,15 \square$

F2

$e^{\{5,10,15\}$   
(148.413159103 22026...

ENTER (press  $\square$  to display more)

< > NAMES EDIT OPS

## Displaying the Integer Part of Real Numbers in a List

- ① Display the MATH menu. (The MATH menu automatically replaces the LIST menu from the last activity.)

$\boxed{2\text{nd}}$  [MATH]

MATH menu →

- ② Select **NUM** to display the MATH NUM menu. The MATH menu shifts up.

$\boxed{F1}$

MATH NUM menu →

- ③ Select the **iPart** (integer part) function from the MATH NUM menu. **iPart** is pasted to the screen. (The previous entry was left on the screen to illustrate the effect of **iPart** on the previous answer.)

$\boxed{F2}$

- ④ Paste **Ans** to the cursor location. (The result list from the previous activity is stored to **Ans**.)

$\boxed{2\text{nd}}$  [ANS]

- ⑤ Display the integer part of the result list elements from the previous activity.

$\boxed{\text{ENTER}}$

## Removing (Exiting) a Menu

- ① In the previous example, the MATH menu and the MATH NUM menu are displayed ( $\boxed{2\text{nd}}$  [MATH]  $\boxed{F1}$ ).

- ② Remove the MATH NUM menu from the screen.

$\boxed{\text{EXIT}}$

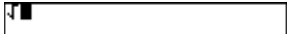
- ③ Remove the MATH menu from the screen.

$\boxed{\text{EXIT}}$


## Finding the Square Root

- 1 Paste the square root function to the screen.
- 2 Enter a value for which you want to find the square root.
- 3 Evaluate the expression. The square root of 144 is displayed.

(CLEAR) [2nd] [ $\sqrt{\quad}$ ]



144



[ENTER]



## Calculating Derivatives

- 1 Display the CALC menu, and then select **der1**.
- 2 Enter an expression ( $x^2$ ) with respect to a variable ( $x$ ) at a given point (**8**).
- 3 Evaluate. The first derivative of  $x^2$  with respect to  $x$  at 8 is displayed.

(CLEAR)

[2nd] [CALC]

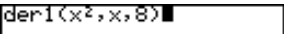
[F3]

CALC menu →

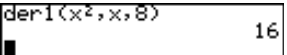


[x-VAR] [ $x^2$ ] [, ] [x-VAR]

[,] [8] [)]



[ENTER]



When you press  $\boxed{\text{ENTER}}$ , the TI-86 stores the expression or instruction you entered to the built-in memory storage area called ENTRY.

### Retrieving, Editing, and Re-evaluating the Previous Entry

- 1 Retrieve the last entry from the previous example. (The last activity was not cleared.)  $\boxed{2\text{nd}} \boxed{\text{ENTRY}}$
- 2 Edit the retrieved entry.  $\boxed{\leftarrow} \boxed{\leftarrow} \boxed{3}$
- 3 Evaluate. The first derivative of  $x^2$  with respect to  $x$  at **3** is displayed.  $\boxed{\text{ENTER}}$

```
der1(x^2,x,8)
der1(x^2,x,8) 16
```

```
der1(x^2,x,8)
der1(x^2,x,3 16
```

```
der1(x^2,x,8) 16
der1(x^2,x,3) 6
█
```

### Converting Degrees Fahrenheit to Degrees Celsius

- 1 Display the CONV menu.  $\boxed{(\text{CLEAR})} \boxed{2\text{nd}} \boxed{\text{CONV}}$
  - 2 Display the CONV TEMP menu. The CONV menu shifts up and **TEMP** is highlighted.  $\boxed{\text{F5}}$
  - 3 Enter the known measurement. If the measurement is negative, use parentheses. In this example, if you omit parentheses, the TI-86 converts  $4^\circ\text{F}$  to about  $-15.5^\circ\text{C}$ , which it then negates (changes the sign of), returning a positive  $15.5^\circ\text{C}$ .  $\boxed{(} \boxed{-} \boxed{4} \boxed{)}$
  - 4 Select  $^\circ\text{F}$  to designate Fahrenheit as the known measurement unit.  $^\circ\text{F}$  and the conversion symbol ( $\blacktriangleright$ ) are displayed after the measurement.  $\boxed{\text{F2}}$
- (continued)*

```
LNGLTH AREA VOL TIME TEMP
LNGLTH AREA VOL TIME TEMP
°C °F °K °R
```

```
(-4) █
LNGLTH AREA VOL TIME TEMP
°C °F °K °R
```

```
(-4) °F ▶
LNGLTH AREA VOL TIME TEMP
°C °F °K °R
```

When expressing a measurement for a conversion, you do not enter a unit symbol manually. For example, you need not enter  $^\circ$  to designate degrees.

- 5 Select °C to designate Celsius as the unit to which you want to convert.

F1

- 6 Convert. The °C equivalent of -4°F is displayed.

ENTER

### Storing an Unevaluated Expression to an Equation Variable

When storing to an equation variable using =, enter the equation variable first, then =, and then the unevaluated expression. This is the opposite from the order for storing to most other variables on the TI-86.

- 1 Enter the built-in equation variable **y1**. (CLEAR) 2nd [alpha] [Y] 1

- 2 Enter the equals sign (=).

ALPHA [=]

- 3 Enter an expression in terms of **x**.

5 ( ) SIN [x-VAR] ( )

- 4 Store the expression.

ENTER

The next section shows how to graph the functions  $y1=5(\sin x)$  and  $y2=5(\cos x)$ .

## Plotting Functions on the Graph Screen

The TI-86 plots four types of functions on the graph screen. To plot a graph, you must store an unevaluated expression to a built-in equation variable.

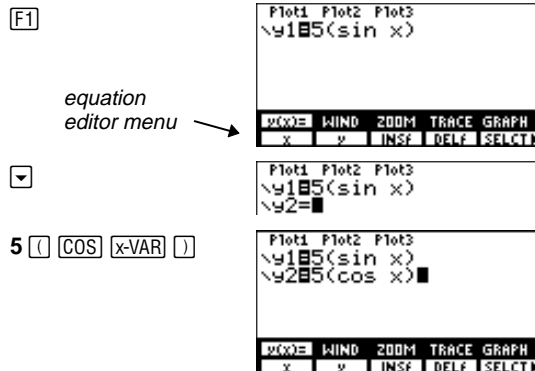
Each activity in this section builds upon the activity that precedes it. You must start here and perform the activities in the sequence in which they are presented. The first activity in this section assumes you are continuing from the last activity in the previous section.

### Displaying and Entering Functions in the Equation Editor

- 1 Display the GRAPH menu. (continued) GRAPH


In the equation editor, you must express each equation in terms of the independent variable  $x$  (in Func graphing mode only; Chapter 5).

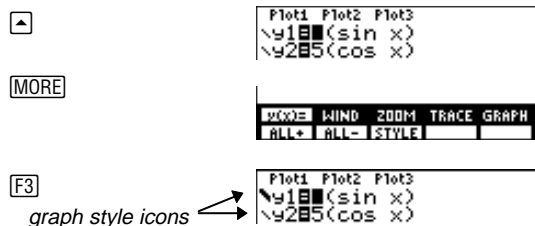
- 2 Select  $y(x)=$  from the GRAPH menu to display the equation editor.  $5(\sin x)$  is the unevaluated expression stored to  $y1$  in the previous activity. The equation editor menu is displayed as the lower menu.
- 3 Move the cursor down. The  $y2=$  prompt is displayed.
- 4 Enter the expression  $5(\cos x)$  at the  $y2=$  prompt. Notice that the equals sign (=) of  $y2$  is highlighted after you enter 5. Also, the equals sign of  $y1$  is highlighted. This indicates that both equations are selected to be graphed (Chapter 5).



### Changing the Graph Style of a Function

In the equation editor, the icon to the left of each equation specifies the style in which the graph of that equation appears when you plot it on the graph screen.

- 1 Move the cursor to  $y1$ .
- 2 Display the next menu group of the equation editor menu. (▶ at the end of a menu group indicates that the menu has more items.)
- 3 Select **STYLE** from the equation editor menu to set  (thick) graph style for  $y1$ .

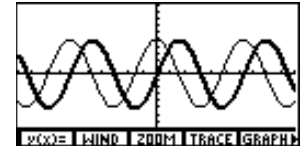


To display up to seven graph styles, depending on the graphing mode, repeat **F3**.

## Plotting a Function on the Graph Screen

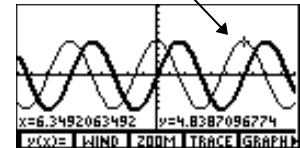
- 1 Select **GRAPH** from the GRAPH menu to plot the graph on the graph screen. The  $x$ - and  $y$ -axes and GRAPH menu are displayed. Then each selected graph is plotted in the order in which it is listed in the equation editor.
- 2 When the graph is plotted, you can move the free-moving cursor ( + ) around the graph screen. The cursor coordinates are displayed at the bottom of the graph.

[2nd] [M5]



[▶] [▼] [◀] [▲]

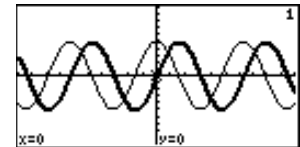
free-moving cursor



## Tracing a Function

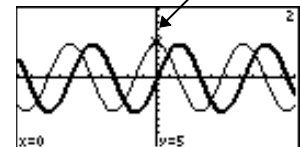
- 1 Select **TRACE** from the GRAPH menu to activate the trace cursor, with which you can trace along the graph of any selected function. The number of the current function (the 1 in **y1**) is displayed in the top-right corner.
- 2 Move the trace cursor from the function **y1** to the function **y2**. The 1 in the top-right corner changes to 2; the  $y$  value changes to the value of **y2** at  $x=0$ . (continued)

[F4]



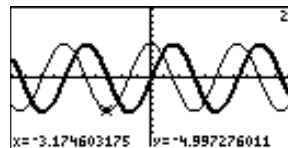
[▲]

trace cursor



- ③ Trace the function **y2**. As you trace, the displayed **y** value is the solution for **5(cos x)** at the current **x** value, which also is displayed on the screen.

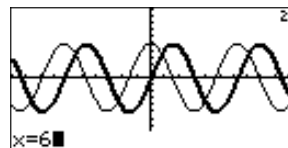
▶ and ◀



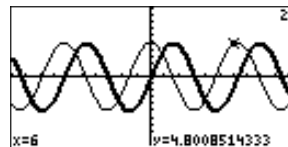
### Evaluating **y** for a Specific **x** Value (During a Trace)

- ① Enter a real number (or an expression that resolves to a real number) that is within the dimensions of the current graph screen. When you enter the first character, the **x=** prompt is displayed.
- ② Evaluate **y2** at **x=6**. The trace cursor moves directly to the solution. The **y** value, or solution of the equation at **x**, is displayed on the screen.

6



ENTER



### Changing a Window Variable Value

- ① Display the GRAPH menu.
- ② Select **WIND** from the GRAPH menu to display the window editor.

GRAPH

F2



*The window variables values determine the dimensions of the graph screen.*

(continued)

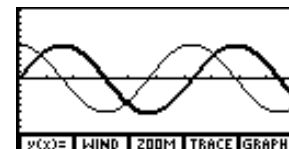


- ③ Change the value stored in the **xMin** window variable to **0**.
- ④ Plot the graph on the redefined graph screen. Since **xMin=0**, only the first and fourth quadrants of the graph plane are displayed.

0

WINDOW  
xMin=0

[F5]



### Deselecting a Function

- ① Select **y(x)=** from the GRAPH menu to display the equation editor and equation editor menu. The GRAPH menu shifts up and **y(x)=** is highlighted.
- ② Select **SELCT** from the equation editor menu to deselect the function **y1=**. The equals sign is no longer highlighted.
- ③ Plot the graph on the graph screen. Since you deselected **y1**, the TI-86 only plots **y2**. To select a function in the equation editor, repeat these steps. (**SELCT** both selects and deselects functions.)

[F1]

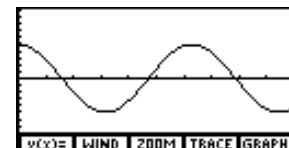
Plot1 Plot2 Plot3  
y1=(sin x)  
y2=5(cos x)

x(x)= WIND ZOOM TRACE GRAPH  
x y INSP DELF SELCT

[F5]

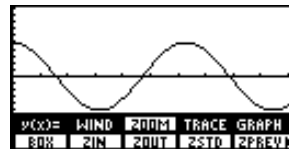
Plot1 Plot2 Plot3  
y1=(sin x)  
y2=5(cos x)

[2nd] [M5]

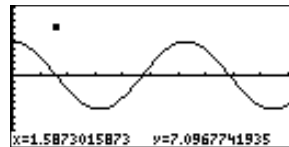


## Zooming In on a Portion of the Graph Screen

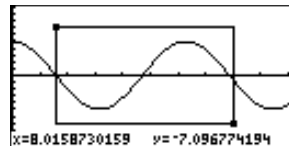
- ① Select **ZOOM** to display the GRAPH ZOOM menu. The GRAPH menu shifts up and **ZOOM** is highlighted. [F3]



- ② Select **BOX** from the GRAPH ZOOM menu to activate the zoom-box cursor. [F1]

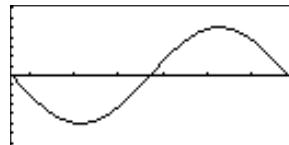


- ③ Move the zoom-box cursor to a point that is to be a corner of the redefined graph screen, and then mark the point with a small square. [Right] [Down] [Left] [Up]  
[ENTER]



- ④ Move the cursor away from the small square to a point that is to be the opposite corner of the redefined graph screen. As you move the cursor, a rectangle is drawn on the graph. [Right] [Down] [Left] [Up]

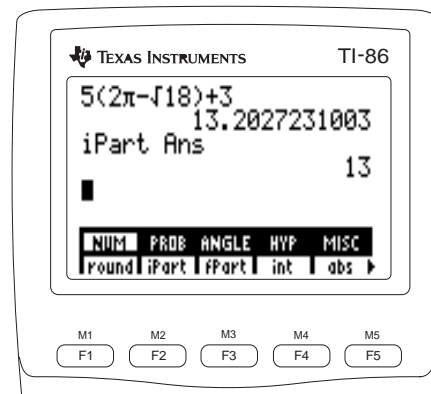
- ⑤ Zoom in on the graph. The window variables change automatically to the specifications of the zoom box. [ENTER]



- ⑥ Clear the menus from the graph screen. [CLEAR]

# 1 Operating the TI-86

Installing or Replacing Batteries .....	16
Turning On and Turning Off the TI-86 .....	17
Adjusting the Display Contrast .....	17
The Home Screen .....	18
Entering Numbers .....	19
Entering Other Characters .....	20
Entering Expressions and Instructions .....	24
Diagnosing an Error .....	27
Reusing Previous Entries and the Last Answer .....	28
Using TI-86 Menus .....	31
Viewing and Changing Modes .....	34



## Installing or Replacing Batteries

Your new TI-86 includes four AAA alkaline batteries. You must install them before you can turn on the calculator. A lithium backup battery is installed in the calculator already.

To express `[2nd]` and `[ALPHA]` keystroke combinations, this guidebook places brackets ( [ and ] ) around the word above the key to press.

Do not remove the lithium backup battery unless four fresh AAA batteries are in place.

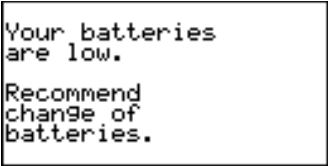
Properly dispose of the old batteries.

- 1 If the calculator is on, turn it off (press `[2nd]` `[OFF]`) to avoid loss of information stored in memory.
- 2 Slide the protective cover over the keyboard.
- 3 Holding the calculator upright, push down on the battery cover latch, and then remove the cover.
- 4 Remove all four old batteries.
- 5 Install four new AAA alkaline batteries, arranged according to the polarity (+ and -) diagram inside the battery compartment.
- 6 Replace the battery cover by inserting the two prongs into the two slots at the bottom of the battery compartment, and then push the cover until the latch snaps closed.

### When to Replace Batteries

If you do not use your TI-86 frequently, the AAA batteries could last more than two weeks after the first low-battery message.

When the AAA batteries are low, a low-battery message is displayed as you turn on the calculator. Generally, the calculator will continue to operate for one or two weeks after the low-battery message is first displayed. Eventually, the TI-86 will turn off automatically and will not operate until you replace the AAA batteries.



```
Your batteries  
are low.  
Recommend  
change of  
batteries.
```

The lithium backup battery is inside the battery compartment, above the AAA batteries. It retains all memory when the AAA batteries are low or have been removed. To avoid loss of data, do not remove the lithium battery unless four fresh AAA batteries are installed. Replace the lithium backup battery about every three or four years.

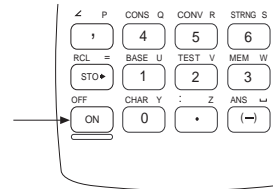
Properly dispose of the old battery.

To replace the lithium backup battery, remove the battery cover and unscrew the tiny screw holding the BACK UP BATTERY cover in place. Install a new CR1616 or CR1620 battery according to the polarity (+ and -) diagram on the cover. Replace the cover and screw.

## Turning On and Turning Off the TI-86

To turn on the TI-86, press **[ON]**.

- ◆ If you previously had turned off the calculator by pressing **[2nd]** **[OFF]**, the TI-86 clears any errors and displays the home screen as it was last displayed.
- ◆ If Automatic Power Down™ (APD™) previously had turned off the calculator, the TI-86 will return as you left it, including the display, cursor, and any error.



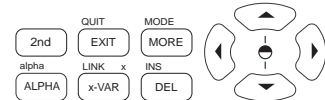
To turn off the TI-86 manually, press **[2nd]** **[OFF]**. All settings and memory contents are retained by the Constant Memory™ feature. Any error condition is cleared.

APD turns off the TI-86 automatically after about four minutes of non-use to extend battery life.

## Adjusting the Display Contrast

If you release **[↑]** or **[↓]** while adjusting the contrast, you must press **[2nd]** again to continue the adjustment.

- ① Press and release the yellow **[2nd]** key.
- ② Press and hold **[↑]** or **[↓]** (above or below the half-shaded circle).
  - ◆ To darken the screen contrast, press and hold **[↑]**.
  - ◆ To lighten the screen contrast, press and hold **[↓]**.



*The TI-86 has 40 contrast settings, so each number 0 through 9 represents four settings.*

You can adjust the display contrast anytime to suit your viewing angle and lighting conditions. As you adjust, a number from **0** (lightest) to **9** (darkest) in the top-right corner indicates the current contrast setting. The number is not visible when the contrast is extremely light or dark.

As the batteries weaken over time, the actual contrast level of each number shifts. For example, say you set the contrast to **3** with fresh batteries. As the batteries weaken, you will need to set the contrast to **4**, then **5**, then **6**, and so on, to retain the original contrast level. However, you need not replace the batteries until the low-battery message is displayed.

## The Home Screen

When you first turn on your TI-86, the home screen is displayed. Initially, the home screen is a blank screen, except for the entry cursor (■) in the top-left corner. If you do not see the cursor, press **[2nd]**, and then press and hold **[↓]** or **[↑]** to adjust the contrast (page 17).

On the home screen, you can enter and evaluate expressions, and view the results. You also can execute instructions, store and recall variable values, and set up graphs and editors.

To return to the home screen from any other screen, press **[2nd]** **[QUIT]**.

## Displaying Entries and Answers

The home screen displays up to eight lines with a maximum of 21 characters per line. If an expression or series of instructions exceeds 21 characters and spaces, it automatically continues on the next line.

After all eight lines are full, text scrolls off the top of the display. You can press **[↑]** to scroll up the home screen, only as far as the first character in the current entry. To retrieve, edit, and re-execute previous entries, use **[2nd]** **[ENTRY]** (page 28).

*You need not clear the home screen to begin a new entry.*

The mode settings control the way the TI-86 interprets expressions and displays answers (page 34).

When an entry is executed on the home screen, the answer is displayed on the right side of the next line. When you execute an instruction, **Done** is typically displayed on the right side of the next line.

Entry → 109 2  
 Answer → .301029995664

If an answer is too long to display on the screen, an ellipsis (...) is displayed, initially to the right. To view more of the answer, press  $\blacktriangleright$ . When you do, an ellipsis is displayed to the left. To scroll back, press  $\blacktriangleleft$ .

Entry → 2 seq(x,x,1,20)  
 Answer → (2 4 6 8 10 12 14 16...)

The TI-86 on-screen division symbol is a forward slash (/), as in a fraction.

A symbol or abbreviation of each key's primary function is printed in white on the key. For example, when you press  $\boxed{+}$ , a plus sign is pasted to the cursor location. This guidebook describes number-entry keystrokes as **1**, **2**, **3**, and so on, instead of  $\boxed{1}$   $\boxed{2}$   $\boxed{3}$ .

## Entering Negative Numbers

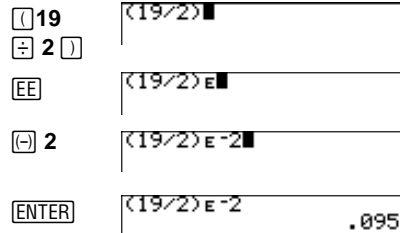
To enter a negative number, press  $\boxed{-}$  (the negate key), and then press the appropriate number keys. For example, to enter **-5**, press  $\boxed{-}$  **5**. Do not attempt to express a negative number using  $\boxed{-}$  (the subtract key).  $\boxed{-}$  and  $\boxed{-}$  are two different keys with different uses.

Always use parentheses to clarify negation when you use conversion instructions (Chapter 4).

The order in which the TI-86 evaluates negation and other functions within an expression is governed by the Equation Operating System™ (Appendix). For example, the result of **-4<sup>2</sup>** is **-16**, while the result of **(-4)<sup>2</sup>** is **16**. If you are unsure about the order of evaluation, use  $\boxed{[$  and  $\boxed{]}$  to clarify the intended use of the negation symbol.

## Using Scientific or Engineering Notation

- 1 Enter the mantissa (part of the number that precedes the exponent). This value can be an expression.
- 2 Paste **E** to the cursor location.
- 3 If the exponent is negative, paste **-** to the cursor location. Then enter a one-, two-, or three-digit exponent.
- 4 Evaluate the expression.



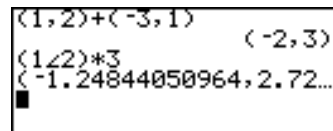
*In scientific notation only, one digit precedes the decimal.*

*In engineering notation, one, two, or three digits precede the decimal and the power of 10 exponent is a multiple of 3.*

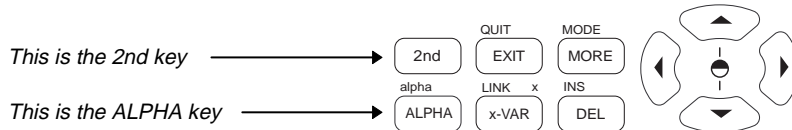
When you include scientific- or engineering-notation numbers in an expression, the TI-86 does not necessarily display answers in scientific or engineering notation. The mode settings (page 34) and the size of the number determine the notation of displayed answers.

## Entering Complex Numbers

On the TI-86, the complex number  $a+bi$  is entered as  $(a,b)$  in rectangular complex-number form or as  $(r\angle\theta)$  in polar complex-number form. For more information about complex numbers, read Chapter 4.



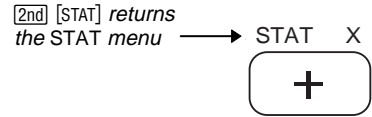
## Entering Other Characters





### The 2nd Key

The **[2nd]** key is yellow. When you press **[2nd]**, the cursor becomes **¶** (the 2nd cursor). When you press the next key, the yellow character, abbreviation, or word printed above that key is activated, instead of the key's primary function.

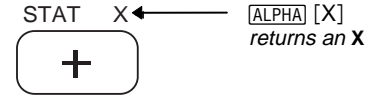


To enter a space within text, press **[ALPHA]** **[\_]**. Spaces are not valid within variable names.

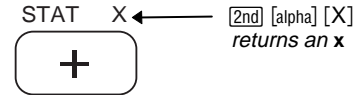
For convenience, you can press **[x-VAR]** instead of **[2nd]** **[alpha]** **[X]** to enter the commonly used **x** variable.

### The ALPHA Key

The **[ALPHA]** key is blue. When you press **[ALPHA]**, the cursor becomes **Ⓐ** (the uppercase ALPHA cursor). When you press the next key, the blue uppercase character printed above that key is pasted to the cursor location.



When you press **[2nd]** **[alpha]**, the cursor becomes **ⓐ** (the lowercase alpha cursor). When you press the next key, the lowercase version of the blue character is pasted to the cursor location.



### ALPHA-lock and alpha-lock

The **Name=** prompt and store symbol (→) set ALPHA-lock automatically.

To enter more than one uppercase or lowercase alpha character consecutively, set ALPHA-lock (for uppercase letters) or alpha-lock (for lowercase letters).

To set ALPHA-lock when the entry cursor is displayed, press **[ALPHA]** **[ALPHA]**.

- ◆ To cancel ALPHA-lock, press **[ALPHA]**.
- ◆ To switch from ALPHA-lock to alpha-lock, press **[2nd]** **[alpha]**.

To set alpha-lock when the entry cursor is displayed, press **[2nd]** **[alpha]** **[ALPHA]**.

- ◆ To cancel alpha-lock, press **[ALPHA]** **[ALPHA]**.

- ◆ To switch from alpha-lock to ALPHA-lock, press **[ALPHA]**.

You can use **[2nd]** when ALPHA-lock or alpha-lock is on. Also, if you press a key that has no blue character above it, such as **[GRAPH]**, **[DEL]**, or **[◀]**, the key's primary function still applies.

### Common Cursors

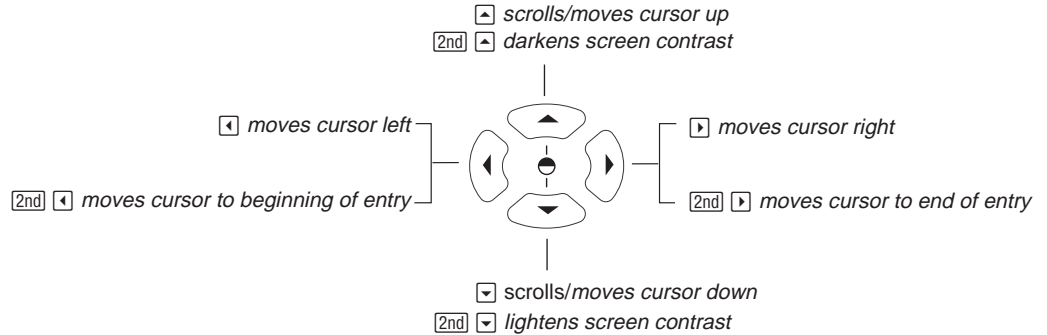
*In most cases, the appearance of the cursor indicates what will happen when you press the next key.*

*Graphs and editors sometimes use additional cursors, which are described in other chapters.*

Entry	■	Enters a character at the cursor, overwriting any existing character
Insert	—	Inserts a character at the cursor location and shifts remaining characters right
Second	Ⓛ	Enters a 2nd character or executes a 2nd operation (yellow on the keyboard)
ALPHA	Ⓐ	Enters an uppercase ALPHA character (blue on the keyboard)
alpha	ⓐ	Enters the lowercase version of an ALPHA character (blue on the keyboard)
Full	▣	Accepts no data; maximum characters are entered at a prompt or memory is full

- ◆ If you press **[ALPHA]** after **[2nd]** **[INS]**, the cursor becomes an underlined **A** (**A**).
- ◆ If you press **[2nd]** **[ALPHA]** after **[2nd]** **[INS]**, the cursor becomes an underlined **a** (**a**).
- ◆ If you press **[2nd]** after **[2nd]** **[INS]**, the insert cursor becomes an underlined **↑** (**↑**).

## Cursor Direction Keys



If you hold down ▶, ▼, ◀, or ▲, the cursor continues to move.

## Inserting, Deleting, and Clearing Characters

The entry cursor (■) overwrites characters.

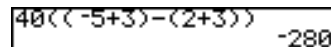
- 2nd [INS] Changes the cursor to the insert cursor ( \_ ); inserts characters at the insert cursor and shifts remaining characters right; to cancel insert, press 2nd [INS] or press ▶, ▼, ◀, or ▲
- [DEL] Deletes a character at the cursor; to continue deleting to the right, hold down [DEL]
- [CLEAR] Clears the current entry on the home screen; [CLEAR] [CLEAR] clears the entire home screen

## Entering Expressions and Instructions

### Entering an Expression

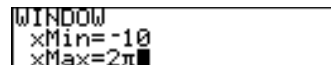
An expression is any combination of numbers and variables that serve as arguments for one or more functions. On the TI-86, you typically enter an expression in the same order as you would write it on paper. For example,  $\pi r^2$ ,  $5 \tan x$ Stat, and  $40((-5+3)-(2+3))$  are expressions.

You can use an expression on the home screen to calculate an answer.



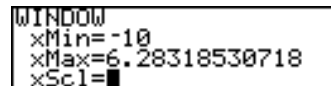
40((-5+3)-(2+3))  
-280

In most places where a value is required, you can use an expression to enter the value.



WINDOW  
xMin=-10  
xMax=2π

For example, enter an expression as a window variable value (Chapter 5). When you press  $\downarrow$ ,  $\uparrow$ ,  $\text{ENTER}$ , or  $\text{EXIT}$ , the TI-86 evaluates the expression and replaces it with the result.

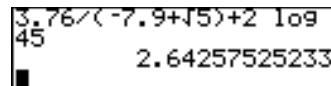


WINDOW  
xMin=-10  
xMax=6.28318530718  
xScl=

To enter an expression, you enter numbers, variables, and functions from the keyboard and menus (page 31). When you press  $\text{ENTER}$ , the expression is evaluated (regardless of the cursor location) according to EOS order-of-evaluation rules (Appendix), and the answer is displayed.

To enter the expression  $3.76 \div (-7.9 + \sqrt{5}) + 2 \log 45$  and then evaluate it, you would press these keys:

3  $\square$  76  $\square$   $\square$   $\square$  7  $\square$  9  $\square$  2nd  $\square$   $\square$  5  $\square$  +  $\square$  2  $\square$  LOG  $\square$  45  $\square$  ENTER



3.76/(-7.9+sqrt(5))+2 log 45  
2.64257525233

*In this guidebook, optional arguments are shown in brackets ( [ and ] ). Do not include these brackets when you enter the arguments.*

## Using Functions in Expressions

A function returns a value. Some examples of functions are  $\div$ ,  $-$ ,  $+$ ,  $\sqrt{\quad}$ , and **log**. To use functions, you usually must enter one or more valid arguments.

When this guidebook describes the syntax of a function or instruction, each argument is in italics. For example: **sin** *angle*. Press **[SIN]** to enter **sin**, and then enter a valid *angle* measurement (or an expression that resolves to *angle*). For functions or instructions with more than one argument, you must separate each argument from the other with a comma.

Some functions require the arguments to be in parentheses. When you are unsure of the evaluation order, use parentheses to clarify a function's place within an expression.

## Using an Instruction

*The A to Z Reference describes all TI-86 functions and instructions, including their required and optional arguments.*

An instruction initiates an action. For example, **CIDrw** is an instruction that, when executed, clears all drawn elements from a graph. You cannot use an instruction in an expression. Generally, the first letter of each instruction name is uppercase on the TI-86. Some instructions take more than one argument, as indicated by an open parenthesis ( ( ) at the end of the name. For example, **Circl**( requires three arguments, **Circl**(*x,y,radius*).

## Entering Functions, Instructions, and Operators

*In the CATALOG, to move to the first item beginning with a letter, press that letter (as in [L] in the example).*

You can enter a function, instruction, or operator in any of three ways (**log 45**, for example).

- ◆ Paste it to the cursor location from the keyboard or a menu (**[LOG] 45**).
- ◆ Paste it to the cursor location from the CATALOG (**[2nd] [CATLG-VARS] [F1] [L] [F1] [F1] [ENTER] 45**).
- ◆ Enter it letter by letter (**[2nd] [alpha] [ALPHA] [L] [O] [G] [−] [ALPHA] [ALPHA] 45**).

As you can see in the example, using the built-in function or instruction typically is easier.

When you select a function, instruction, or operator, a symbol comprising one or more characters is pasted to the cursor location. Once the symbol is pasted to the cursor location, you can edit individual characters.

For example, assume that you pressed  $\boxed{2\text{nd}}$   $\boxed{\text{CATLG-VARS}}$   $\boxed{\text{MORE}}$   $\boxed{\text{MORE}}$   $\boxed{\text{F5}}$   $\boxed{\text{F1}}$   $\boxed{\text{F1}}$   $\boxed{\text{ENTER}}$  to paste **yMin** to the cursor location as part of an expression. Then you realized you wanted **xMin**. Instead of pressing nine keys to select **xMin**, you can simply press  $\boxed{\leftarrow}$   $\boxed{\leftarrow}$   $\boxed{\leftarrow}$   $\boxed{\leftarrow}$   $\boxed{\text{x-VAR}}$ .

### Entering Consecutive Entries

To enter two or more expressions or instructions consecutively, separate each from the next with a colon ( $\boxed{2\text{nd}}$   $\boxed{[:]}$ ). When you press  $\boxed{\text{ENTER}}$ , the TI-86 executes each entry from left to right and displays the result of the last expression or instruction. The entire group entry is stored in last entry (page 28).

(2π)→A:5A→B:A\*B  
197.392088022

*In the example, the → symbol indicates that the value before it is to be stored to the variable after it (Chapter 2). To paste → to the screen, press  $\boxed{\text{STO} \blacktriangleright}$ .*

### The Busy Indicator

When the TI-86 is calculating or graphing, a moving vertical line is displayed as the busy indicator in the top-right corner of the screen. When you pause a graph or a program, the busy indicator is replaced by the pause indicator, a moving vertical dotted line.

### Interrupting a Calculation or Graph

To interrupt a calculation or graph in progress, press  $\boxed{\text{ON}}$ . When you interrupt a calculation, the **ERROR 06 BREAK** message and menu are displayed.

- ◆ To return to the home screen, select **QUIT** (press  $\boxed{\text{F5}}$ ).
- ◆ To go to the beginning of the expression, select **GOTO** (press  $\boxed{\text{F1}}$ ). Press  $\boxed{\text{ENTER}}$  to recalculate the expression.

Chapter 5: Function Graphing introduces graphing.

If a syntax error occurs within a stored equation during program execution, select **GOTO** to return to the equation editor, not to the program (Chapter 5).

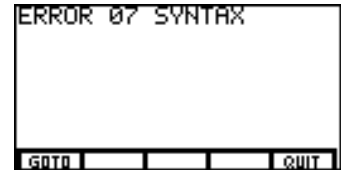
When you interrupt a graph, a partial graph and the GRAPH menu are displayed.

- ◆ To return to the home screen, press **[CLEAR]** **[CLEAR]** or any non-graphing key.
- ◆ To restart graphing, select an instruction that displays the graph.

## Diagnosing an Error

When the TI-86 detects an error, it returns an error message, such as **ERROR 04 DOMAIN** or **ERROR 07 SYNTAX**. The Appendix describes each error type and possible reasons for the error.

- ◆ If you select **QUIT** (or press **[2nd]** **[QUIT]** or **[CLEAR]** **[CLEAR]**), the home screen is displayed.
- ◆ If you select **GOTO**, the previous screen is displayed with the cursor on or near the error.



## Correcting an Error

- 1 Note the error type (**ERROR ## errorType**).
- 2 Select **GOTO**, if available. The previous screen is displayed with the cursor on or near the error.
- 3 Determine the cause for the error. If you cannot, refer to the Appendix for possible causes.
- 4 Correct the error and continue.

## Reusing Previous Entries and the Last Answer

### Retrieving the Last Entry

When you press  $\boxed{\text{ENTER}}$  on the home screen to evaluate an expression or to execute an instruction, the entire expression or instruction is placed in a storage area called ENTRY (last entry). When you turn off the TI-86, ENTRY is retained in memory.

To retrieve the last entry, press  $\boxed{2\text{nd}} \boxed{\text{ENTRY}}$ . The current line is cleared and the entry is pasted to the line.

### Retrieving and Editing the Last Entry

- ① On the home screen, retrieve the previous entry.  $\boxed{2\text{nd}} \boxed{\text{ENTRY}}$
- ② Edit the retrieved entry.  $\boxed{\leftarrow} \boxed{\leftarrow} \boxed{\leftarrow} \boxed{\leftarrow} \boxed{\leftarrow} \boxed{32}$
- ③ Re-execute the edited entry.  $\boxed{\text{ENTER}}$

### Retrieving Previous Entries

The TI-86 retains as many previous entries as possible in ENTRY, up to a capacity of 128 bytes. To scroll from the newest to the older previous entries stored to ENTRY, repeat  $\boxed{2\text{nd}} \boxed{\text{ENTRY}}$ . If you press  $\boxed{2\text{nd}} \boxed{\text{ENTRY}}$  after displaying the oldest stored entry, the newest stored entry is displayed again; continuing to press  $\boxed{2\text{nd}} \boxed{\text{ENTRY}}$  repeats the order.



Consecutively entered entries separated by colons (page 26) are stored as one entry.

The formula for finding the area of a circle is  $A=\pi r^2$ .

The equation solver (Chapter 15) is another tool with which you can perform this task.

## Retrieving Multiple Entries

To store two or more expressions or instructions together to ENTRY, enter them on one line, separating each from the other with a colon, and then press **ENTER**. Upon execution, the entire group is stored in ENTRY. The example below shows one of many ways you can manipulate this feature to avoid tedious manual re-entry.

- Use trial and error to find the radius of a circle with an area of 200 square centimeters. Store **8** to **r** as your first guess, then execute  $\pi r^2$ .
 

$8$  **[STO]**  $\rightarrow$  **[2nd]** **[alpha]** **[R]**  
 $\rightarrow$  **[2nd]** **[:]** **[2nd]** **[ $\pi$ ]** **[R]** **[ALPHA]**  
 $\rightarrow$  **[ALPHA]** **[x<sup>2</sup>]** **[ENTER]**
- Retrieve  $8\rightarrow r:\pi r^2$  and insert **7.958** as a new guess. Continue guessing to approach the answer of **200**.
 

$\rightarrow$  **[2nd]** **[ENTRY]**  
 $\rightarrow$  **[2nd]** **[ $\leftarrow$ ]** **7** **[2nd]** **[INS]** **[.]** **958**  
 $\rightarrow$  **[ENTER]**

```
8→r:πr2
201.06192983
```

```
8→r:πr2
7.958→r:πr2
201.06192983
198.956321336
```

## Clearing the ENTRY Storage Area

To clear all data from the ENTRY storage area, begin on a blank line on the home screen, select **ClrEnt** from the MEM menu (press **[2nd]** **[MEM]** **[F5]**), and then press **ENTER**.

## Retrieving the Last Answer

When an expression is evaluated successfully on the home screen or in a program, the TI-86 stores the answer to a built-in variable called **Ans** (last answer). **Ans** may be a real or complex number, list, vector, matrix, or string. When you turn off the TI-86, the value in **Ans** is retained in memory.

To copy the variable name **Ans** to the cursor location, press  $\boxed{2\text{nd}} \boxed{[\text{ANS}]}$ . You can use the variable **Ans** anywhere that the value stored to it is valid. When the expression is evaluated, the TI-86 calculates the result using the value stored in **Ans**.

- Calculate the area of a garden plot **1.7** meters by **4.2** meters.  $1 \boxed{.}$   $7 \boxed{\times}$   $4 \boxed{.}$   $2 \boxed{}$   
 $\boxed{\text{ENTER}}$
- Calculate the yield per square meter if the plot produces a total of **147** tomatoes.  $147 \boxed{\div}$   $\boxed{2\text{nd}} \boxed{[\text{ANS}]}$   
 $\boxed{\text{ENTER}}$

1.7*4.2	7.14
147/Ans	20.5882352941

### Using Ans Preceding a Function

Previous answers are stored to **Ans**. If you begin an expression by entering a function that requires a preceding argument, the TI-86 automatically enters **Ans** as the argument.

- Enter and execute an expression.  $5 \boxed{\div}$   $2 \boxed{\text{ENTER}}$
- Enter a function without an argument. **Ans** is pasted to the screen, followed by the function.  $\boxed{\times}$   $9 \boxed{.}$   $9 \boxed{\text{ENTER}}$

5/2	2.5
Ans*9.9	24.75

### Storing Results to a Variable

- Calculate the area of a circle with radius **5** meters.  $\boxed{2\text{nd}} \boxed{[\pi]} 5 \boxed{x^2}$   
 $\boxed{\text{ENTER}}$
- Calculate the volume of a cylinder of radius **5** meters and height **3.3** meters.  $\boxed{\times}$   $3 \boxed{.}$   $3 \boxed{\text{ENTER}}$
- Store the result to the variable **V**.  $\boxed{\text{STO}\rightarrow} \boxed{[V]}$   
 $\boxed{\text{ENTER}}$

$\pi 5^2$	78.5398163397
Ans*3.3	259.181393921
Ans→V	259.181393921

## Using TI-86 Menus

The symbols for many TI-86 features are found in menus instead of on the TI-86 keyboard.

### Displaying a Menu

The way to display a particular menu depends on the menu's location on the TI-86.

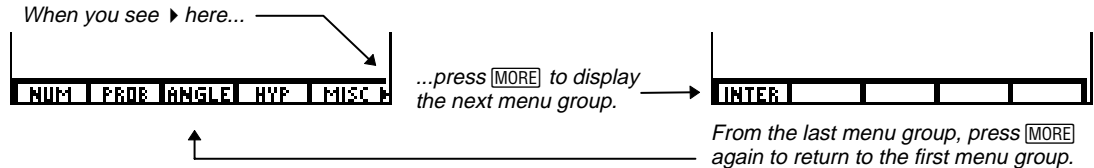
Menu-Displaying Method	Example
Press a key that has a menu name on it	[GRAPH] displays the GRAPH menu
Press [2nd] and then a 2nd-key menu name	[2nd] [MATH] displays the MATH menu
Select a menu name from another menu	[2nd] [MATH] [F1] displays the MATH NUM menu
Select an editor or selection screen	[2nd] [LIST] [F4] displays the list editor menu with the list editor
Accidentally commit an error	1 [STO▶] [ENTER] displays the error menu

*Some TI-86 menus have as many as 25 items.*

When you display a menu, a menu group of one to five items is displayed on the bottom of the screen. If the more symbol (▶) is displayed after the fifth item in a menu group, the menu continues for at least one more menu group. To view the next menu group, press [MORE]. The last menu group of one to five items does not have a ▶ symbol.

For example, press [2nd] [MATH] to display the MATH menu.

[▶], [◀], [↵], and [⏏] do not work on menus.



## The Menu Keys

**2nd** upper menu keys → M1 M2 M3 M4 M5

lower menu keys → **F1** **F2** **F3** **F4** **F5**

**2nd** **[QUIT]** clears all menus → **QUIT**

**2nd** **[M1]** through **[M5]** selects → **2nd** **EXIT** **[MORE]**  
upper menu items

**[MORE]** scrolls lower menu groups

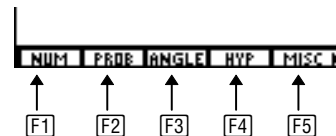
**EXIT** removes the lower menu

## Selecting a Menu Item

The Appendix Menu Map shows every TI-86 menu.

Typically, a TI-86 menu item is five characters long or less.

When you display a menu, one to five items are displayed. To select a menu item, press the menu selection key directly below the item. For example, in the MATH menu to the right, press **F1** to select **NUM**, press **F2** to select **PROB**, and so on.



When you select a menu item that displays another menu, the first menu moves up one line on the screen to make room for the new menu. All items on the original menu are displayed in reverse type, except the item you selected.

**[MORE]** only scrolls the lower menu; it does not scroll the upper menu.



...the MATH menu moves up and the MATH NUM menu is displayed.



To remove the MATH NUM menu and move the MATH menu down, press **EXIT**.

To select an item from the upper menu, press  $\boxed{2\text{nd}}$  and the appropriate key  $\boxed{[M1]}$  through  $\boxed{[M5]}$ .

To select **PROB** from the upper menu, press  $\boxed{2\text{nd}}$   $\boxed{[M2]}$ .

To select **iPart** from the lower menu, press  $\boxed{[F2]}$ .



When an editor menu is displayed as the upper menu, and you select an item from the lower menu that displays yet another menu, the editor menu remains as the upper menu.

When you select **NUM** from the lower menu...



Upper: equation editor menu  
Lower: MATH menu

...the equation editor menu remains and the MATH NUM menu is displayed.

The MATH menu disappears.



Upper: equation editor menu  
Lower: MATH NUM menu

## Exiting (Removing) a Menu

To remove a menu from the bottom of a graph screen, press  $\boxed{\text{CLEAR}}$  after plotting the graph (Chapter 5).

To remove the lower menu from the screen, press  $\boxed{\text{EXIT}}$ .

When you press  $\boxed{\text{EXIT}}$ ...



Upper: MATH menu  
Lower: MATH NUM menu

...the MATH NUM menu disappears and the MATH menu moves down.



Lower: MATH menu

Press  $\boxed{\text{EXIT}}$  again, and the MATH menu disappears.



No menu

In the screen to the right, the default mode settings are highlighted along the left side of the screen.

## Viewing and Changing Modes

To display the mode settings, press  $\boxed{2\text{nd}}$   $\boxed{[\text{MODE}]}$ . The current settings are highlighted. Mode settings control how the TI-86 displays and interprets numbers and graphs. The Constant Memory feature retains current mode settings when the TI-86 is turned off. All numbers, including elements of matrices and lists, are displayed according to the mode settings.

```
Normal Sci Eng
Float 012345678901
Radian Degree
RectO PolarC
Func Pol Param DifEq
Dec Bin Oct Hex
RectU CylU SphereU
dxDer1 dxNDer
```

This example changes the decimal mode setting to 2, as in U.S. dollars and cents.

### Changing a Mode Setting

- 1 Move the cursor to the line of the setting that you want to change (decimal setting in the example).
- 2 Move the cursor to the setting you want (2 decimal places).
- 3 Execute the change.



```
Normal Sci Eng
Float 012345678901
Radian Degree
RectO PolarC
Func Pol Param DifEq
Dec Bin Oct Hex
RectU CylU SphereU
dxDer1 dxNDer
```

In **Normal** notation, if the answer is more than 12 digits or the absolute value of the answer  $< .001$ , it is displayed in scientific notation.

Notation modes do not affect how you enter numbers.

### Notation Modes

- Normal** Displays results with digits to the left and right of the decimal (as in **123456.7**)
- Sci** (scientific) Displays results in two parts: significant digits (with one digit to the left of the decimal) are displayed to the left of **E** and the appropriate power of 10 is displayed to the right of **E** (as in **1.234567E5**)
- Eng** (engineering) Displays results in two parts: significant digits (with one, two, or three digits to the left of the decimal) are displayed to the left of **E** and the appropriate power of 10 (which is always a multiple of 3) is displayed to the right of **E** (as in **123.4567E3**)

### Decimal Modes

- Float** (floating) Displays results up to 12 digits, plus any sign and the floating decimal point
- (fixed) (**012345678901**; each number is a setting) Displays results with the specified number of digits to the right of the decimal point (rounds answers to the specified decimal place); the second **0** sets 10; the second **1** sets 11

### Angle Modes

- Radian** Interprets angle values as radians; displays answers in radians
- Degree** Interprets angle values as degrees; displays answers in degrees

### Complex Number Modes

- RectC** (rectangular complex mode) Displays complex-number results as *(real,imaginary)*
- PolarC** (polar complex mode) Displays complex-number results as *(magnitude∠angle)*

### Graphing Modes

- Func** (function graphing) Plots functions where **y** is a function of **x**
- Pol** (polar graphing) Plots functions where **r** is a function of  $\theta$
- Param** (parametric graphing) Plots relations where **x** and **y** are functions of **t**
- DifEq** (differential equation graphing) Plots differential equations in terms of **t**

### Number Base Modes

- Dec** (decimal number base) Interprets and displays numbers as decimal (base 10)
- Bin** (binary number base) Interprets numbers as binary (base 2); displays **b** suffix with answers
- Oct** (octal number base) Interprets numbers as octal (base 8); displays **o** suffix with answers
- Hex** (hexadecimal number base) Interprets numbers as hexadecimal (base 16); displays **h** suffix with answers

*Non-decimal modes are valid only on the home screen or in the program editor.*

*Vector modes do not affect how you enter vectors.*

### Vector Coordinate Modes

- RectV** (rectangular vector coordinates) Displays answers in the form  $[x\ y]$  for two-element vectors and  $[x\ y\ z]$  for three-element vectors
- CylV** (cylindrical vector coordinates) Displays results in the form  $[r\ \angle\ \theta]$  for two-element vectors and  $[r\ \angle\ \theta\ z]$  for three-element vectors
- SphereV** (spherical vector coordinates) Displays results in the form  $[r\ \angle\ \theta]$  for two-element vectors and  $[r\ \angle\ \theta\ \phi]$  for three-element vectors

### Differentiation Modes

- dxDer1** (exact differentiation) Uses **der1** (Chapter 3) to differentiate exactly and calculate the value for each function in an expression (**dxDer1** is more accurate than **dxNDer**, but it restricts the kinds of functions that are valid in the expression)
- dxNDer** (numeric differentiation) Uses **nDer** to differentiate numerically and calculate the value for an expression (**dxNDer** is less accurate than **dxDer1**, but more kinds of functions are valid in the expression)

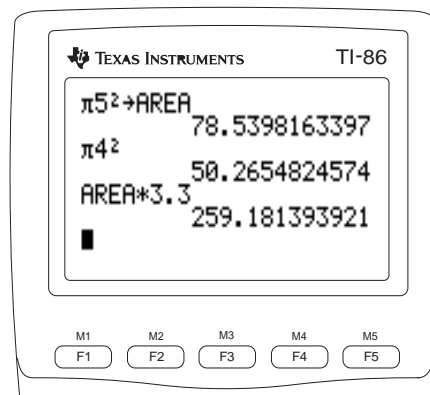
*The value stored to  $\delta$  affects **dxNDer** (Appendix).*



# 2

## The CATALOG, Variables, and Characters

The CATALOG .....	38
Storing Data to Variables.....	39
Classifying Variables as Data Types.....	42
The CUSTOM Menu .....	44
The CHAR (Character) Menu.....	45

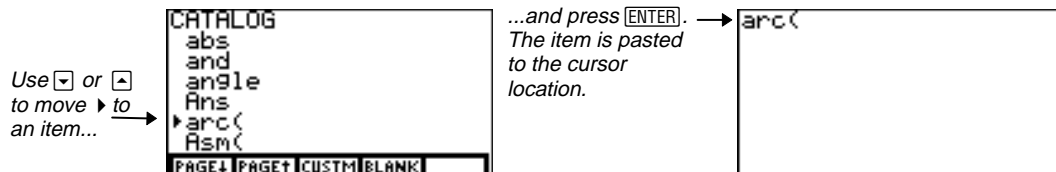


The CATALOG is the first item on the CATLG-VARS menu.

## The CATALOG [2nd] [CATLG-VARS] [F1]

The CATALOG displays all TI-86 functions and instructions in alphabetical order. Items that do not begin with a letter (such as **+** or **Bin**) are at the end of the CATALOG.

The selection cursor (**▶**) indicates the current item. To select an item from the CATALOG, move the selection cursor to the item and press [ENTER]. The CATALOG disappears and the name is pasted to the previous cursor location.



### To jump...

To the first item beginning with a particular letter

To special characters at the end of the CATALOG

Down one whole screen

Up one whole screen

### Do this:

Press the letter; ALPHA-lock is on

Press [down] from the first CATALOG item

Select **PAGE↓** from the CATALOG menu ([F1])

Select **PAGE↑** from the CATALOG menu ([F2])

The menu items **CUSTM** and **BLANK** are on the CATALOG menu and each VARS screen menu. With them, you can create and edit your own CUSTOM menu of up to 15 CATALOG items and variables, including program names. For details about the CUSTOM menu, read page 44.

*This chapter describes the first two data storage methods listed here. The other methods are described in the appropriate chapters.*

## Storing Data to Variables

On the TI-86, data can be stored to variables in several ways. You can:

- ◆ Use  $\boxed{\text{STO}}\blacktriangleright$  to store a value to a variable.
- ◆ Use  $=$  to store an unevaluated expression to an equation variable.
- ◆ Use an editor's **Name=** prompt to store several types of data to a variable.
- ◆ Change TI-86 settings or reset defaults and memory to the factory settings.
- ◆ Execute functions that cause the TI-86 to store data automatically to built-in variables.

The TI-86 has built-in variable names with specific purposes, such as equation variables, list names, statistical result variables, window variables, and **Ans**. You can store values to some of them. They are introduced in the appropriate chapters of this guidebook.

### Creating a Variable Name

You can create your own variable name when you use  $\boxed{\text{STO}}\blacktriangleright$ ,  $=$ , or a **Name=** prompt to store data. When you create a user-created variable name, follow these guidelines.

- ◆ The user-created variable name can be from one to eight characters long.
- ◆ The first character must be a letter, which includes all CHAR GREEK menu items, as well as  $\tilde{N}$ ,  $\tilde{n}$ ,  $\zeta$ , and  $\zeta$  from the CHAR MISC menu.
- ◆ A user-created variable name cannot replicate a TI-86 feature symbol or built-in variable. For example, you cannot create **abs**, because **abs** is the absolute value function symbol. You cannot create **Ans**, because it is already a built-in variable name.
- ◆ The TI-86 distinguishes between uppercase and lowercase characters in variable names. For example, **ANS**, **Ans**, and **ans** are three different variable names. Therefore, only **Ans** is a built-in variable name; **ANS** and **ans** can be user-created variable names.

### Storing a Value to a Variable Name

- |   |  |                               |   |
|---|--|-------------------------------|---|
| ① | Enter a value, which can be an expression.   | [2nd] [π] 5 [x <sup>2</sup> ] | <div style="border: 1px solid black; padding: 2px; width: fit-content;">π5<sup>2</sup></div>                        |
| ② | Enter → (the store symbol) next to the value.  | [STO→]                        | <div style="border: 1px solid black; padding: 2px; width: fit-content;">π5<sup>2</sup>→</div>                       |
| ③ | Create a variable name one to eight characters long, starting with a letter. ALPHA-lock is on. | [A] [R] [E] [A]               | <div style="border: 1px solid black; padding: 2px; width: fit-content;">π5<sup>2</sup>→AREA</div>                   |
| ④ | Store the value to the variable. The value stored to the variable is displayed as a result.    | [ENTER]                       | <div style="border: 1px solid black; padding: 2px; width: fit-content;">π5<sup>2</sup>→AREA<br/>78.5398163397</div> |

### Storing an Unevaluated Expression

When you store an expression to memory using [STO→] (with the → sign), the expression is evaluated and the result is stored to a variable.

When you store an unevaluated expression using [ALPHA] [=], or the equation editor (Chapter 5), or the equation solver (Chapter 15), the unevaluated expression is stored to an equation variable.

To store an unevaluated expression on the home screen or in a program, the syntax is:

*variable=expression*

where *variable* always precedes the equals sign and *expression* always follows the equals sign.

You can use = to store a mathematical expression to an equation variable. For example, **F=M\*A**.

*When you use =, variable is first, then =, then expression. In contrast, when you use →, value is first, then →, then variable.*

In the example, the TI-86 multiplies the value stored to **AREA** times 3.3.

To paste **AREA** to the cursor location, you can press  $\boxed{2\text{nd}}$   $\boxed{[\text{CATLG-VARS}]}$   $\boxed{F3}$ , move the selection cursor ( $\blacktriangleright$ ) to **AREA**, and press  $\boxed{\text{ENTER}}$ .

To paste  $\blacktriangleright$  to the cursor location, press  $\boxed{\text{STO}\blacktriangleright}$ .

To paste a variable name, you can select it from a VARS menu (page 42).

## Storing an Answer

To store an answer to a variable before you evaluate another expression, use  $\boxed{\text{STO}\blacktriangleright}$  and **Ans**.

- 1 Enter and evaluate an expression.

$\boxed{\text{ALPHA}}$   $\boxed{\text{ALPHA}}$   
 $\boxed{[A]}$   $\boxed{[R]}$   $\boxed{[E]}$   $\boxed{[A]}$   $\boxed{\text{ALPHA}}$   
 $\boxed{\times}$   $\boxed{3}$   $\boxed{.}$   $\boxed{3}$   $\boxed{\text{ENTER}}$

```
AREA*3.3
      259.181393921
```

- 2 Store the answer to a user-created variable or to a valid built-in variable.  
The value stored to the variable is displayed as a result.

$\boxed{\text{STO}\blacktriangleright}$   $\boxed{[V]}$   $\boxed{[O]}$   $\boxed{[L]}$   $\boxed{\text{ENTER}}$

```
AREA*3.3
      259.181393921
Ans→VOL
      259.181393921
```

## Copying a Variable Value

To copy the contents of *variableA* into *variableB*, the syntax is:  
*variableA* $\blacktriangleright$ *variableB*

For example, **RegEq** $\blacktriangleright$ **y1** stores the regression equation (Chapter 14) to the variable **y1**.

## Displaying a Variable Value

- 1 With the cursor on a blank line on the home screen, paste the variable name to the cursor location, as described above.
- 2 Display the contents of the variable.

$\boxed{2\text{nd}}$   $\boxed{[\text{CATLG-VARS}]}$   $\boxed{F3}$   
 $\boxed{\blacktriangledown}$  (location may vary)  $\boxed{\text{ENTER}}$   
 $\boxed{\text{ENTER}}$

```
VOL
      259.181393921
```

You also can display variables containing some data types by displaying them in the appropriate editor (such as the list editor or window variable editor) or graph. These methods are detailed in subsequent chapters of this guidebook.

### Recalling a Variable Value

- 1 Move the cursor to where you want to insert the recalled variable value. **100**
- 2 Display the **Rcl** prompt at the bottom of the screen. ALPHA-lock is on. **2nd**  [RCL]
- 3 Enter the variable name you want to recall. **[V]**  [O] [L]
- 4 Recall the variable contents to the cursor location. The **Rcl** prompt disappears and the entry cursor returns. **ENTER**

```
100*
```

```
Rcl 
```

```
Rcl VOL
```

```
100*259.181393921
```

To cancel RCL, press  [CLEAR].

Editing a recalled value does not change the value stored to the variable.

### Classifying Variables as Data Types

The TI-86 classifies variables according to data type and places each variable on a data-type selection screen. You can display each screen by selecting the appropriate data type from the CATLG-VARS menu, as described on page 43. Here are some examples.

If data...	The TI-86 classifies the data type as...	For example:
Begins with { and ends with }	A list (VARS LIST screen)	{1,2,3}
Begins with [ and ends with ]	A vector (VARS VECTR screen)	[1,2,3]
Begins with [[ and ends with ]]	A matrix (VARS MATRX screen)	[[1,2,3][4,5,6][7,8,9]]

When you store data in an editor, the TI-86 recognizes the data type according to the editor. For example, only vectors are stored using the vector editor.

To display additional menu groups, press **[MORE]**.

### The CATLG-VARS (CATALOG-Variables) Menu

**[2nd]** **[CATLG-VARS]**

<b>CATLG</b>	<b>ALL</b>	<b>REAL</b>	<b>CPLX</b>	<b>LIST</b>	▶	<b>VECTR</b>	<b>MATRX</b>	<b>STRNG</b>	<b>EQU</b>	<b>CONS</b>	
						▶	<b>PRGM</b>	<b>GDB</b>	<b>PIC</b>	<b>STAT</b>	<b>WIND</b>

The list names **fStat**, **xStat**, and **yStat** are statistical result variables on the VARS STAT screen.

<b>CATLG</b>	Displays the CATALOG
<b>ALL</b>	Displays a selection screen with all variables and names of all data types
<b>REAL</b>	Displays a selection screen with all real number variables
<b>CPLX</b>	Displays a selection screen with all complex number variables
<b>LIST</b>	Displays a selection screen with all list names
<b>VECTR</b>	Displays a selection screen with all vector names
<b>MATRX</b>	Displays a selection screen with all matrix names
<b>STRNG</b>	Displays a selection screen with all string variables
<b>EQU</b>	Displays a selection screen with all equation variables
<b>CONS</b>	Displays a selection screen with all user-defined constants
<b>PRGM</b>	Displays a selection screen with all program names
<b>GDB</b>	Displays a selection screen with all graph database names
<b>PIC</b>	Displays a selection screen with all picture names
<b>STAT</b>	Displays a selection screen with all statistical result variables
<b>WIND</b>	Displays a selection screen with all window variables

The example assumes that the real-number variables **AREA** and **VOL** from the example on page 41 have not been deleted from memory.

### Selecting a Variable Name

- 1 Select the appropriate data-type selection screen from the CATLG-VARS menu. 2nd [CATLG-VARS] [F3]
- 2 Move the cursor to the variable you want to select. ▼
- 3 Select the variable you want. ENTER



### The CUSTOM Menu 2nd [CATLG-VARS] [F1] [F3]

You can select up to 15 items from the CATALOG and VARS screens – program names, functions, instructions, and other items – to create your own CUSTOM menu. To display your CUSTOM menu, press CUSTOM. Use [F1] through [F5] and [MORE] to select items like any other menu.

### Entering CUSTOM Menu Items

- 1 Select **CUSTOM** from the CATALOG. The CUSTOM menu is displayed. ALPHA-lock is on. 2nd [CATLG-VARS] [F1] [F3]
- 2 Move the selection cursor (▶) to the item you want to copy to the CUSTOM menu. [C] ▼ ▼ ▼
- 3 Copy the item to the CUSTOM menu cell you select, replacing any previous item. [F3]
- 4 To enter more items, repeat steps 2 and 3 using different items and cells.
- 5 Display the CUSTOM menu. 2nd[QUIT] [CUSTOM]



When copying items into the CUSTOM menu, you can skip menu cells and menu groups.



To clear an item from the second or third menu group, press **[MORE]** until the item is displayed, and then select it.

### Clearing CUSTOM Menu Items

- 1 Select **BLANK** from the CATALOG menu. The CUSTOM BLANK menu is displayed. **[2nd] [CATLG-VARS] [F1] [F4]**
- 2 Clear the menu item. **[F3]**
- 3 To clear more items, repeat steps 2 and 3.



You cannot delete a TI-86 built-in variable.

You cannot delete a program variable using **DelVar** .

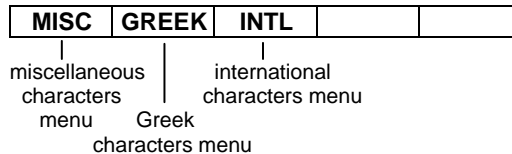
### Deleting a Variable from Memory

From the home screen or in a program, to delete from memory one user-created variable name (except a program name) and its contents, the syntax is:

**DelVar(variable)**

To delete user-created variable names and their contents (including program names), display the MEM DELET menu (**[2nd] [MEM] [F2]**), select the data type, select the variable, and then press **[ENTER]** (Chapter 16). Deleting a variable does not remove it from the CUSTOM menu (page 44).

### The CHAR (Character) Menu **[2nd] [CHAR]**



Ñ, ñ, Ç, and ç are valid as any character of a variable name, including the first letter.

%, ' , and ! can be functions.

All CHAR GREEK menu items are valid variable-name characters, including the first letter.  $\pi$  ( [2nd] [π] ) is not valid as a character;  $\pi$  is a constant on the TI-86.

### The CHAR MISC (Miscellaneous) Menu [2nd] [CHAR] [F1]

MISC	GREEK	INTL		
?	#	&	%	'

▶

!	@	\$	~	
---	---	----	---	--

▶

¿	Ñ	ñ	Ç	ç
---	---	---	---	---

### The CHAR GREEK Menu [2nd] [CHAR] [F2]

MISC	GREEK	INTL		
$\alpha$	$\beta$	$\gamma$	$\Delta$	$\delta$

▶

$\epsilon$	$\theta$	$\lambda$	$\mu$	$\rho$
------------	----------	-----------	-------	--------

▶

$\Sigma$	$\sigma$	$\tau$	$\phi$	$\Omega$
----------	----------	--------	--------	----------

### The CHAR INTL (International) Menu [2nd] [CHAR] [F3]

MISC	GREEK	INTL		
'	`	^	..	

You can combine modifiers on the CHAR INTL menu with uppercase or lowercase vowels to create vowels used in some languages. You can use these vowels in variable names and text.

### Adding a Modifier to a Vowel

- 1 Select the modifier from the CHAR INTL menu. ALPHA-lock is on. If necessary, switch to alpha-lock.
- 2 Enter the uppercase or lowercase vowel over which you want the modifier.

[2nd] [CHAR] [F3] [F4]  
[2nd] [alpha]

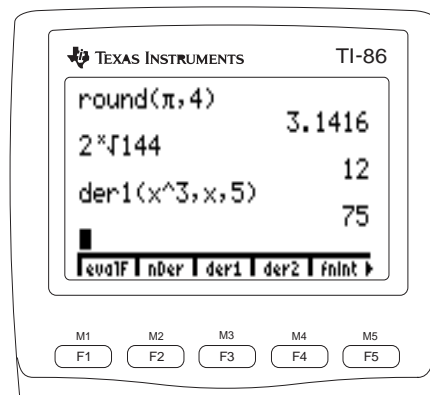
[O]



# 3

## Math, Calculus, and Test Operations

Keyboard Mathematical Functions .....	48
The MATH Menu.....	49
The CALC (Calculus) Menu .....	54
The TEST (Relational) Menu.....	55



The A to Z Reference details which data types are valid arguments for each function.

The most common mathematical functions are on the TI-86 keyboard. For syntax, details, and examples of these functions, refer to the A to Z Reference.

$x^{-1}$  (the multiplicative inverse) is equivalent to the reciprocal,  $1/x$ .

## Keyboard Mathematical Functions

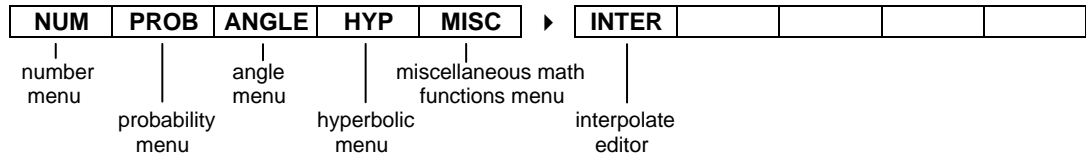
You can use these mathematical functions in expressions with real or complex values. You can use some of them with lists, vectors, matrices, or strings.

When you use lists, vectors, or matrices, the valid functions return a list of results calculated on an element-by-element basis. If you use two lists, vectors, or matrices in the same expression, they must be equal in dimension.

Key	Function	Key	Function
$\boxed{+}$	<b>+</b> (add)	$\boxed{\text{SIN}}$	<b>sin</b> (sine)
$\boxed{-}$	<b>-</b> (subtract)	$\boxed{\text{COS}}$	<b>cos</b> (cosine)
$\boxed{\times}$	<b>*</b> (multiply)	$\boxed{\text{TAN}}$	<b>tan</b> (tangent)
$\boxed{\div}$	<b>/</b> (divide)	$\boxed{2\text{nd}} \boxed{[\text{SIN}^{-1}]}$	<b>sin<sup>-1</sup></b> (arcsine; inverse of sine)
$\boxed{(-)}$	<b>-</b> (negate)	$\boxed{2\text{nd}} \boxed{[\text{COS}^{-1}]}$	<b>cos<sup>-1</sup></b> (arccosine; inverse of cosine)
$\boxed{x^2}$	<b>2</b> (square)	$\boxed{2\text{nd}} \boxed{[\text{TAN}^{-1}]}$	<b>tan<sup>-1</sup></b> (arctangent; inverse of tangent)
$\boxed{2\text{nd}} \boxed{[\sqrt{\quad}]}$	$\sqrt{\quad}$ (square root)	$\boxed{\text{LOG}}$	<b>log</b> (logarithm)
$\boxed{2\text{nd}} \boxed{[x^{-1}]}$	<b>-1</b> (inverse)	$\boxed{\text{LN}}$	<b>ln</b> (natural log)
$\boxed{\wedge}$	<b>^</b> (raise to a specified power)	$\boxed{2\text{nd}} \boxed{[e^x]}$	<b>e<sup>x</sup></b> (constant e raised to a power)
$\boxed{2\text{nd}} \boxed{[10^x]}$	<b>10^</b> (10 to a specified power)	$\boxed{2\text{nd}} \boxed{[\pi]}$	$\pi$ (constant pi; 3.1415926535898)
$\boxed{\text{EE}}$	<b>E</b> (exponent)		

## The MATH Menu

**2nd** **[MATH]**



## The MATH NUM (Number) Menu

**2nd** **[MATH]** **[F1]**



*value* can sometimes be an expression, list, vector, or matrix. For details about specific syntax options and examples, refer to the A to Z Reference.

**round**(*value*[, #ofDecimals])

Rounds *value* to 12 decimal places or to #ofDecimals

**iPart** *value*

Returns the integer part or parts of *value*

**fPart** *value*

Returns the fractional part or parts of *value*

**int** *value*

Returns the largest integer less than or equal to *value*

**abs** *value*

Returns the absolute value or magnitude of *value*

**sign** *value*

Returns **1** if *value* is positive; **0** if *value* is **0**; **-1** if *value* is negative

**min**(*valueA*, *valueB*)

Returns the smaller of *valueA* and *valueB*

**min**(*list*)

Returns the smallest element of *list*

**max**(*valueA*, *valueB*)

Returns the larger of *valueA* and *valueB*

**max**(*list*)

Returns the largest element of *list*

**mod**(*numberA*, *numberB*)

Returns *numberA* modulo *numberB*

## The MATH PROB (Probability) Menu

[2nd] [MATH] [F2]

NUM	PROB	ANGLE	HYP	MISC						
!	nPr	nCr	rand	randIn	▶	randN	randBi			

! (factorial) is valid for non-integers.

<i>value</i> !	Returns the factorial of a real <i>value</i>
<i>items</i> nPr <i>number</i>	Returns the number of permutations of <i>items</i> ( <b>n</b> ) taken <i>number</i> ( <b>r</b> ) at a time
<i>items</i> nCr <i>number</i>	Returns the number of combinations of <i>items</i> ( <b>n</b> ) taken <i>number</i> ( <b>r</b> ) at a time
rand	Returns a random number $> 0$ and $< 1$ ; to control a random number sequence, first store an integer seed value to <b>rand</b> (such as <b>0</b> → <b>rand</b> )
randInt( <i>lower</i> , <i>upper</i> [,#ofTrials])	(random integer) Returns a random integer bound by the specified integers, $lower \leq integer \leq upper$ ; to return a list of random integers, specify an integer $> 1$ for #ofTrials
randNorm( <i>mean</i> , <i>stdDeviation</i> [,#ofTrials])	(random normal) Returns a random real number from a normal distribution specified by <i>mean</i> and <i>stdDeviation</i> ; to return a list of random numbers, specify an integer $> 1$ for #ofTrials
randBin(#ofTrials, <i>probabilityOfSuccess</i> [,#ofSimulations])	(random binomial) Returns a random real number from a binomial distribution, where #ofTrials $\geq 1$ and $0 \leq probabilityOfSuccess \leq 1$ ; to return a list of random numbers, specify an integer $> 1$ for #ofSimulations

randInt, randNorm, and randBin are abbreviated in the MATH PROB menu.

### The MATH ANGLE Menu 2nd [MATH] F3

NUM	PROB	ANGLE	HYP	MISC
°	r	'	►DMS	

*angle* can be a list for °, r, and ►DMS.

In a calculation, the result of a *degrees'minutes'seconds'* entry is treated as degrees in **Degree** angle mode only. It is treated as radians in **Radian** angle mode.

*angle*°

Overrides current angle mode setting to express *angle* in degrees

*angle*r

Overrides current angle mode setting to express *angle* in radians

*degrees'minutes'seconds'*

Designates an angle as *degrees, minutes, and seconds*

*angle*►DMS

Displays *angle* in degrees°minutes'seconds" format, even though you use *degrees'minutes'seconds'* to enter a DMS angle

### The MATH HYP (Hyperbolic) Menu 2nd [MATH] F4

NUM	PROB	ANGLE	HYP	MISC
sinh	cosh	tanh	sinh <sup>-1</sup>	cosh <sup>-1</sup>
			► tanh <sup>-1</sup>	

*value* can sometimes be an expression, list, vector, or matrix. For details about specific syntax options and examples, refer to the A to Z Reference.

**sinh** *value*

Returns the hyperbolic sine of *value*

**cosh** *value*

Returns the hyperbolic cosine of *value*

**tanh** *value*

Returns the hyperbolic tangent of *value*

**sinh<sup>-1</sup>** *value*

Returns the hyperbolic arcsine of *value*

**cosh<sup>-1</sup>** *value*

Returns the hyperbolic arccosine of *value*

**tanh<sup>-1</sup>** *value*

Returns the hyperbolic arctangent of *value*

## The MATH MISC (Miscellaneous) Menu

[2nd] [MATH] [F5]

NUM	PROB	ANGLE	HYP	MISC						
sum	prod	seq	lcm	gcd	▶	►Frac	%	pEval	$x\sqrt{\quad}$	eval

*value* can sometimes be an expression, list, vector, or matrix. For details about specific syntax options, refer to the A to Z Reference.

**sum** *list*Returns the sum of the elements of *list***prod** *list*Returns the product of the elements of *list***seq**(*expression,variable,begin,end[,step]*)Returns a list in which each element is the value of *expression* evaluated for *variable* from *begin* to *end* by *step***lcm**(*valueA,valueB*)Returns the least common multiple of *valueA* and *valueB***gcd**(*valueA,valueB*)Returns the greatest common divisor of *valueA* and *valueB**value*►FracDisplays *value* as a fraction*value*%Returns *value* divided by 100 (multiplied by .01)*percent*%*number*Returns *percent* of *number***pEval**(*coefficientList,xValue*)Returns the value of a polynomial (whose coefficients are given in *coefficientList*) at *xValue* $x^{\text{th}}\text{root}\sqrt{x}\text{value}$ Returns the  $x^{\text{th}}\text{root}$  of *value***eval** *value*Returns a list of the values of all selected functions in the current graphing mode for the real *value* of the independent variable



To interpolate  $y$  from the home screen, select **inter** from the CATALOG, and then enter **inter**( $x1,y1,x2,y2,x$ ).

To interpolate  $x$  from the home screen, enter **inter**( $y1,x1,y2,x2,y$ ).

You can store individual values with the **STO►** key (Chapter 2).

### The Interpolate/Extrapolate Editor [2nd] [MATH] [MORE] [F1]

Using the interpolate/extrapolate editor, you can interpolate or extrapolate a value linearly, given two known pairs and the  $x$ -value or  $y$ -value of the unknown pair.

- 1 Display the interpolate/extrapolate editor. [2nd] [MATH] [MORE] [F1]
- 2 Enter real values for the first known pair ( $x1,y1$ ). The values can be expressions. 3 [ENTER] 5 [ENTER]
- 3 Enter values for the second known pair ( $x2,y2$ ). 4 [ENTER] 4 [ENTER]
- 4 Enter a value for either the  $x$  value or the  $y$  value of the unknown pair. 1 [ENTER]
- 5 If necessary, move the cursor to the value for which you want to solve ( $x$  or  $y$ ). ▲ or ▼
- 6 Select **SOLVE**. [F5]



The result is interpolated or extrapolated and displayed; the variables  $x$  and  $y$  are not changed. A solid square in the first column indicates the interpolated or extrapolated value.

After solving for a value, you can continue to use the interpolate/extrapolate editor.

You must set **Dec** mode to use the calculus functions.

For **evalF**, **nDer**, **der1**, and **der2**, *variable* can be a real or complex number or list.

You can use **der1** and **der2** in *expression*. You can use **nDer** once in *expression*.

For **fnInt**, **fMin**, and **fMax**, *lower* < *upper* must be true.

## The CALC (Calculus) Menu

**2nd** [CALC]

<b>evalF</b>	<b>nDer</b>	<b>der1</b>	<b>der2</b>	<b>fnInt</b>	▶	<b>fMin</b>	<b>fMax</b>	<b>arc</b>		
--------------	-------------	-------------	-------------	--------------	---	-------------	-------------	------------	--	--

The calculus functions return values with respect to any user-created variable, to built-in variables **eqn** and **exp**, and to graphing variables such as **x**, **t**, and **θ**.

<b>evalF</b> ( <i>expression,variable,value</i> )	Returns the value of <i>expression</i> with respect to <i>variable</i> for a given variable <i>value</i>
<b>nDer</b> ( <i>expression,variable</i> [, <i>value</i> ])	Returns an approximate numerical derivative of <i>expression</i> with respect to <i>variable</i> for the current variable value or specified variable <i>value</i>
<b>der1</b> ( <i>expression,variable</i> [, <i>value</i> ])	Returns the value of the first derivative of <i>expression</i> with respect to <i>variable</i> for the current variable value or specified variable <i>value</i>
<b>der2</b> ( <i>expression,variable</i> [, <i>value</i> ])	Returns the value of the second derivative of <i>expression</i> with respect to <i>variable</i> for the current variable value or specified variable <i>value</i>
<b>fnInt</b> ( <i>expression,variable,lower,upper</i> )	Returns the numerical integral of <i>expression</i> with respect to <i>variable</i> between <i>lower</i> and <i>upper</i> boundaries
<b>fMin</b> ( <i>expression,variable,lower,upper</i> )	Returns the minimum value of <i>expression</i> with respect to <i>variable</i> between <i>lower</i> and <i>upper</i> boundaries
<b>fMax</b> ( <i>expression,variable,lower,upper</i> )	Returns the maximum value of <i>expression</i> with respect to <i>variable</i> between <i>lower</i> and <i>upper</i> boundaries
<b>arc</b> ( <i>expression,variable,start,end</i> )	Returns the length of a segment of a curve defined by <i>expression</i> with respect to <i>variable</i> between <i>start</i> and <i>end</i>

The built-in variable  $\delta$  defines the step size in calculating **nDer**( in **dxNDer** differentiation mode only) and **arc**(. The built-in variable **tol** defines the tolerance in calculating **fnInt**(, **fMin**(, **fMax**(, and **arc**(. The value of each must be  $>0$ . These factors affect the accuracy of the calculations. As  $\delta$  becomes smaller, the approximation typically is more accurate. For example, **nDer(A^3,A,5)** returns **75.0001** if  $\delta=.01$ , but returns **75** if  $\delta=.0001$  (Appendix).

The function integral error value is stored to the variable **fnIntErr** (Appendix).

For **arc**( and **fnInt**( while **dxDer1** mode is set, these functions are not valid in *expression*: **evalF**(, **der1**(, **der2**(, **fMin**(, **fMax**(, **nDer**(, **seq**(, and any equation variable, such as **y1**.

You can approximate the fourth derivative at the current value of **x** with this formula: **nDer(nDer(der2(x^4,x),x),x)**.

## The TEST (Relational) Menu

2nd [TEST]

==	<	>	≤	≥	▶	≠				
----	---	---	---	---	---	---	--	--	--	--

*Relational functions are valid for two lists of the same length. When  $valueA$  and  $valueB$  are lists, a list of results calculated element by element is returned.*

- $valueA == valueB$  (equal to) Returns **1** if  $valueA$  is equal to  $valueB$ ; returns **0** if not equal;  $valueA$  and  $valueB$  can be real or complex numbers, lists, vectors, matrices, or strings
- $valueA < valueB$  (less than) Returns **1** if  $valueA$  is less than  $valueB$ ; returns **0** if  $valueA$  is not less than  $valueB$ ;  $valueA$  and  $valueB$  must be real numbers or lists
- $valueA > valueB$  (greater than) Returns **1** if  $valueA$  is greater than  $valueB$ ; returns **0** if  $valueA$  is not greater than  $valueB$ ;  $valueA$  and  $valueB$  must be real numbers or lists
- $valueA \leq valueB$  (less than or equal to) Returns **1** if  $valueA$  is less than or equal to  $valueB$ ; returns **0** if  $valueA$  is not less than or equal to  $valueB$ ;  $valueA$  and  $valueB$  must be real numbers or lists

$valueA \geq valueB$  (greater than or equal to) Returns **1** if  $valueA$  is greater than or equal to  $valueB$ ; returns **0** if  $valueA$  is not greater than or equal to  $valueB$ ;  $valueA$  and  $valueB$  must be real numbers or lists

$valueA \neq valueB$  (not equal to) Returns **1** if  $valueA$  is not equal to  $valueB$ ; returns **0** if  $valueA$  is equal to  $valueB$ ;  $valueA$  and  $valueB$  can be real or complex numbers, lists, vectors, matrices, or strings

### Using Tests in Expressions and Instructions

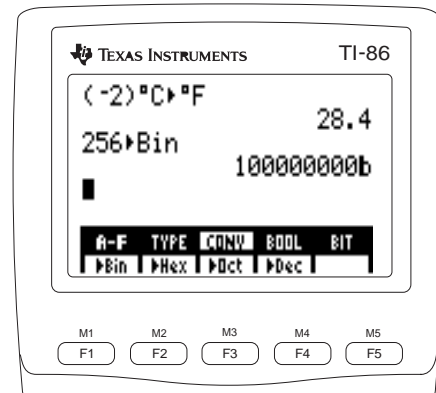
The TI-86 Evaluation Operating System (Appendix) performs all operations except Boolean operators before it performs relational functions. For example:

- ◆ The expression  $2+2==2+3$  evaluates to **0**. The TI-86 performs the addition first, and then compares 4 to 5.
- ◆ The expression  $2+(2==2)+3$  evaluates to **6**. The TI-86 performs the test in parentheses first, and then adds 2, 1, and 3.

*You can use relational functions to control program flow (Chapter 16).*

# 4 Constants, Conversions, Bases, and Complex Numbers

Using Built-In and User-Created Constants .....	58
Converting Units of Measure .....	61
Number Bases.....	65
Using Complex Numbers .....	70

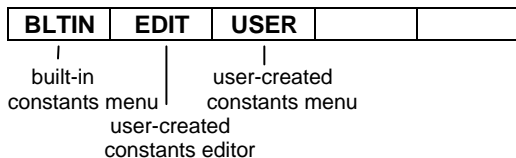


## Using Built-In and User-Created Constants

A constant is a variable with a specific value stored to it. The CONS BLTIN menu items are common constants built into the TI-86. You cannot edit the value of a built-in constant.

You can create your own constants and add them to the user-created constant menu for easy access. To enter a user-created constant, you must use the user-created constant editor (page 60); you cannot use  $\boxed{\text{STO}}\blacktriangleright$  or = to create a constant.

### The CONS (Constants) Menu $\boxed{2\text{nd}}\ [\text{CONS}]$



### The CONS BLTIN (Built-In Constants) Menu $\boxed{2\text{nd}}\ [\text{CONS}]\ [\text{F1}]$

<b>BLTIN</b>	<b>EDIT</b>	<b>USER</b>		
<b>Na</b>	<b>k</b>	<b>Cc</b>	<b>ec</b>	<b>Rc</b>

▶

<b>Gc</b>	<b>g</b>	<b>Me</b>	<b>Mp</b>	<b>Mn</b>
-----------	----------	-----------	-----------	-----------

▶

$\mu\mathbf{0}$	$\epsilon\mathbf{0}$	<b>h</b>	<b>c</b>	<b>u</b>
-----------------	----------------------	----------	----------	----------

*You can select built-in constants from the CONS BLTIN menu or enter them using the keyboard and the CHAR GREEK menu.*

Built-In Constant	Constant Name	Constant Value
<b>Na</b>	Avogadro's number	6.0221367E23 mole <sup>-1</sup>
<b>k</b>	Boltzman's constant	1.380658E-23 J/K
<b>Cc</b>	Coulomb constant	8.9875517873682E9 N m <sup>2</sup> /C <sup>2</sup>
<b>ec</b>	Electron charge	1.60217733E-19 C
<b>Rc</b>	Gas constant	8.31451 J/mole K
<b>Gc</b>	Gravitational constant	6.67259E-11 N m <sup>2</sup> /kg <sup>2</sup>
<b>g</b>	Earth acceleration due to gravity	9.80665 m/sec <sup>2</sup>
<b>Me</b>	Mass of an electron	9.1093897E-31 kg
<b>Mp</b>	Mass of a proton	1.6726231E-27 kg
<b>Mn</b>	Mass of a neutron	1.6749286E-27 kg
<b>μ0</b>	Permeability of a vacuum	1.2566370614359E-6 N/A <sup>2</sup>
<b>ε0</b>	Permittivity of a vacuum	8.8541878176204E-12 F/m
<b>h</b>	Planck's constant	6.6260755E-34 J sec
<b>c</b>	Speed of light in a vacuum	299,792,458 m/sec
<b>u</b>	Atomic mass unit	1.6605402E-27 kg
<b>π</b>	Pi	3.1415926535898
<b>e</b>	Base of natural log	2.718281828459

To use  $\pi$ , press  $\boxed{2nd} [\pi]$  or select it from the CATALOG.

To use  $e^\wedge$ , press  $\boxed{2nd} [e^x]$ .

To use  $e$ , press  $\boxed{2nd} [\alpha] [E]$ .

CONS USER menu items are the names of all stored user-created constants, arranged alphanumerically.

196.9665 is the atomic weight of gold (Au).

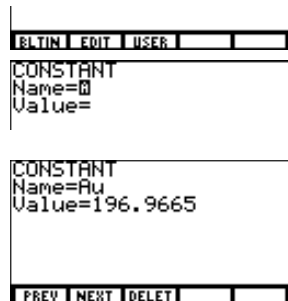
You can enter a value later.

If you select **PREV** when the first constant name is displayed, or **NEXT** when the last constant name is displayed, the CONS USER menu replaces the CONS EDIT menu.

You also can delete a constant from the MEM DELET CONS screen.

### Creating or Redefining a User-Created Constant

- 1 Display the CONS menu. [2nd] [CONS]
- 2 Display the constant editor. The **Name=** prompt, **Value=** prompt, and CONS USER menu are displayed. ALPHA-lock is on. [F2]
- 3 Enter a constant name. Either enter a new name one to eight characters long, starting with a letter, or select a name from the CONS USER menu. The cursor moves to the **Value=** prompt and the CONS EDIT menu is displayed (see below). [A] [2nd] [alpha]  
[U] [ENTER]
- 4 Enter the real or complex constant value, which can be an expression. The value is stored to the constant as you enter it. The user-created constant becomes a CONS USER menu item. 196 [.] 9665



### The Constant Editor Menu [2nd] [CONS] [F2] name [ENTER] or ▾

PREV	NEXT	DELET	
------	------	-------	--

- PREV** Displays the name and value (if any) of the previous constant on the CONS USER menu
- NEXT** Displays the name and value (if any) of the next constant on the CONS USER menu
- DELET** Deletes the name and value of the constant currently displayed in the constant editor



## Entering a Constant Name in an Expression

You can enter a constant in an expression in any of three ways.

- ◆ Select the constant name from the CONS BLTIN menu or the CONS USER menu.
- ◆ Select a user-created constant name from the VARS CONS screen.
- ◆ Use the ALPHA keys, alpha keys, and other character keys to enter a constant name.

## Converting Units of Measure

*You can enter a conversion expression anywhere that an expression is valid.*

With the TI-86, you can convert a value measured in one unit into its equivalent value in another unit of measure. For example, you can convert inches to yards, quarts to liters, or degrees Fahrenheit to degrees Celsius.

The units of measure from which and to which you convert must be compatible. For example, you cannot convert inches to degrees Fahrenheit, or yards to calories. Each menu item on the CONV menu (page 62) represents a unit-of-measure group, such as length (**LNGLH**), volume (**VOL**), and pressure (**PRESS**). Within each menu, all units are compatible.

### Converting a Unit of Measure

To use any conversion instruction, the syntax is:

*(value)currentUnit*►*newUnit*

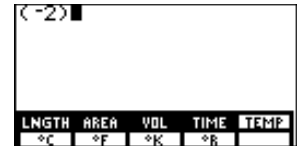
*In the example, -2 degrees Celsius is converted to degrees Fahrenheit. Always use parentheses when value is negative.*

- ① Enter the real *value* to be converted.
- ② Display the CONV menu.
- ③ Select the **TEMP** conversion group.

[ ] (-) 2 [ ]

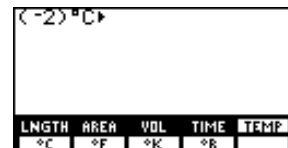
[2nd] [CONV]

[F5]



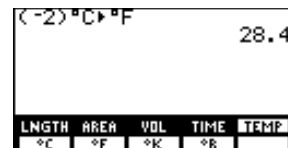
- 4 Select the current unit of measure ( $^{\circ}\text{C}$ ) from the conversion group menu. The unit abbreviation and conversion symbol ( $\blacktriangleright$ ) are pasted to the cursor location.

[F1]



- 5 Select the new unit of measure ( $^{\circ}\text{F}$ ) from the conversion group menu. The unit abbreviation is pasted to the cursor location.

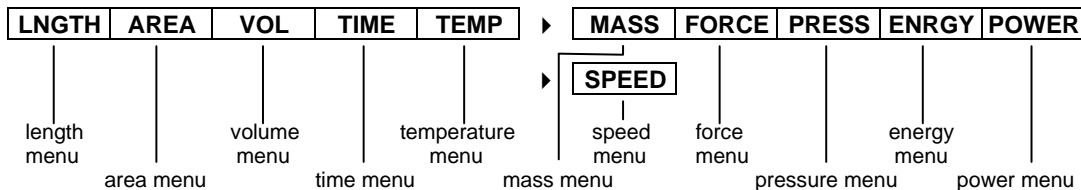
[F2]



- 6 Convert the measurement.

[ENTER]

### The CONV (Conversions) Menu $[\text{2nd}] [\text{CONV}]$



**Important:** When you convert a negative value, you must enclose in parentheses the value and its negation sign, as in  $(-4)$ . Otherwise, the TI-86 order of evaluation will perform the conversion first, and then apply the negation to the converted value.

If you enter...

...The TI-86 converts it to...

 $(-4)^{\circ}\text{C}^{\circ}\text{F}$ 24.8 degrees Fahrenheit ( $-4^{\circ}$  Celsius converted to degrees Fahrenheit) $-4^{\circ}\text{C}^{\circ}\text{F}$ -39.2 degrees Fahrenheit ( $4^{\circ}$  Celsius converted to degrees Fahrenheit, then negated)

**The CONV LENGH (Length) Menu** [2nd] [CONV] [F1]

<b>mm</b>	millimeters	<b>yd</b>	yards	<b>mil</b>	mils
<b>cm</b>	centimeters	<b>km</b>	kilometers	<b>Ang</b>	Angstroms
<b>m</b>	meters	<b>mile</b>	miles	<b>fermi</b>	fermis
<b>in</b>	inches	<b>nmile</b>	nautical miles	<b>rod</b>	rods
<b>ft</b>	feet	<b>lt-yr</b>	light-years	<b>fath</b>	fathoms

**The CONV AREA Menu** [2nd] [CONV] [F2]

<b>ft<sup>2</sup></b>	square feet	<b>km<sup>2</sup></b>	square kilometers	<b>cm<sup>2</sup></b>	square centimeters
<b>m<sup>2</sup></b>	square meters	<b>acre</b>	acres	<b>yd<sup>2</sup></b>	square yards
<b>mi<sup>2</sup></b>	square miles	<b>in<sup>2</sup></b>	square inches	<b>ha</b>	hectares

**The CONV VOL (Volume) Menu** [2nd] [CONV] [F3]

<b>liter</b>	liters	<b>cm<sup>3</sup></b>	cubic centimeters	<b>tsp</b>	teaspoons
<b>gal</b>	gallons	<b>in<sup>3</sup></b>	cubic inches	<b>tbsp</b>	tablespoons
<b>qt</b>	quarts	<b>ft<sup>3</sup></b>	cubic feet	<b>ml</b>	milliliters
<b>pt</b>	pints	<b>m<sup>3</sup></b>	cubic meters	<b>galUK</b>	UK gallons
<b>oz</b>	ounces	<b>cup</b>	cups	<b>ozUK</b>	UK ounces

**The CONV TIME Menu** [2nd] [CONV] [F4]

<b>sec</b>	seconds	<b>day</b>	days	<b>ms</b>	milliseconds
<b>mn</b>	minutes	<b>yr</b>	years	<b>μs</b>	microseconds
<b>hr</b>	hours	<b>week</b>	weeks	<b>ns</b>	nanoseconds

**The CONV TEMP (Temperature) Menu** [2nd] [CONV] [F5]

**°C** degrees Celsius    **°F** degrees Fahrenheit    **°K** degrees Kelvin    **°R** degrees Rankin

**The CONV MASS Menu** [2nd] [CONV] [MORE] [F1]

<b>gm</b>	grams	<b>amu</b>	atomic mass units	<b>ton</b>	tons
<b>kg</b>	kilograms	<b>slug</b>	slugs	<b>mton</b>	metric tons
<b>lb</b>	pounds				

**The CONV FORCE Menu** [2nd] [CONV] [MORE] [F2]

<b>N</b>	Newtons	<b>tonf</b>	ton force	<b>lbf</b>	pound force
<b>dyne</b>	dynes	<b>kgf</b>	kilogram force		

**The CONV PRESS (Pressure) Menu** [2nd] [CONV] [MORE] [F3]

<b>atm</b>	atmospheres	<b>lb/in<sup>2</sup></b>	pounds per square inch	<b>inHg</b>	inches of mercury
<b>bar</b>	bars	<b>mmHg</b>	millimeters of mercury	<b>inH<sub>2</sub>O</b>	inches of water
<b>N/m<sup>2</sup></b>	Newtons per square meter	<b>mmH<sub>2</sub>O</b>	millimeters of water		

**The CONV ENRGY (Energy) Menu** [2nd] [CONV] [MORE] [F4]

<b>J</b>	Joules	<b>ft-lb</b>	foot-pounds	<b>erg</b>	ergs
<b>cal</b>	calories	<b>kw-hr</b>	kilowatt hours	<b>l-atm</b>	liter-atmospheres
<b>Btu</b>	British thermal units	<b>eV</b>	electron Volts		

**The CONV POWER Menu** [2nd] [CONV] [MORE] [F5]

<b>hp</b>	horsepower	<b>ftlb/s</b>	foot-pounds per second	<b>Btu/m</b>	British thermal units per minute
<b>W</b>	Watts	<b>cal/s</b>	calories per second		

**The CONV SPEED Menu** [2nd] [CONV] [MORE] [MORE] [F1]

<b>ft/s</b>	feet per second	<b>mi/hr</b>	miles per hour	<b>knot</b>	knots
<b>m/s</b>	meters per second	<b>km/hr</b>	kilometers per hour		

To enter a forward slash (/), you can use the  $\frac{\square}{\square}$  key or paste it from the CATALOG.

### Converting a Value Expressed as a Rate

To convert a value expressed as a rate on the home screen, you can use parentheses and the division operator ( $\div$ ). For example, if a car travels 325 miles in 4 hours, and you want to know the rate of speed in kilometers per hour, enter this expression:

**(325/4)mi/hr** $\rightarrow$ **km/hr**      This expression returns **131 km/hr** (rounded up).

You also can return this result using only a forward slash, as in: **325mile** $\rightarrow$ **km/4hr** $\rightarrow$ **hr**

### Number Bases

The number base mode setting (Chapter 1) controls how the TI-86 interprets an entered number and displays results on the home screen. However, you can enter numbers in any number base using number base designators **b**, **o**, **d**, and **h**. Then you can display the result on the home screen in any number base using number base conversions.

All numbers are stored internally as decimal. If you perform an operation in a mode setting other than **Dec**, the TI-86 performs integer mathematics, truncating to an integer after every calculation and expression.

For example, in **Hex** mode, **1/3+7** returns **7h** (1 divided by 3, truncated to 0, and then added to 7).

### Number Base Ranges

Binary, octal, and hexadecimal numbers on the TI-86 are defined in these ranges.

Type	Low Value/High Value	Decimal Equivalent
Binary	1000 0000 0000 0001 <b>b</b>	-32,767
	0111 1111 1111 1111 <b>b</b>	32,767
Octal	5120 6357 4134 0001 <b>o</b>	-99,999,999,999,999
	2657 1420 3643 7777 <b>o</b>	99,999,999,999,999
Hexadecimal	<b>FFFF A50C EF85 C001h</b>	-99,999,999,999,999
	0000 <b>5AF3 107A 3FFFh</b>	99,999,999,999,999

### One's and Two's Complements

To obtain the one's complement of a binary number, enter the **not** function (page 68) before the number. For example, **not 111100001111** in **Bin** mode returns **1111000011110000b**.

To obtain the two's complement of a binary number, press  $\square$  before entering the number. For example, **-111100001111** in **Bin** mode returns **1111000011110001b**.

### The (Number) BASE Menu $\square$ [2nd] [BASE]

A-F	TYPE	CONV	BOOL	BIT
hexadecimal characters menu	base type menu	base conversion menu	Boolean operator menu	rotate/shift menu

BASE A-F menu items and BASE TYPE menu items are not the same as regular alphabetical characters.

In the example, the upper menu is the list editor menu ( $\text{[2nd] [LIST]}$  in Dec number base mode).

If Hex number base mode is not set, you must enter the h designator, even if the number contains a special hexadecimal character.

### The BASE A-F (Hexadecimal Characters) Menu $\text{[2nd] [BASE] [F1]}$

This is the BASE A-F menu displayed on the home screen. To use A, press  $\text{[2nd] [M1]}$ .

A	TYPE	CONV	BOOL	BIT
B	C	D	E	F

When an editor menu is the upper menu, A and B are combined in one cell. If you press  $\text{[F1]}$  or  $\text{[MORE]}$ ...

{	}	NAMES	"	OPS
A-B	C	D	E	F

...A and B move to two separate cells, and E and F are combined. To switch back, press  $\text{[F5]}$  or  $\text{[MORE]}$ .

{	}	NAMES	"	OPS
A	B	C	D	E-F

### Entering Hexadecimal Digits

To enter a hexadecimal number, use the number keys as you would for a decimal number. Select the hexadecimal characters A through F from the menu as needed.

### The BASE TYPE Menu $\text{[2nd] [BASE] [F2]}$

A-F	TYPE	CONV	BOOL	BIT
b	h	o	d	

In an expression, you can designate a number in any number base, regardless of mode. After you enter the number, select the appropriate base type symbol from the BASE TYPE menu. The base type symbol is pasted to the cursor location. Here are some examples.

In <b>Dec</b> mode (default):	$10\text{b}+10$ $\text{[ENTER]}$	12	In <b>Oct</b> mode:	$10\text{b}+10$ $\text{[ENTER]}$	12o
	$10\text{h}+10$ $\text{[ENTER]}$	26		$10\text{d}+10$ $\text{[ENTER]}$	22o
In <b>Bin</b> mode:	$10\text{h}+10$ $\text{[ENTER]}$	10010b	In <b>Hex</b> mode:	$10\text{b}+10$ $\text{[ENTER]}$	12h
	$10\text{d}+10$ $\text{[ENTER]}$	1100b		$10\text{d}+10$ $\text{[ENTER]}$	1Ah

**The BASE CONV (Conversion) Menu** [2nd] [BASE] [F3]

A-F	TYPE	CONV	BOOL	BIT
►Bin	►Hex	►Oct	►Dec	

*value can be an expression, list, vector, or matrix. For detailed syntax descriptions, refer to the A to Z Reference.*

*value*►Bin Displays *value* as binary  
*value*►Hex Displays *value* as hexadecimal  
*value*►Oct Displays *value* as octal  
*value*►Dec Displays *value* as decimal

**Converting Number Bases**

- ① In **Dec** mode, solve  $10b + Fh + 10o + 10$ .  $10b + Fh + 10o + 10$  [ENTER] 35
- ② Add 1 to the result and convert it to **Bin** number base display. Ans+1►Bin [ENTER] 100100b
- ③ Add 1 to the result and convert it to **Hex** number base display. Ans+1►Hex [ENTER] 25h
- ④ Add 1 to the result and convert it to **Oct** number base display. Ans+1►Oct [ENTER] 46o
- ⑤ Add 1 to the result and convert it to **Dec** number base display. Ans+1 [ENTER] 39

**The BASE BOOL (Boolean) Menu** [2nd] [BASE] [F4]

A-F	TYPE	CONV	BOOL	BIT
and	or	xor	not	

*valueA* and *valueB*      *valueA* or *valueB*      *valueA* xor *valueB*      not *value*



Both the argument and the result must be within defined number ranges (page 66).

## Results of Boolean Operations

When a Boolean expression is evaluated, the arguments are converted to hexadecimal integers and the corresponding bits of the arguments are compared, as this table shows.

If <i>valueA</i> is...	...and <i>valueB</i> is...	Results			
		and	or	xor	not ( <i>valueA</i> )
1	1	1	1	0	0
1	0	0	1	1	0
0	1	0	1	1	1
0	0	0	0	0	1

The result is displayed according to the current mode setting. For example:

◆ In **Bin** mode, **101 and 110** returns **100b**.

◆ In **Hex** mode, **5 and 6** returns **4h**.

## The BASE BIT Menu 2nd BASE F5

A-F	TYPE	CONV	BOOL	BIT
rotR	rotL	shftR	shftL	

**rotR** *value* Rotates *value* right

**rotL** *value* Rotates *value* left

**shftR** *value* Shifts *value* right

**shftL** *value* Shifts *value* left

Rotate and shift operate on 16 base digits. To minimize an overflow error, enter the argument in binary form.

## Using Complex Numbers

Variable names with complex numbers stored to them are listed on the VARS CPLX screen (Chapter 2).

Lists, matrices, and vectors can have complex elements.

A complex number has two components: real ( $a$ ) and imaginary ( $+bi$ ). On the TI-86, you enter the complex number  $a+bi$  as:

- ◆  $(real,imaginary)$  in rectangular form
- ◆  $(magnitude\angle angle)$  in polar form

You can enter a complex number in rectangular or polar form, regardless of the current complex number mode setting. The separator ( , or  $\angle$  ) determines the form.

- ◆ To enter rectangular form, separate *real* and *imaginary* with a comma ( , ).
- ◆ To enter polar form, separate *magnitude* and *angle* with an angle symbol ( $\angle$ ).

Each component (*real*, *imaginary*, *magnitude*, or *angle*) can be a real number or an expression that evaluates to a real number; expressions are evaluated when you press  $\boxed{\text{ENTER}}$ .

When **RectC** complex number mode is set, complex numbers are displayed in rectangular form, regardless of the form in which you enter them (as shown to the right).

```
(6,1)          (6,1)
(6∠1)
(3.24181383521,5.048...
```

When **PolarC** complex number mode is set, complex numbers are displayed in polar form, regardless of the form in which you enter them (as shown to the right).

```
(6,1)          (6,1)
(6.0827625303∠.16514...
(6∠1)          (6∠1)
```

### Complex Results

Complex numbers in results, including list, matrix, and vector elements, are displayed in the form (rectangular or polar) specified by the mode setting (Chapter 1) or by a display conversion instruction (page 61).

- ◆ When **Radian** angle mode is set, results are displayed as  $(magnitude\angle angle)$ .
- ◆ When **Degree** angle mode is set, results are displayed as  $(real,imaginary)$ .

The graph format settings **RectGC** and **PolarGC** (Chapter 5) determine the complex number form of graph screen coordinates.

For example, when **PolarC** and **Degree** modes are set, **(2,1)-(1∠45)** returns **(1.32565429614∠12.7643896828)**.

### Using a Complex Number in an Expression

- ◆ Enter the complex number directly.
- ◆ Use the ALPHA keys, alpha keys, and other character keys to enter a complex variable.
- ◆ Select a complex variable from the VARS CPLX screen.

### The CPLX (Complex Number) Menu 2nd [CPLX]

<b>conj</b>	<b>real</b>	<b>imag</b>	<b>abs</b>	<b>angle</b>	<b>▶</b>	<b>▶Rec</b>	<b>▶Pol</b>			
-------------	-------------	-------------	------------	--------------	----------	-------------	-------------	--	--	--

You can enter the name or a complex list, vector, or matrix as an argument for any CPLX menu item.

<b>conj</b> ( <i>real,imaginary</i> )	Returns the complex conjugate of a complex value, list, vector or matrix; the result is ( <i>real,-imaginary</i> )
<b>conj</b> ( <i>magnitude∠angle</i> )	Returns ( <i>magnitude∠-angle</i> )
<b>real</b> ( <i>real,imaginary</i> )	Returns the real portion of a complex number, list, vector, or matrix as a real number; the result is <i>real</i>
<b>real</b> ( <i>magnitude∠angle</i> )	Returns <i>magnitude</i> *cosine( <i>angle</i> )
<b>imag</b> ( <i>real,imaginary</i> )	Returns the imaginary (non-real) portion of a complex number, list, vector, or matrix as a real number; the result is <i>imaginary</i>
<b>imag</b> ( <i>magnitude∠angle</i> )	Returns <i>magnitude</i> *sine( <i>angle</i> )
<b>abs</b> ( <i>real,imaginary</i> )	(Absolute value) Returns the magnitude (modulus) of a complex number, list, vector, or matrix of complex numbers; the result is $\sqrt{(real^2+imaginary^2)}$
<b>abs</b> ( <i>magnitude∠angle</i> )	Returns <i>magnitude</i>

<b>angle</b> ( <i>real,imaginary</i> )	Returns the polar angle of a complex number, list, vector, or matrix calculated as $\tan^{-1}(\textit{imaginary}/\textit{real})$ (adjusted by $\pi$ in the second quadrant or $-\pi$ in the third quadrant); the result is $\tan^{-1}(\textit{imaginary}/\textit{real})$
<b>angle</b> ( <i>magnitude</i> $\angle$ <i>angle</i> )	Returns <i>angle</i> (where $-\pi < \textit{angle} \leq \pi$ )
<i>complexValue</i> <b>►Rec</b>	Displays <i>complexValue</i> in rectangular format ( <i>real,imaginary</i> ), regardless of complex mode setting; valid only at the end of a command and only when <i>complexValue</i> is indeed complex
<i>complexValue</i> <b>►Pol</b>	Displays <i>complexValue</i> in polar format ( <i>magnitude</i> $\angle$ <i>angle</i> ), regardless of complex mode setting; valid only at the end of a command and only when <i>complexValue</i> is indeed complex

Select { and } from the LIST menu.

You must enter commas to separate list elements.

You can enter a complex list, vector, or matrix directly. The syntax below is for lists. To enter a complex vector or matrix, substitute brackets for braces below and use the correct form for either data type (Chapters 12 and 13).

In rectangular form, to use lists of complex numbers with **conj**, **real**, **imag**, **abs**, and **angle**, the syntax is:

**conj**{(*realA,imaginaryA*),(*realB,imaginaryB*),(*realC,imaginaryC*),...}

In polar form, to use lists of complex numbers with **conj**, **real**, **imag**, **abs**, and **angle**, the syntax is:

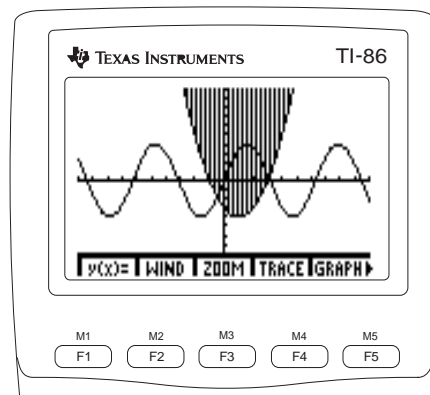
**real**{(*magnitudeA* $\angle$ *angleA*),(*magnitudeB* $\angle$ *angleB*),(*magnitudeC* $\angle$ *angleC*),...}

When you use a list the TI-86 calculates the result element by element and returns a list, in which each element is expressed according to the complex mode setting.

# 5

## Function Graphing

Defining a Graph.....	74
Setting the Graph Mode .....	74
The GRAPH Menu .....	75
Using the Equation Editor.....	76
Setting the Window Variables .....	81
Setting the Graph Format .....	83
Displaying a Graph .....	85



## Defining a Graph

This chapter describes the process for graphing functions in **Func** graphing mode, but the process is similar for each TI-86 graphing mode. Chapters 8, 9, and 10 describe the unique aspects of polar, parametric, and differential equation graphing modes. Chapter 6 describes various graphing tools, many of which you can use in all graphing modes.

*Some of these steps are not necessary every time you define a graph.*

- ❶ Set the graphing mode (page 74).
- ❷ Define, edit, or select one or more functions in the equation editor (pages 76 and 77).
- ❸ Select the graph style for each function (page 79).
- ❹ Deselect stat plots, if necessary (page 81).
- ❺ Set the viewing window variables (page 81).
- ❻ Select the graph format settings (page 83).

## Setting the Graph Mode

To display the mode screen, press  $\boxed{2\text{nd}}$  [MODE]. All default mode settings, including **Func** graphing mode, are highlighted in the picture to the right. The graphing modes are on the fifth line.

- ◆ **Func** (function graphing)
- ◆ **Pol** (polar graphing; Chapter 8)
- ◆ **Param** (parametric graphing; Chapter 9)
- ◆ **DifEq** (differential equation graphing; Chapter 10)

```

Normal Sci Eng
Float 012345678901
Radian Degree
RectC PolarC
Func Pol Param DifEq
Dec Bin Oct Hex
RectI CylU SphereU
dxDer1 dxNDer
  
```

Each graphing mode has a unique equation editor. You must select the graphing mode and **Dec** number base mode before you enter the functions. The TI-86 retains in memory all equations stored to the **Func**, **Pol**, **Param**, and **DifEq** equation editors. Each mode also has unique graph format settings and window variables.

Stat plot on/off status, zoom factors, mode settings, and tolerance apply to all graphing modes; changing the graphing mode does not affect them.

These mode settings affect graphing results.

- ◆ **Radian** or **Degree** angle mode affects the interpretation of some functions.
- ◆ **dxDer1** or **dxNDer** differentiation mode affects plotting of selected functions.

## The GRAPH Menu

GRAPH
-------

y(x)=	WIND	ZOOM	TRACE	GRAPH	▶	MATH	DRAW	FORMT	STGDB	RCGDB
					▶	EVAL	STPIC	RCPIC		

<b>y(x)=</b>	Displays the equation editor; use this screen to enter functions to be graphed
<b>WIND</b>	Displays the window editor; use this editor to change graph screen dimensions
<b>ZOOM</b>	Displays the GRAPH ZOOM menu; use these items to change the graph screen dimensions
<b>TRACE</b>	Activates the trace cursor; use this cursor to trace along the graph of a specific function
<b>GRAPH</b>	Displays the graph screen; graphs all selected functions and turned on stat plots
<b>MATH</b>	Displays the GRAPH MATH menu; use this menu to explore graphs mathematically
<b>DRAW</b>	Displays the GRAPH DRAW menu; use this menu to draw on graphs or test pixels

Chapter 1 describes all mode settings in detail.

Chapter 6 describes these GRAPH menu items:  
**ZOOM**, **TRACE**, **MATH**,  
**DRAW**, **STGDB**, **RCGDB**,  
**EVAL**, **STPIC**, and **RCPIC**.

<b>FORMT</b>	Displays the graph format screen; use this screen to select graph format settings
<b>STGDB</b>	Displays the <b>Name=</b> prompt and STGDB menu; use this prompt to enter a <b>GDB</b> variable
<b>RCGDB</b>	Displays the <b>Name=</b> prompt and RCGDB menu; use this menu to recall a graph database
<b>EVAL</b>	Displays the <b>Eval x=</b> prompt; enter an <b>x</b> for which you want to solve the current function
<b>STPIC</b>	Displays the <b>Name=</b> prompt and STPIC menu; use this prompt to enter a <b>PIC</b> variable
<b>RCPIC</b>	Displays the <b>Name=</b> prompt and RCPIC menu; use this menu to recall a picture

## Using the Equation Editor

To display the equation editor in **Func** graphing mode, select **y(x)=** from the GRAPH menu (**GRAPH** **F1**). The GRAPH menu shifts up and the equation editor menu is displayed as the lower menu. You can store up to 99 functions in the equation editor, if sufficient memory is available.



If a function is selected, its equals sign (=) is highlighted in the equation editor. If the function is deselected, its equals sign is not highlighted. Only selected functions are plotted when the TI-86 plots a graph.

### The Equation Editor (**GRAPH y(x)=**) Menu **GRAPH** **F1**

<b>y(x)=</b>	<b>WIND</b>	<b>ZOOM</b>	<b>TRACE</b>	<b>GRAPH</b>	
x	y	INSf	DELf	SELCT	▶ <b>ALL+</b> <b>ALL-</b> <b>STYLE</b>



<b>x</b>	Pastes the variable <b>x</b> to the current cursor location (same as $\boxed{\text{x-VAR}}$ or $\boxed{2\text{nd}}$ [alpha] [X])
<b>y</b>	Pastes the variable <b>y</b> to the current cursor location (same as $\boxed{2\text{nd}}$ [alpha] [Y])
<b>INSf</b>	Inserts a deleted equation variable (function) name above the current cursor location (only the variable name is inserted)
<b>DELf</b>	Deletes the function that the cursor is on
<b>SELECT</b>	Changes the selection status of the function that the cursor is on (selects or deselects)
<b>ALL+</b>	Selects all defined functions in the equation editor
<b>ALL-</b>	Deselects all defined functions in the equation editor
<b>STYLE</b>	Assigns the next of seven available graph styles to the function that the cursor is on

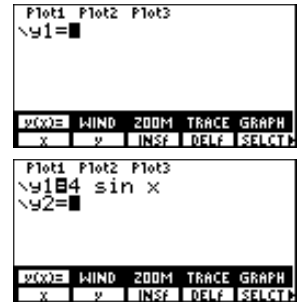
### Defining a Function in the Equation Editor

To move from the first equation editor function to the last, press  $\boxed{2\text{nd}}$   $\boxed{\Delta}$ .

To move to the beginning or end of an equation, press  $\boxed{2\text{nd}}$   $\boxed{\leftarrow}$  or  $\boxed{2\text{nd}}$   $\boxed{\rightarrow}$ .

An ellipsis indicates that an equation continues beyond the screen.

- 1 Display the equation editor.  $\boxed{\text{GRAPH}}$   $\boxed{\text{F1}}$
- 2 If functions are stored in the equation editor, move the cursor down until a blank function is displayed.  $\boxed{\downarrow}$  or  $\boxed{\text{ENTER}}$
- 3 Enter an equation in terms of **x** to define the function. When you enter the first character, the function is selected automatically. (The function's equals sign is highlighted.) 4  $\boxed{\text{SIN}}$   $\boxed{\text{x-VAR}}$
- 4 Move the cursor to the next function.  $\boxed{\text{ENTER}}$  or  $\boxed{\downarrow}$



### Notes about Defining Function Equations

- ◆ You can include functions, variables, constants, matrices, matrix elements, vectors, vector elements, lists, list elements, complex values, or other equations in the equation.
- ◆ If you include matrices, vectors, or complex values, the equation must evaluate to a real number at each point.
- ◆ You can include another defined function in an equation. For example, given  $y_1 = \sin x$  and  $y_2 = 4 + y_1$ , the function  $y_2$  would equal 4 plus the sine of  $x$ .
- ◆ To enter a function name, select **y** from the equation editor menu, and then enter the appropriate number.
- ◆ To insert the contents of an equation variable, use RCL (Chapter 1). To enter the equation variable at the **Rcl** prompt, use the ALPHA keys, alpha keys, and other character keys.
- ◆ To select all functions from the home screen or in the program editor, select **FnOn** from the CATALOG (or enter the individual characters) and press **ENTER**.
- ◆ To select specific functions from the home screen or in the program editor, select **FnOn** from the CATALOG (or enter the individual characters), enter the number of each function, and press **ENTER**. For example, to select  $y_1$ ,  $y_3$ , and  $y_5$ , enter **FnOn 1,3,5**.
- ◆ To deselect functions from the home screen or in the program editor, use **FnOff** the same way you use **FnOn** to select functions.
- ◆ When a function evaluates to a non-real number, the value is not plotted on the graph; no error is returned.

*You can edit expressions you inserted using **Rcl**.*

The TI-86 graphs all selected functions on the same graph screen.

## Selecting Graph Styles

Depending on which graphing mode is set, the TI-86 offers up to seven distinct graph styles. You can assign these styles to specific functions to visually differentiate each from the others.

For example, you can show **y1** as a connected line (↘**y1=** in the equation editor) and **y2** as a dotted line (⋯**y2=**), and shade the area above **y3** (▒**y3=**).

Also, you can manipulate the styles to illustrate actual phenomena graphically, such as a ball flying through the air (using ⦿) or the circular movement of a chair on a Ferris wheel (using ⦿).

Icon	Style	Characteristics of the Plotted Function
↘	Line	A solid line connects each plotted point; this is the default in <b>Connected</b> mode
▒	Thick	A thick solid line connects each plotted point
▒	Above	Shades the area above the function
▒	Below	Shades the area below the function
⦿	Path	A circle cursor traces the leading edge of the function and draws a path as it plots
⦿	Animate	A circle cursor traces the leading edge of the function as it plots; does not draw a path
⋯	Dot	A small dot represents each plotted point; this is the default in <b>Dot</b> mode

▒ (shade above) and  
▒ (shade below) are available  
only in **Func** graphing mode.

⋯ (dot) is available in all  
graphing modes except  
**DifEq** graphing mode.

To set the graph style from a program, select **GrStil** from the CATALOG (A to Z Reference).

In the example,  $\text{■}$  (shade above) is selected for **y2**. All window variables are set to the default values (page 82).

### Setting the Graph Style in the Equation Editor

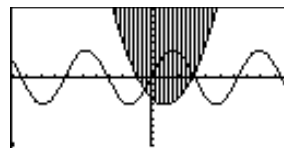
- ① Display the equation editor. GRAPH F1
- ② Move the cursor to the function or functions for which you want to set the graph style. ▼
- ③ Display the equation editor menu item **STYLE**. MORE
- ④ Select **STYLE** repeatedly to scroll the graph style icons to the left of the equation name. F3 F3
- ⑤ View the graph with the new graph style. 2nd F5
- ⑥ Clear the GRAPH menu to view the graph only. CLEAR

```

Plot1 Plot2 Plot3
y1=sin x
y2=x^2-2x-3
  
```

```

Plot1 Plot2 Plot3
y1=4sin x
y2=x^2-2x-3
STYLE
WIND ZOOM TRACE GRAPH
ALL- ALL- STYLE
  
```



### Using Shading Patterns to Differentiate Functions

When you select  $\text{■}$  (shade above) or  $\text{■}$  (shade below) for more than one function, the TI-86 rotates through a series of four shading patterns.

- ◆ First shaded function: vertical lines
- ◆ Second shaded function: horizontal lines
- ◆ Third shaded function: negatively sloping diagonal lines
- ◆ Fourth shaded function: positively sloping diagonal lines

The rotation returns to vertical lines for the fifth shaded function and repeats the order.

If you assign  $\text{■}$  or  $\text{■}$  to a function that graphs a family of curves (page 86), the same pattern rotation applies to the members of the family of curves.

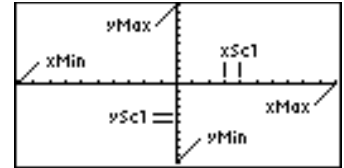
## Viewing and Changing On/Off Status of Stat Plots

**Plot1 Plot2 Plot3** on the top line of the equation editor displays the on/off status of each stat plot (Chapter 14). When a plot name is highlighted on this line, the plot is on.

To change the on/off status of a stat plot from the equation editor, press  $\uparrow$ ,  $\downarrow$ , and  $\leftarrow$  to place the cursor on **Plot1**, **Plot2**, or **Plot3**, and then press  $\boxed{\text{ENTER}}$ .

## Setting the Window Variables

The graph screen window represents the portion of the coordinate plane displayed on the graph screen. By setting window variables, you can define the graph screen window boundaries and other attributes.



**xMin**, **xMax**, **yMin**, and **yMax** are the graph screen boundaries.

**xSc1** (x scale) is the number of units represented by the distance from one tick mark to the next tick mark on the x-axis.

**ySc1** (y scale) is the number of units represented by the distance from one tick mark to the next tick mark on the y-axis.

**xRes** sets pixel resolution for function graphs only, using integers **1** through **8**.

- ◆ At **xRes=1** (the default), functions are evaluated and graphed at each pixel on the x-axis.
- ◆ At **xRes=8**, functions are evaluated and graphed at every eighth pixel along the x-axis.

*To remove tick marks from both axes, set **xSc1=0** and **ySc1=0**.*

*Small **xRes** values improve graph resolution but may cause the TI-86 to plot graphs more slowly.*

## Displaying the Window Editor

To display the window editor, select **WIND** from the GRAPH menu ( $\boxed{\text{GRAPH}}$   $\boxed{\text{F2}}$ ). Each graphing mode has a unique window editor. The window editor to the right shows the default values in **Func** graphing mode.  $\downarrow$  indicates that **xRes=1** (x resolution) is below **ySc1** on the window editor.

```

WINDOW
xMin=-10
xMax=10
xSc1=1
yMin=-10
yMax=10
↓ySc1=1
┌──┴──┐
└──┬──┘
WIND ZOOM TRACE GRAPH
  
```

## Changing a Window Variable Value

**xMin<xMax** and **yMin<yMax** both must be true to graph successfully.

In the example, **yMin** is changed to **0**.

- 1 Display the window editor.  $\boxed{\text{GRAPH}}$   $\boxed{\text{F2}}$
- 2 Move the cursor to the window variable you want to change.  $\downarrow$   $\downarrow$   $\downarrow$
- 3 Edit the value, which can be an expression. **0**
- 4 Evaluate any expressions and store the value.  $\boxed{\text{ENTER}}$  or  $\downarrow$

```

WINDOW
xMin=-10
xMax=10
xSc1=1
yMin=0
yMax=10
↓ySc1=1
┌──┴──┐
└──┬──┘
WIND ZOOM TRACE GRAPH
  
```

To change a window variable value from the home screen or in the program editor, enter the value, and then press  $\boxed{\text{STO}\rightarrow}$ . Either select the window variable from the VARS WIND screen ( $\boxed{2\text{nd}}$   $\boxed{\text{CATLG-VARS}}$   $\boxed{\text{MORE}}$   $\boxed{\text{MORE}}$  **WIND**) or enter individual characters. Press  $\boxed{\text{ENTER}}$ .

### Setting Graphing Accuracy with $\Delta x$ and $\Delta y$

The window variables  $\Delta x$  and  $\Delta y$  define the distance from the center of one pixel to the center of any adjacent pixel. When you display a graph, the values of  $\Delta x$  and  $\Delta y$  are calculated from **xMin**, **xMax**, **yMin**, and **yMax** using these formulas:

$$\Delta x = (\text{xMin} + \text{xMax}) / 126$$

$$\Delta y = (\text{yMin} + \text{yMax}) / 62$$

$\Delta x$  and  $\Delta y$  are not on the window editor. To change them, you must follow the steps above for changing a window variable value from the home screen or in the program editor. When you change the values stored to  $\Delta x$  and  $\Delta y$ , the TI-86 automatically recalculates **xMax** and **yMax** from  $\Delta x$ , **xMin**,  $\Delta y$ , and **yMin**, and the new values are stored.

### Setting the Graph Format

*The TI-86 retains independent format settings for each graphing mode.*

*In **DifEq** graphing mode, the graph format screen key sequence is **GRAPH** **MORE** **F1** (Chapter 10).*

To display the graph format screen, select **FORMT** from the **GRAPH** menu (**GRAPH** **MORE** **F3**). The graph format settings define various characteristics of the displayed graph. The current settings are highlighted.

To change a setting, move the cursor onto the new setting, and then press **ENTER**, the same as on the mode screen.



*DifEq graphing mode has a unique set of graph format settings (Chapter 10).*

<b>RectGC</b>	Displays the cursor location as rectangular graph coordinates <b>x</b> and <b>y</b> ; when <b>RectGC</b> is set, plotting the graph, moving the free-moving cursor, and tracing update <b>x</b> and <b>y</b> ; if <b>CoordOn</b> format also is selected, <b>x</b> and <b>y</b> are displayed
<b>PolarGC</b>	Displays the cursor location as polar graph coordinates <b>R</b> and $\theta$ ; when <b>PolarGC</b> is set, plotting the graph, moving the free-moving cursor, and tracing update <b>x</b> , <b>y</b> , <b>R</b> and $\theta$ ; if <b>CoordOn</b> format also is selected, <b>R</b> and $\theta$ are displayed
<b>CoordOn</b>	Displays the cursor coordinates at the bottom of the graph
<b>CoordOff</b>	Does not display the cursor coordinates at the bottom of the graph
<b>DrawLine</b>	Draws a line between the points calculated for the functions in the equation editor
<b>DrawDot</b>	Plots only the calculated points for the functions in the equation editor
<b>SeqG</b>	(sequential graphing) Evaluates and plots one function completely before evaluating and plotting the next function
<b>SimulG</b>	(simultaneous graphing) Evaluates and plots all selected functions for a single value of <b>x</b> and then evaluates and plots them for the next value of <b>x</b>
<b>GridOff</b>	Omits the grid points from the display
<b>GridOn</b>	Displays grid points
<b>AxesOn</b>	Displays the axes
<b>AxesOff</b>	Omits the axes from the display; <b>AxesOff</b> overrides the <b>LabelOff/LabelOn</b> format setting
<b>LabelOff</b>	Omits the axis labels from the display
<b>LabelOn</b>	Labels the axes, if <b>AxesOn</b> is also selected; <b>x</b> and <b>y</b> for <b>Func</b> , <b>Pol</b> , and <b>Param</b> modes; various labels in <b>DifEq</b> mode

*Grid points cover the graph screen in rows that correspond to the tick marks on each axis.*



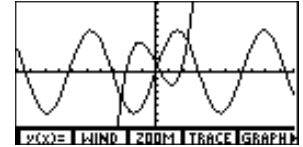
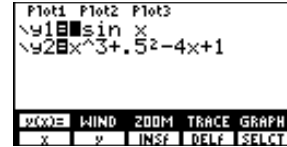
In the example graph to the right, all default settings related to graphing are set.

To view the graph without the GRAPH menu on the bottom line, press **[CLEAR]** after plotting the graph.

When you pause, the busy indicator in the top-right corner becomes a dotted line.

## Displaying a Graph

To display a graph, select **GRAPH** from the GRAPH menu. The graph screen is displayed. If the graph is newly defined, the busy indicator is displayed at the top-right corner as the TI-86 draws the graph.



- ◆ In **SeqG** format, the TI-86 draws each selected function one by one, in function-name order (for example, **y1** is graphed first, **y2** is graphed second, and so on).
- ◆ In **SimulG** format, the TI-86 draws all selected graphs simultaneously.

You can display and explore a graph from a program (Chapter 16). To use graphing commands on the home screen, select them from the CATALOG or entering the individual characters.

## Pausing or Stopping a Graph in Progress

- ◆ To pause graph plotting, press **[ENTER]**. To resume plotting, press **[ENTER]** again.
- ◆ To stop graph plotting, press **[ON]**. To replot, select **GRAPH** from the GRAPH menu.

## Modifying a Drawn Graph

To remove these items from the graph screen:

Cursor, coordinate values, or menus (To restore menus, press **[EXIT]** or **[GRAPH]**)

Free-moving cursor and coordinate values but not the menus

Cursor and coordinate values but not the menus

Press (or select):

**[CLEAR]**

**[ENTER]**

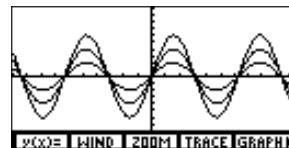
**[GRAPH]** or **GRAPH**

## Graphing a Family of Curves

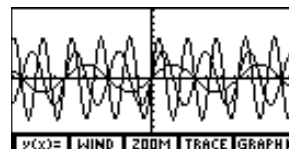
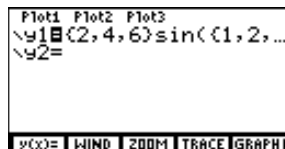
If you enter a list as an element in an equation, the TI-86 plots the function for each value in the list, graphing a family of curves. In **SimulG** graphing order mode, the TI-86 graphs all functions sequentially for the first element in each list, then for the second element, and so on.

*When you use more than one list in an expression, all lists must have the same dimension.*

For example,  $\{2,4,6\} \sin x$  graphs three functions:  
 $2 \sin x$ ,  $4 \sin x$ , and  $6 \sin x$ .



The equation  $\{2,4,6\} \sin (\{1,2,3\} x)$  also graphs three functions:  
 $2 \sin x$ ,  $4 \sin (2x)$ , and  $6 \sin (3x)$ .



## Smart Graph

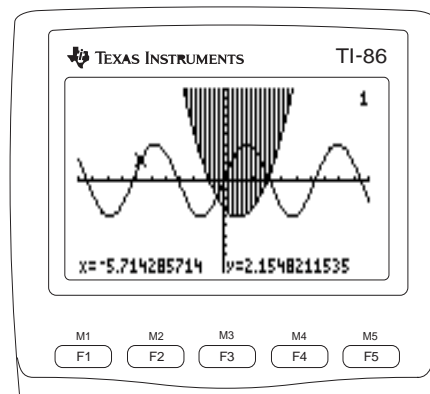
Smart Graph displays the previously displayed graph when you press **[GRAPH]**, as long as all factors that would cause replotting are unchanged since the graph was last displayed. Smart Graph replots if you performed any of these actions since the graph was last displayed.

- ◆ Changed a mode setting that affects graphs
- ◆ Changed a function or stat plot that was plotted on the last graph screen
- ◆ Selected or deselected a function or stat plot
- ◆ Changed the value of a variable in a selected function
- ◆ Changed the value of a window variable setting
- ◆ Changed a graph format setting

# 6

## Graph Tools

Graph Tools on the TI-86.....	88
Tracing a Graph .....	90
Resizing the Graph Screen with ZOOM Operations .....	91
Using Interactive Math Functions .....	95
Evaluating a Function for a Specified x.....	101
Drawing on a Graph .....	101



## Graph Tools on the TI-86

Chapter 5 describes how to use the GRAPH menu items **y(x)=**, **WIND**, **GRAPH**, and **FORMT** to define and display the graph of a function in **Func** graphing mode. This chapter describes how to use the other GRAPH menu items to apply preset graph screen dimensions, explore the graph and trace specific functions, perform mathematical analyses, draw on graphs, and store and recall graphs and drawings. You can use most graph tools in all four graphing modes.

### The GRAPH Menu

GRAPH

y(x)=	WIND	ZOOM	TRACE	GRAPH	▶	MATH	DRAW	FORMT	STGDB	RCGDB	
						▶	EVAL	STPIC	RCPIC		

*This is the GRAPH menu in **Func** graphing mode. The GRAPH menu differs slightly from graphing mode to graphing mode.*

- ZOOM** Displays the GRAPH ZOOM menu; use these items to apply preset graph screen dimensions
- TRACE** Activates the trace cursor; use this cursor to trace along graphs of specific functions
- MATH** Displays the GRAPH MATH menu; use this menu to explore graphs mathematically
- DRAW** Displays the GRAPH DRAW menu; use this menu to draw on graphs
- STGDB** Displays the **Name=** prompt and GDB menu; use this prompt to enter a **GDB** variable
- RCGDB** Displays the **Name=** prompt and GDB menu; use this menu to recall a **GDB** variable
- EVAL** Displays the **Eval x=** prompt; use this prompt to enter an **x** value for which you want to solve the current function
- STPIC** Displays the **Name=** prompt and PIC menu; use this prompt to enter a **PIC** variable
- RCPIC** Displays the **Name=** prompt and PIC menu; use this menu to recall **PIC** variable

In the example, the function  $y(x)=x^3+3x^2-4x$  is graphed.

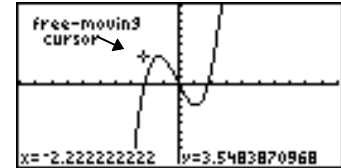
The numeric display mode settings do not affect coordinate display.

### Using the Free-Moving Cursor

When you select **GRAPH** from the GRAPH menu, the graph screen is displayed with the free-moving cursor at the center of the screen.

The cursor appears as a plus sign with a flashing center pixel. To move the cursor, press  $\leftarrow$ ,  $\rightarrow$ ,  $\uparrow$ , or  $\downarrow$ ; it moves in the direction of the cursor key you press.

- ◆ In **RectGC** format, each cursor movement updates the variables **x** and **y**. In **PolarGC** format, each cursor movement updates **x**, **y**, **R**, and  $\theta$ .
- ◆ In **CoordOn** format, the **x** and **y** cursor coordinates are displayed at the bottom of the graph screen as you move the cursor.



### Graphing Accuracy

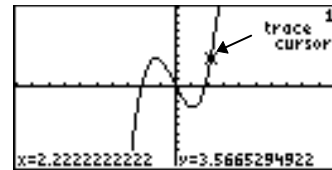
The coordinate values displayed as you move the cursor approximate actual mathematical coordinates, accurate to within the width and height of the pixel. As the difference between **xMin** and **xMax** and between **yMin** and **yMax** becomes smaller (for example, when you zoom in on a graph), graphing is more accurate and coordinate values approximate the actual mathematical coordinates more closely.

The free-moving cursor coordinates represent the cursor location on the graph screen. Moving the free-moving cursor precisely from one plotted point to the next along a function is very difficult. To move along a function easily, use the trace cursor (page 90).

## Tracing a Graph

In the example, the function  $y(x)=x^3+3x^2-4x$  is graphed.

To display the graph and begin a trace, select **TRACE** from the GRAPH menu. The trace cursor appears as a small square with a flashing diagonal line at each corner. Initially, the trace cursor appears on the first selected function, at the  $x$  value closest to the middle of the screen.



If **CoordOn** format is selected, the cursor coordinates are displayed at the bottom of the screen.

When you enter the first character of an independent variable value, an  $x=$  prompt is displayed (or  $\theta=$  or  $t=$ ). The value can be an expression.

If the function is undefined at an  $x$  value, then the  $y$  value is blank.

To move the trace cursor...

To the next larger or next smaller plotted point in a function

To any valid independent-variable value ( $x$ ,  $\theta$ , or  $t$ ) on the current equation

From one function to another function at  $x$ , in the order or reverse order of the selected functions in the equation editor

From one member to another member of a family of curves (Chapter 5)

Press these keys:

or

value

or

or

As you move the trace cursor along a function, the  $y$  value is calculated from the  $x$  value. That is,  $y=y(x)$ . When you trace beyond the top or bottom of the graph screen, the coordinates displayed on the screen continue to change as if the cursor were still on the screen.

**Panning:** To view function coordinates to the left or right of the current graph screen, press and hold  or  while tracing. When you pan beyond the left or right side of the screen during a trace, the TI-86 automatically changes the values of **xMin** and **xMax**.

**Quick Zoom:** While tracing, you can press  $\boxed{\text{ENTER}}$  to adjust the graph screen so that the trace cursor location becomes the center of a new graph screen, even if you have moved the cursor beyond the top or bottom of the display. In effect, this is vertical panning.

### Stopping and Resuming a Trace

To stop tracing and restore the free-moving cursor, press  $\boxed{\text{CLEAR}}$  or  $\boxed{\text{GRAPH}}$ .

To resume tracing, select **TRACE** from the GRAPH menu. If Smart Graph has not replotted the graph (Chapter 5), the trace cursor is at the point where you stopped tracing.

## Resizing the Graph Screen with ZOOM Operations

The standard TI-86 graph screen displays the portion of the xy plane defined by the values stored to the window variables. With the GRAPH ZOOM menu items, you can change some or all of the window variable values and redisplay the graph, usually with one keystroke. As a result, a smaller or larger portion of the xy plane is displayed.

*To view the current window variable values, select **WIND** from the GRAPH menu.*

### The GRAPH ZOOM Menu $\boxed{\text{GRAPH}}$ $\boxed{\text{F3}}$

$y(x)=$	WIND	ZOOM	TRACE	GRAPH	
BOX	ZIN	ZOUT	ZSTD	ZPREV	▶ $\boxed{\text{ZFIT}}$ $\boxed{\text{ZSQR}}$ $\boxed{\text{ZTRIG}}$ $\boxed{\text{ZDECM}}$ $\boxed{\text{ZDATA}}$
					▶ $\boxed{\text{ZRCL}}$ $\boxed{\text{ZFACT}}$ $\boxed{\text{ZOOMX}}$ $\boxed{\text{ZOOMY}}$ $\boxed{\text{ZINT}}$
					▶ $\boxed{\text{ZSTO}}$ $\boxed{\phantom{\text{ZSTO}}}$ $\boxed{\phantom{\text{ZSTO}}}$ $\boxed{\phantom{\text{ZSTO}}}$ $\boxed{\phantom{\text{ZSTO}}}$

To cancel the effect of any ZOOM menu item and return to the default window variable values, select **ZSTD**.

If you graph a circle but it appears elliptical, you can use **ZSQR** to reset the window variable values so that the circle graph appears circular.

<b>BOX</b>	Draws a box to define the graph screen
<b>ZIN</b>	(zoom in) Magnifies the graph around the cursor by factors of <b>xFact</b> and <b>yFact</b>
<b>ZOUT</b>	(zoom out) Displays more of the graph around the cursor by factors of <b>xFact</b> and <b>yFact</b>
<b>ZSTD</b>	Displays the graph in standard dimensions; resets the default window variable values
<b>ZPREV</b>	Reverses the last zoom operation; window variables revert to previous values
<b>ZFIT</b>	Recalculates <b>yMin</b> and <b>yMax</b> to include the minimum and maximum <b>y</b> values of the selected functions between the current <b>xMin</b> and <b>xMax</b>
<b>ZSQR</b>	Sets equal-size pixels on the x-axis and y-axis; adjusts window variable values in one direction so that $\Delta x = \Delta y$ , while <b>xScl</b> and <b>yScl</b> remain unchanged; the midpoint of the current graph (not the axes intersection) becomes the midpoint of the new graph
<b>ZTRIG</b>	Sets built-in window variables appropriate for trigonometric functions in <b>Radian</b> mode: <b>xMin</b> =-8.24668071567 <b>xScl</b> =1.5707963267949 ( $\pi/2$ ) <b>yMax</b> =4 <b>xMax</b> =8.24668071567 <b>yMin</b> =-4 <b>yScl</b> =1
<b>ZDECM</b>	Sets $\Delta x = .1$ , $\Delta y = .1$ , <b>xMin</b> =-6.3, <b>xMax</b> =6.3, <b>xScl</b> =1, <b>yMin</b> =-3.1, <b>yMax</b> =3.1, and <b>yScl</b> =1
<b>ZDATA</b>	Sets window variable values to display all statistical data points; adjusts <b>xMin</b> and <b>xMax</b> only; applies to histograms, scatter plots, and stat plots only (Chapter 14)
<b>ZRCL</b>	Uses window variable values stored in the user-defined zoom-window variables ( <b>ZSTO</b> )
<b>ZFACT</b>	Displays the ZOOM FACTORS screen
<b>ZOOMX</b>	Zooms out by a factor of <b>xFact</b> only; ignores <b>yFact</b> (page 93)
<b>ZOOMY</b>	Zooms out by a factor of <b>yFact</b> only; ignores <b>xFact</b>
<b>ZINT</b>	Sets integer values on the axes; sets $\Delta x = 1$ , $\Delta y = 1$ , <b>xScl</b> =10, and <b>yScl</b> =10; the current cursor becomes the center of the new graph screen after you press <b>ENTER</b>
<b>ZSTO</b>	Stores current window variable values to user-defined zoom-window variables ( <b>ZRCL</b> )



Before you begin these steps, enter a function in the equation editor. In the example, the function  $y(x)=x^3+3x^2-4x$  is graphed.

To cancel **BOX** without redefining the graph screen, press **CLEAR**.

When you replot the graph, the TI-86 updates the window variable values.

## Defining a Custom Zoom In

Using **BOX**, you can zoom in on any rectangular area within the current graph screen.

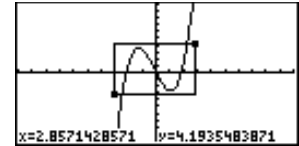
- 1 Select **BOX** from the GRAPH ZOOM menu. The zoom cursor is displayed at center screen.
- 2 Move the cursor to any spot you want to define as a corner of the zoom box; mark the corner with a small square.
- 3 Move the cursor away from the first corner, creating an adjustable box whose diagonal corners are the small square and the cursor.
- 4 When you have defined the box, replot all selected functions in the new graph screen.
- 5 Clear the menus from the screen.

**GRAPH** **F3**

**F1**

▸ ▾ ◀ ▶

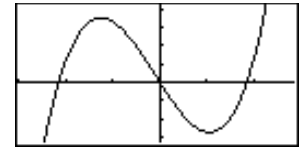
**ENTER**



▸ ▾ ◀ ▶

**ENTER**

**CLEAR**



## Setting Zoom Factors

Zoom factors define the magnification or reduction factor by which **ZIN**, **ZOUT**, **ZoomX**, and **ZoomY** zoom in or zoom out around a point. To display the zoom factors editor, select **ZFACT** from the GRAPH ZOOM menu (press **GRAPH** **F3** **MORE** **MORE** **F2**). **xFact** and **yFact** must be  $\geq 1$ . The default value for both factors is **4** in all graphing modes.

## Zooming In and Zooming Out on a Graph

**ZIN** magnifies the part of the graph surrounding the cursor location. **ZOUT** displays a greater portion of the graph, centered on the cursor location. **xFact** and **yFact** determine the extent. The steps below describe how to use **ZIN**. To use **ZOUT**, select it instead of **ZIN** in step 2.

To store to **xFact** or **yFact** from the home screen or in the program editor, you can select it from the VARS ALL screen or enter it using ALPHA and alpha keys.

In the example, the function  $y(x)=x^3+.3x^2-4x$  is graphed.

When you select a ZOOM feature, Smart Graph displays the current graph.

To cancel a zoom before you complete it, press **[CLEAR]**.

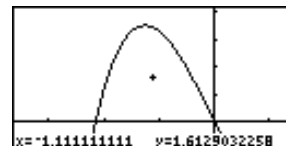
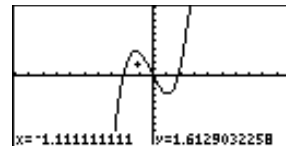
- 1 Check **xFact** and **yFact**; change as needed.
- 2 Select **ZIN** from the GRAPH ZOOM menu to display the zoom cursor.
- 3 Move the zoom cursor to the intended new center point of the graph screen.
- 4 Zoom in. The TI-86 adjusts the graph screen by **xFact** and **yFact**, updates window variable values, and replots the selected functions centered on the cursor location.

**[GRAPH]** **[F3]**  
**[MORE]** **[MORE]**  
**[F2]**

**[F3]** **[F2]**  
**[▶]** **[▼]** **[◀]** **[▲]**

**[ENTER]**

```
ZOOM FACTORS
xFact=4
yFact=4
```



You can continue to zoom in (or zoom out) on the current graph, unless you press a key other than **[ENTER]**, **[▶]**, **[▼]**, **[◀]**, or **[▲]**.

- ◆ To zoom in (or zoom out) again at the same point, press **[ENTER]**.
- ◆ To zoom in (or zoom out) at a new center point, move the cursor and press **[ENTER]**.

To zoom out only on the horizontal axis by a factor of **xFact**, select **ZOOMX** instead of **ZIN** in step 2 above. **ZOOMX** plots the selected functions centered on the cursor location and updates some window variable values; **yMin** and **yMax** are unchanged.

To zoom out only on the vertical axis by a factor of **yFact**, select **ZOOMY** instead of **ZIN** in step 2 above. **ZOOMY** plots the selected functions centered on the cursor location and updates some window variable values; **xMin** and **xMax** are unchanged.

You can select all zoom-window variables from the VARS WIND screen in any graph mode.

You also can enter the variable characters individually.

The zoom-window variables resume their standard default values when you reset defaults.

## Storing and Recalling Zoom-Window Variable Values

- ◆ To store all current zoom-window variable values simultaneously as a user-defined custom zoom feature, select **ZSTO** from the GRAPH ZOOM menu.
- ◆ To execute a user-defined custom zoom, which resets the graph screen to the stored zoom-window variables, select **ZRCL** from the GRAPH ZOOM menu.

Using **ZSTO** in these graphing modes:

**Func**, **Pol**, **Param**, and **DifEq** graphing modes

**Pol** graphing mode only

**Param** graphing mode only

**DifEq** graphing mode only

Stores to these zoom-window variables:

**zxMin**, **zxMax**, **zxScl**, **zyMin**, **zyMax**, and **zyScl**

**zθMin**, **zθMax**, and **zθStep**

**ztMin**, **ztMax**, and **ztStep**

**ztMin**, **ztMax**, **ztStep**, and **ztPlot**

## Using Interactive Math Functions

When you select a GRAPH MATH operation, Smart Graph displays the current graph with the trace cursor. To perform the GRAPH MATH operation, press  $\square$  and  $\square$  to move to the function.

When a GRAPH MATH menu operation prompts you to specify left bound, right bound, and guess, the accuracy of the values you specify will affect the length of time the TI-86 spends calculating the answer; the better the guess, the shorter the calculation time.

The **GRAPH MATH** Menu  $\square$   $\square$   $\square$

<b>MATH</b>	<b>DRAW</b>	<b>FORMT</b>	<b>STGDB</b>	<b>RCGDB</b>
ROOT	dy/dx	f(x)	FMIN	FMAX

▶	INFLC	YICPT	ISECT	DIST	ARC
▶	TANLN				

The GRAPH MATH menu differs slightly for **Pol** and **Param** graphing modes (Chapters 8 and 9).

**DifEq** graphing mode has no GRAPH MATH menu.

<b>ROOT</b>	Finds the root of a function using a specified left bound, right bound, and guess
<b>dy/dx</b>	Finds a numeric derivative (slope) of a function at the trace cursor location
<b>∫f(x)</b>	Finds a function's numerical integral using a specified left bounds and right bound
<b>FMIN</b>	Finds a function's minimum using a specified left bound, right bound, and guess
<b>FMAX</b>	Finds a function's maximum using a specified left bound, right bound, and guess
<b>INFLC</b>	Finds a function's inflection point using a specified left bound, right bound, and guess
<b>YICPT</b>	Finds a function's y-intercept ( <b>y</b> at <b>x=0</b> )
<b>ISECT</b>	Finds the intersection of two functions using a specified left bound, right bound, and guess
<b>DIST</b>	Finds the straight-line distance between a specified left bound and right bound
<b>ARC</b>	Finds the distance along a function between two specified points on the function
<b>TANLN</b>	Draws the tangent line at a specified point

### Settings That Affect GRAPH MATH Operations

- ◆ The tolerance variable **tol** (Appendix) affects the accuracy of **∫f(x)**, **FMIN**, **FMAX**, and **ARC**. Accuracy increases as the tolerance value becomes smaller.
- ◆ The step-size variable  $\delta$  (Appendix) affects the accuracy of **dy/dx**, **INFLC** in **dxNDer** differentiation mode (Chapter 1), **ARC**, and **TANLN**. Accuracy increases as the step-size value becomes smaller.
- ◆ The differentiation mode setting affects **dy/dx**, **INFLC**, **ARC**, and **TANLN**; **dxDer1** (exact) mode is more accurate than **dxNDer** (numeric) mode (Chapter 1).

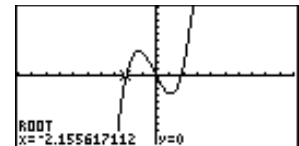
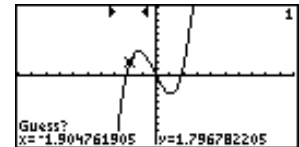
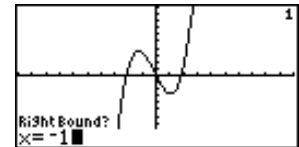
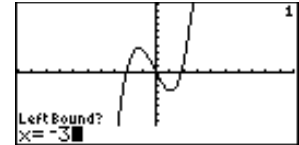
### Using ROOT, FMIN, FMAX, or INFLC

The steps for **ROOT**, **FMIN**, **FMAX**, and **INFLC** are the same, except for the menu selection in step 1.

In the example, the function  $y(x)=x^3+3x^2-4x$  is selected. Step 2 is not necessary here because only one function is selected.

When you enter a value directly for the left bound, right bound, or guess, an  $x=$  prompt is displayed on the bottom of the graph screen.

- 1 Select **ROOT** from the GRAPH MATH menu. A **Left Bound?** prompt is displayed.
  - [GRAPH] [MORE]
  - [F1] [F1]
  - ▼ ▲
- 2 Move the cursor onto the function for which you want to find a root.
  - [←] 3 [ENTER] (or [←] [▶] [ENTER])
- 3 Specify the left bound for  $x$ . Either move the trace cursor to the left bound or enter a value directly. **Right Bound?** is displayed.
  - [←] 1 [ENTER] (or [←] [▶] [ENTER])
- 4 Specify the right bound for  $x$  as in step 3. **Guess?** is displayed.
  - [←] [▶] (or [←] 2)
- 5 Guess an  $x$  value near the root between the left bound and the right bound. Either move the cursor or enter a value.
  - [ENTER]
- 6 Solve for  $x$ . The result cursor is displayed at the solution point, the cursor coordinate values are displayed, and the  $x$  value is stored in **Ans**.



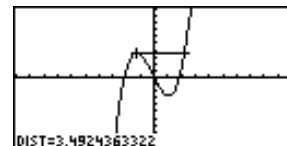
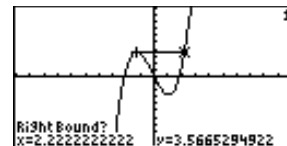
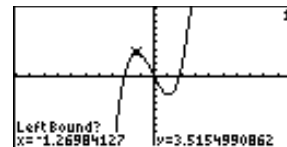
### Using $\int f(x)$ , DIST, or ARC

The steps for using  $\int f(x)$ , **DIST**, and **ARC** are the same, except for the menu selection in step 1.

In the example, the function  $y(x)=x^3+3x^2-4x$  is selected. Steps 2 and 4 are not necessary here because only one function is selected.

For **DIST**, when you are specifying the right bound, a line is drawn from the left bound to the right bound.

- 1 Select **DIST** from the GRAPH MATH menu. The current graph is displayed with a **Left Bound?** prompt.
  - 
  -
- 2 Move the cursor onto the function on which the left bound is a point.
  -
- 3 Select the left bound for  $x$ . Either move the cursor to the left bound or enter the  $x$  value. **Right Bound?** is displayed.
  - or *value*
- 4 (**DIST** only) If you want the right bound to be a point on another function, move the cursor to the other function.
  -
- 5 Select the right bound. Either move the cursor to the right bound or enter its  $x$  value.
  - or *value*
  -
- 6 Solve.
  - ◆ For **DIST**, the solution **DIST=** is displayed and stored in **Ans**.
  - ◆ For **ARC**, the solution **ARC=** is displayed and stored in **Ans**.
  - ◆ For  $\int f(x)$ , the solution  $\int f(x)=$  is displayed, shaded, and stored in **Ans**. The function integral error value is stored to the variable **fnIntErr** (Appendix). To remove the shading, select **CLDRW** from the GRAPH DRAW menu (page 103).



In the example, the function  $y(x)=x^3+3x^2-4x$  is selected.

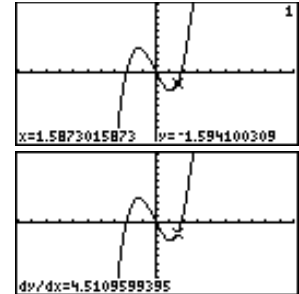
**TANLN** (GRAPH MATH menu) and **TanLn** (GRAPH DRAW menu) both draw a tangent line on the graph; only **TANLN** displays the solution,  $dy/dx$ .

### Using $dy/dx$ or TANLN

The steps for using  $dy/dx$  and **TANLN** are the same, except for the menu selection in step 1.

- ❶ Select  $dy/dx$  from the GRAPH MATH menu. The current graph is displayed.
  - ❷ Move the cursor to the function with the point for which you want to find the derivative, or slope.
  - ❸ Move the cursor to the point (or enter the  $x$  value).
  - ❹ Solve.
    - ◆ For  $dy/dx$ , the solution  $dy/dx=$  is displayed and stored in **Ans**.
    - ◆ For **TANLN**, a tangent line also is displayed.
- To remove the tangent line and  $dy/dx=$  prompt, select **CLDRW** from the GRAPH DRAW menu.

GRAPH MORE  
 F1 F2  
 ▾ ▲  
 ◀ ▶  
 ENTER



In the example, the functions  $y(x)=x^3+3x^2-4x$  and  $y(x)=x^2+3x-3$  are selected.

### Using ISECT

- 1 Select **ISECT** from the GRAPH MATH menu. The current graph is displayed with **First Curve?** at the bottom of the graph screen.
- 2 Select the first function (curve). The cursor moves to the next function and **Second Curve?** is displayed.
- 3 Select the second function (curve). **Guess?** is displayed.
- 4 Guess the intersection. Either move the cursor to a point near an intersection or enter an **x** value.
- 5 Solve. The result cursor is displayed at the intersection, the cursor coordinates are the result, and the **x** value is stored to **Ans**.

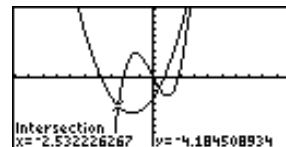
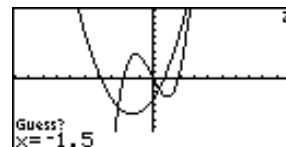
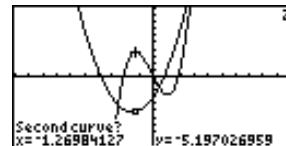
[GRAPH] [MORE]  
[F1] [MORE] [F3]

▼ ▲ [ENTER]

▼ ▲ [ENTER]

[◀] 1 [.] 5  
(or [←] [→])

[ENTER]



### Using YICPT

To use **YICPT**, select **YICPT** from the GRAPH MATH menu ([GRAPH] [MORE] [F1] [MORE] [F2]). Press ▼ and ▲ to select a function, and then press [ENTER]. The result cursor is displayed at the y-intercept, the cursor coordinate values are displayed, and **y** is stored in **Ans**.



To clear entered numbers from the **Eval x=** prompt, press **[CLEAR]**.

To cancel **EVAL**, press **[CLEAR]** after clearing the **Eval x=** prompt.

Expressions are valid for **x**.

You may continue to enter valid **x** values for which to evaluate the selected functions.

## Evaluating a Function for a Specified **x**

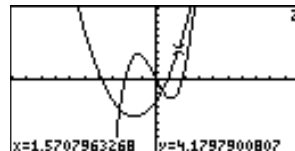
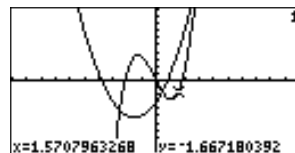
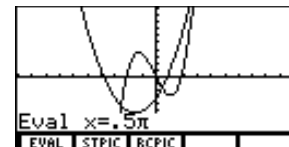
- 1 Select **EVAL** from the **GRAPH** menu. The graph is displayed with the **Eval x=** prompt in the bottom-left corner.
- 2 Enter a real **x** value between window variables **xMin** and **xMax**.
- 3 Evaluate. The result cursor is on the first selected function at the entered **x** value. The coordinate values are displayed. The number in the top-right corner indicates which function is evaluated.
- 4 Move the result cursor to the next or previous selected function. The result cursor is on the next or previous function at entered **x** value, the coordinate values are displayed, and the function number changes.

**[GRAPH]** **[MORE]**  
**[MORE]** **[F1]**

**[.]** **5** **[2nd]** **[π]**

**[ENTER]**

**[↑]** **[↓]**



## Drawing on a Graph

You can use the drawing tools (except **DrInv**) to draw points, lines, circles, shaded areas, and text on the current graph in any graphing mode. The drawing tools use the display's **x**- and **y**-coordinate values.

### Before Drawing on a Graph

All drawings are temporary; they are not stored in a graph database. Any action that causes Smart Graph to replot the graph erases all drawings. Therefore, before you use any drawing tool, consider whether you want to perform any of these graphing activities first.

- ◆ Change a mode setting that affects graphs
- ◆ Select, deselect, or edit a current function or stat plot
- ◆ Change the value of a variable used in a selected function
- ◆ Change a window variable value
- ◆ Change a graph format setting or graph style
- ◆ Clear current drawings with **CLDRW**

### Saving and Recalling Drawn Pictures

*Graph database (GDB) and picture (PIC) variable names can be from one to eight characters long. The first character must be a letter.*

To store the elements that define the current graph to a graph database (**GDB**) variable, select **STGDB** from the GRAPH menu. These information types are stored to a **GDB** variable:

- ◆ Equation editor functions
- ◆ Window variable values
- ◆ Graph style settings
- ◆ Format settings

To recall the stored **GDB** later, select **RCGDB** from the GRAPH menu, and then select the **GDB** variable from the GRAPH RCGDB menu. When you recall a **GDB**, the information stored in the **GDB** replaces any current information of these types.

*The next section describes how to draw lines, points, curves, and text onto a graph; you then can store the drawings to a **PIC** variable.*

To store the current graph display, including drawings, to a picture (**PIC**) variable, select **STPIC** from the GRAPH menu. Only the graph picture is stored to the specified **PIC** variable.

To superimpose one or more stored graph pictures onto a graph later, select **RPCIC** from the GRAPH menu, and then select the **PIC** variable from the GRAPH RPCIC menu.

## Clearing Drawn Pictures

To clear drawn pictures while the graph is displayed, select **CLDRW** from the GRAPH DRAW menu. The graph is replotted and displayed with no drawn elements.

To clear drawn pictures from the home screen, select **CIDrw** from the CATALOG. **CIDrw** is pasted to the cursor location. Press **[ENTER]**. **Done** is displayed; when you display the graph again, no drawings are displayed.

### The GRAPH DRAW Menu

GRAPH MORE F2

MATH	DRAW	FORMT	STGDB	RCGDB
Shade	LINE	VERT	HORIZ	CIRCL
▶				
DrawF	PEN	PTON	PTOFF	PTCHG
▶				
CLDRW	PxOn	PxOff	PxChg	PxTest
▶				
TEXT	TanLn	DrInV		

*DrInV is not available in Pol, Param, or DifEq graphing modes.*

You can use these GRAPH DRAW menu items only on the home screen or in the program editor.

<b>Shade</b> (	Shades a specified area of a graph (See page 104)
<b>DrawF</b> <i>expression</i>	Draws <i>expression</i> as a function
<b>PxOn</b> ( <i>row,column</i> )	Turns on the pixel at ( <i>row,column</i> )
<b>PxOff</b> ( <i>row,column</i> )	Turns off the pixel at ( <i>row,column</i> )
<b>PxChg</b> ( <i>row,column</i> )	Changes the on/off status of the pixel at ( <i>row,column</i> )
<b>PxTest</b> ( <i>row,column</i> )	Returns <b>1</b> if the pixel at ( <i>row,column</i> ) is on, or <b>0</b> if the pixel is off
<b>TanLn</b> ( <i>expression,x</i> )	Draws <i>expression</i> as a function and a tangent line of <i>expression</i> at <i>x</i>
<b>DrInV</b> <i>expression</i>	Draws the inverse of <i>expression</i>

*For PxOn, PxOff, PxChg, and PxTest, row and column are integers, where  $0 \leq \text{row} \leq 62$  and  $0 \leq \text{column} \leq 126$ .*

*For DrawF, TanLn, and DrInV, expression is in terms of  $x$ . Also, you cannot include a list in expression to draw a family of curves.*

## Shading Areas of a Graph

To shade an area of a graph, the syntax is:

**Shade**(*lowerFunc*,*upperFunc*[,*xLeft*,*xRight*,*pattern*,*patternRes*])

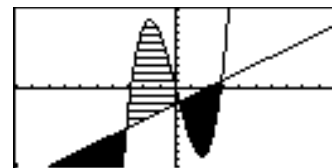
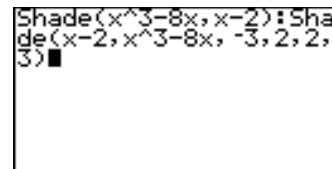
To replicate the example without additional graphs, turn off all equations and stat plots before entering the instructions as shown.

*pattern* specifies one of four shading patterns.

- |   |                      |
|---|----------------------|
| 1 | vertical (default)   |
| 2 | horizontal           |
| 3 | negative slope (45°) |
| 4 | positive slope (45°) |

*patternRes* specifies one of eight shading resolutions.

- |   |                       |
|---|-----------------------|
| 1 | every pixel (default) |
| 2 | every second pixel    |
| 3 | every third pixel     |
| 4 | every fourth pixel    |
| 5 | every fifth pixel     |
| 6 | every sixth pixel     |
| 7 | every seventh pixel   |
| 8 | every eighth pixel    |



- ◆ The area that is specifically above *lowerFunc* and below *upperFunc* is shaded.
- ◆  $xLeft > xMin$  and  $xRight < xMax$  must be true.
- ◆  $xLeft$  and  $xRight$  specify left and right bounds for shading. ( $xMin$  and  $xMax$  are defaults.)

These GRAPH DRAW menu items are interactive. Also, you can use all of them, except **PEN**, on the home screen or in a program (A to Z Reference).

- LINE** Draws a line segment from one point to another point you specify with the cursor
- VERT** Draws a vertical line, which you can move to any displayed  $x$  value

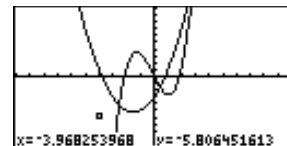
- HORIZ** Draws a horizontal line, which you can move to any displayed **y** value
- CIRCL** Draws a circle with a center point and radius you specify with the cursor
- PEN** Draws the path of the cursor as you move it on the graph screen
- PTON** Turns on the point at the cursor location
- PTOFF** Turns off the point at the cursor location
- PTCHG** Changes the on/off status of a point at the cursor location
- CLDRW** Clears all drawings from the graph screen; replots the graph
- TEXT** Draws characters on the graph at the cursor location

### Drawing a Line Segment

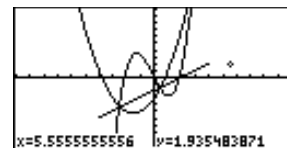
*In the example, the functions  $y(x)=x^3+3x^2-4x$  and  $y(x)=x^2+3x-3$  are selected.*

- 1 Select **LINE** from the GRAPH DRAW menu. The graph is displayed.
- 2 Define one segment endpoint with the cursor.
- 3 Define the other endpoint of the segment. As you move the cursor, a line anchored at the first defined endpoint extends to the cursor.
- 4 Draw the line.

[GRAPH] [MORE]  
[F2] [F2]  
[▶] [▼] [◀] [▲]  
[ENTER]



[▶] [▼] [◀] [▲]  
[ENTER]



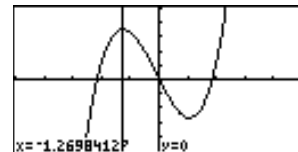
To draw more line segments, repeat steps 2 and 3; to cancel **LINE**, press [CLEAR].

In the example, the function  $y(x)=x^3+.3x^2-4x$  is selected. Also, **ZIN** was executed once with the zoom cursor at  $(0,0)$ , **xFact=2**, and **yFact=2**.

### Drawing a Vertical or Horizontal Line

- 1 Select **VERT** (or **HORIZ**) from the GRAPH DRAW menu. The graph is displayed and a vertical or horizontal line is drawn at the cursor.
- 2 Move the line to the **x** value (or to the **y** value, if horizontal) through which you want the line to pass.
- 3 Draw the line on the graph.

[GRAPH] [MORE]  
 [F2] [F3]  
 (or [F4])  
 [◀] [▶]  
 (or [▲] [▼])  
 [ENTER]



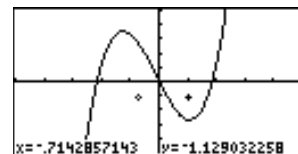
To draw more lines, repeat steps 2 and 3; to cancel **VERT** or **HORIZ**, press [CLEAR].

### Drawing a Circle

In the example, the function  $y(x)=x^3+.3x^2-4x$  is selected. Also, **ZIN** was executed once with the zoom cursor at  $(0,0)$ , **xFact=2**, and **yFact=2**.

- 1 Select **CIRCL** from the GRAPH DRAW menu. The graph is displayed.
- 2 Define the center point of the circle with the cursor.
- 3 Move the cursor to any point on the intended circumference.
- 4 Draw the circle.

[GRAPH] [MORE] [F2]  
 [F5]  
 [▶] [▼] [◀] [▲]  
 [ENTER]  
 [▶] [▼] [◀] [▲]  
 [ENTER]



Here the circle appears as a circle, regardless of window variable values. When you use **Circl**( from the CATALOG to draw a circle, the current window variable values may distort the shape.

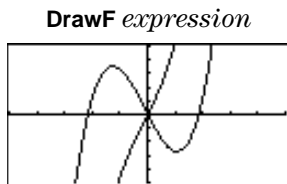
To draw more circles, repeat steps 2 through 4; to cancel **CIRCL**, press [CLEAR].

For **DrawF**, **TanLn**, and **DrInV**, you can use as *expression* any variable to which a valid expression is stored (including deselected equation variables).

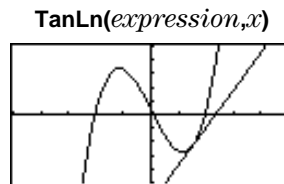
In the illustrations,  $y1=x^3+3x^2-4x$  is selected.

## Drawing a Function, Tangent Line, or Inverse Function

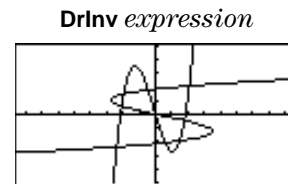
For **DrawF**, **TanLn**, and **DrInV**, *expression* is in terms of  $x$ . When you select **DrawF**, **TanLn**, or **DrInV** from the GRAPH DRAW menu, it is pasted to the home screen or program editor. Upon execution, the drawing is returned. **DrInV** draws the inverse of *expression* by plotting its  $x$  values on the  $y$ -axis and its  $y$  values on the  $x$ -axis. **DrInV** is available only in **Func** graphing mode.



**DrawF**  $x^3+3x^2+4x$



**TanLn**( $y1,1.5$ )



**DrInV**  $y1$

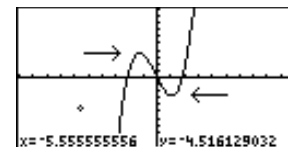
In the example, the function  $y(x)=x^3+3x^2-4x$  is selected. Also, **ZSTD** was executed.

To draw a diagonal line or curve, turn on the pen, press **ENTER** **ENTER**, press  $\leftarrow$   $\uparrow$  (or  $\downarrow$   $\rightarrow$ ), and so on), and repeat.

## Drawing Freehand Points, Lines, and Curves

- 1 Select **PEN** from the GRAPH DRAW menu.
- 2 Move the cursor to where you want to begin drawing.
- 3 Turn on the pen.
- 4 Draw whatever you want.
- 5 Turn off the pen.

**GRAPH** **MORE** **F2**  
**MORE** **F2**  
 $\rightarrow$   $\downarrow$   $\leftarrow$   $\uparrow$   
**ENTER**  
 $\rightarrow$   $\downarrow$   $\leftarrow$   $\uparrow$   
**ENTER**



To draw more points, lines, or curves, repeat steps 2 through 5. To cancel, press **CLEAR**.

This example adds to the **PEN** example drawing. Before you start, you may want to store the arrows to a picture variable (page 102).

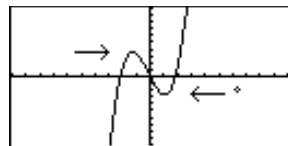
To erase a character when using **TEXT**, move the **TEXT** cursor above it and then press **[ALPHA]** **[\_]** or **[2nd]** **[alpha]** **[\_]** to overwrite it.

In the example, the function  $y(x)=x^3+3x^2-4x$  is selected. Also, **ZSTD** was executed. Points are turned on at  $(-5,5)$ ,  $(5,-5)$ , and  $(-5,-5)$ .

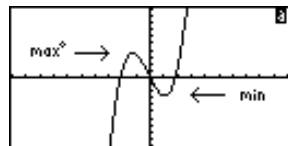
### Placing Text on a Graph

- 1 Select **TEXT** from the GRAPH DRAW menu. The text cursor is displayed.
- 2 Move the cursor to where you want to enter text. Text is entered below the text cursor.
- 3 Set alpha-lock and enter **min**. (The alpha cursor (**Ⓐ**) is displayed in the top-right corner.
- 4 Move the cursor to another location.
- 5 Enter **max** (alpha-lock remains on).

**[GRAPH]** **[MORE]** **[F2]**  
**[MORE]** **[MORE]** **[MORE]**  
**[F1]**  
**[▶]** **[▼]** **[◀]** **[▲]**



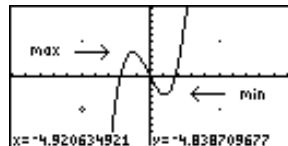
**[2nd]** **[alpha]** **[ALPHA]**  
**[M]** **[I]** **[N]**  
**[▶]** **[▼]** **[◀]** **[▲]**  
**[M]** **[A]** **[X]**



### Turning On or Turning Off Points

- 1 Select **PTON** (or **PTOFF**) from the GRAPH DRAW menu.
- 2 Move the cursor to where you want to draw (or erase) a point.
- 3 Turn on (or turn off) the point.

**[GRAPH]** **[MORE]** **[F2]**  
**[MORE]** **[F3]**  
**[▶]** **[▼]** **[◀]** **[▲]**



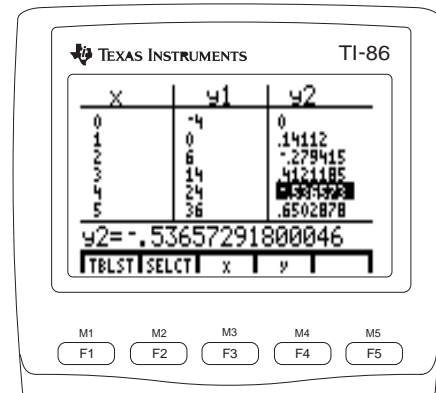
**[ENTER]**

To continue drawing points, repeat steps 2 and 3. To cancel **PTON**, press **[CLEAR]**.



# 7 Tables

Displaying the Table ..... 110  
Setting Up the Table ..... 113  
Clearing the Table..... 114



## Displaying the Table

To display the equation editor, press **GRAPH** **F1** (Chapter 5).

The table displays the independent values and corresponding dependent values for up to 99 selected functions in the equation editor. Each dependent variable in the table represents a selected function stored in the equation editor for the current graphing mode.

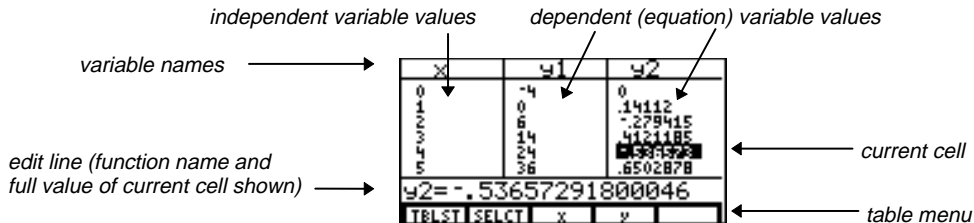
### TABLE Menu **TABLE**



### The Table **TABLE** **F1**

In the example,  $y_1=x^2+3x-4$  and  $y_2=\sin(3x)$  are selected and all defaults set.

The table abbreviates values in the columns, if necessary.



*In DifEq mode, if an equation has an initial conditions list, the table uses the first list element to evaluate the equation (Chapter 10).*

## Independent and Dependent Variables in the Table

Graphing Mode	Independent Variable	Dependent (Equation) Variables
<b>Func</b> (function)	<b>x</b>	<b>y1</b> through <b>y99</b>
<b>Pol</b> (polar)	$\theta$	<b>r1</b> through <b>r99</b>
<b>Param</b> (parametric)	<b>t</b>	<b>xt1/yt1</b> through <b>xt99/yt99</b>
<b>DifEq</b> (differential equation)	<b>t</b>	<b>Q1</b> through <b>Q9</b>

## Navigating the Table

To...	Do this:
Display more dependent variables in the table	Press $\rightarrow$ or $\leftarrow$
Display greater values in any column	Press $\downarrow$ (only when <b>Indpnt: Auto</b> is set; page 112)
Set <b>TblStart</b> to a lower value	Press $\uparrow$ in the independent variable column until the cursor moves past the current <b>TblStart</b> (page 112)
Display the equation in the edit line, where you can edit or deselect it	Press $\leftarrow$ or $\rightarrow$ to move the cursor to an equation variable column, hold $\uparrow$ until the cursor highlights the equation name, and then press <b>[ENTER]</b> ; the equation is displayed in the edit line

**The Table Menus** TABLE F1

The table has a unique menu for each graphing mode, as shown below.

*In Function Graphing Mode*

<b>TBLST</b>	<b>SELCT</b>	<b>x</b>	<b>y</b>	
--------------	--------------	----------	----------	--

*In Polar Graphing Mode*

<b>TBLST</b>	<b>SELCT</b>	$\theta$	<b>r</b>	
--------------	--------------	----------	----------	--

*In Parametric Graphing Mode*

<b>TBLST</b>	<b>SELCT</b>	<b>t</b>	<b>xt</b>	<b>yt</b>
--------------	--------------	----------	-----------	-----------

*In Differential Equation Graphing Mode*

<b>TBLST</b>	<b>SELCT</b>	<b>t</b>	<b>Q</b>	
--------------	--------------	----------	----------	--

**TBLST**

Displays the table setup editor

**SELCT**

On the edit line, deselects or cancels deselection of the equation

**x** and **y**;  $\theta$  and **r**; **t**, **xt**,  
and **yt**; or **t** and **Q**

On the edit line, pastes the variable to the cursor location; the variables change according to graphing mode

- ◆ To add an equation to the table, select it in the equation editor (Chapter 5). **SELCT** only removes equations from the table.
- ◆ To remove an equation from a column in the table, select **SELCT** from the table menu. Remaining equations that follow the removed equation shift left one column.
- ◆ To deselect an equation with **SELCT**, the equation and cursor must be displayed in the edit line. If the equation is in the edit line but the cursor is not, press ENTER.
- ◆ To compare two dependent variables not defined consecutively in the equation editor, use **SELCT** from the table screen menu to deselect the dependent variables in between.

## Setting Up the Table

To display the table using the current table setup settings, select **TABLE** from the **TABLE** menu.

**TblStart** and  $\Delta\text{Tbl}$  must be real numbers; you can enter an expression.

In **DifEq** graphing mode, it is a good practice to set **TblStart** = **tMin** and  $\Delta\text{Tbl}$  = **tStep**.

To display the table setup editor, select **TBLST** from the **TABLE** menu. The screen to the right shows the default table setup settings.

**TblStart** specifies the first independent variable value (**x**,  $\theta$ , or **t**) in the table (only when **Indpnt: Auto** is selected).

$\Delta\text{Tbl}$  (table step) specifies the increment or decrement from one independent variable value to the next independent variable value in the table.

- ◆ If  $\Delta\text{Tbl}$  is positive, then the values of **x**,  $\theta$ , or **t** increase as you scroll down the table.
- ◆ If  $\Delta\text{Tbl}$  is negative, then the values of **x**,  $\theta$ , or **t** decrease as you scroll down the table.

**Indpnt: Auto** displays independent variable values automatically in the first column of the table, starting at **TblStart**.

**Indpnt: Ask** displays an empty table. As you enter **x** values in the **x=** prompt (**x=value** **[ENTER]**), each value is added to the independent variable column and the corresponding dependent variable values are calculated and displayed. When **Ask** is set, you cannot scroll beyond the six independent variable values that are currently displayed in the table.



In the example,  $y1=x^2+3x-4$  and  $y2=\sin(3x)$  are selected and all defaults set.

When you display the equation in the edit line, the column equation name is highlighted.

## Viewing and Editing Dependent Variable Equations

- 1 Display the table. TABLE [F1]
- 2 Move the cursor into the column of the dependent variable you want to edit, and then move up the column until the name is highlighted. [ ] [ ]
- 3 Display the equation in the edit line. [ENTER]
- 4 Edit the equation. [ ] [ ] [ ] 5 [ ]  
+ 1
- 5 Enter the edited equation. [ENTER]
  - ◆ The dependent variable values are recalculated.
  - ◆ The cursor returns to the edited dependent variable's first value.
  - ◆ The equation editor is updated.

x	y1	y2
0	-4	0
1	0	.14112
2	6	-.279415
3	14	.4121185
4	24	-.536573
5	36	.6502878

y1 x<sup>2</sup>+3x-4

TBLST	SELECT	x	y
-------	--------	---	---

x	y1	y2
0	0	0
1	7	.14112
2	15	-.279415
3	25	.4121185
4	37	-.536573
5	51	.6502878

y1=1

TBLST	SELECT	x	y
-------	--------	---	---

## Clearing the Table

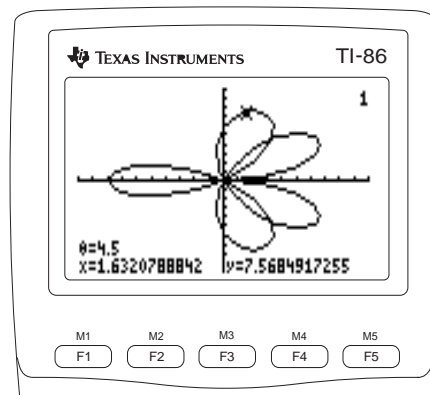
When you use **CITbl** in a program, the table is cleared upon program execution (Chapter 16).

To clear the table when **Indpnt: Ask** is set, select **CITbl** from the CATALOG, and then press **[ENTER]**. All independent and dependent variable columns are cleared. **CITbl** does nothing when **Indpnt: Auto** is set.

# 8

## Polar Graphing

Preview: Polar Graphing .....	116
Defining a Polar Graph .....	117
Using Graph Tools in Pol Graphing Mode.....	119



## Preview: Polar Graphing

The graph of the polar equation  $A \sin(B\theta)$  forms the shape of a flower. Graph the flower for  $A=8$  and  $B=2.5$ . Then explore the appearance of the flower for other values of  $A$  and  $B$ .

- 1 Select **Pol** mode from the mode screen.

2nd [MODE]  $\downarrow$   $\downarrow$   $\downarrow$   
 $\downarrow$   $\rightarrow$  ENTER

```
Normal Sci Eng
Float 012345678901
Radian Degree
RectD PolarC
Func Pol Param Dife4
```

- 2 Display the equation editor and polar equation editor menu.

GRAPH [F1]

```
Plot1 Plot2 Plot3
r1=8 sin (2.5 θ)■
```

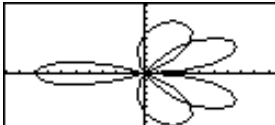
Y(0)= WIND ZOOM TRACE GRAPH  
 # P INSP RELF SELCT

- 3 (Deselect or delete all equations if any.)  
 Store  $r_1(\theta)=8\sin(2.5\theta)$ .

( [MORE] [F2] [MORE] )  
 8 [SIN] [ ] 2 [.] 5 [F1] [ ]

- 4 Select **ZSTD** from the GRAPH ZOOM menu.  $r_1$  is plotted on the graph screen.

2nd [M3] [F4]



r(θ)= WIND ZOOM TRACE GRAPH

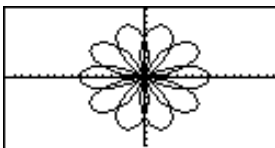
- 5 Display the window editor, and then change  $\theta_{\text{Max}}$  to  $4\pi$ .

[F2]  
 $\downarrow$  4 2nd [π]

```
WINDOW
θMin=0
θMax=4π■
θStep=.130899693899...
```

- 6 Select **ZSQR** from the GRAPH ZOOM menu.  $x_{\text{Min}}$  and  $x_{\text{Max}}$  are changed to display the graph in correct proportion.

[F3] [MORE] [F2]



- 7 Change the values of  $A$  and  $B$  and redisplay the graph.

[F1] (enter other  $A$   
 and  $B$  values)

To remove the GRAPH menu from the graph screen, as shown, press [CLEAR].

To redisplay the GRAPH menu, press [GRAPH].



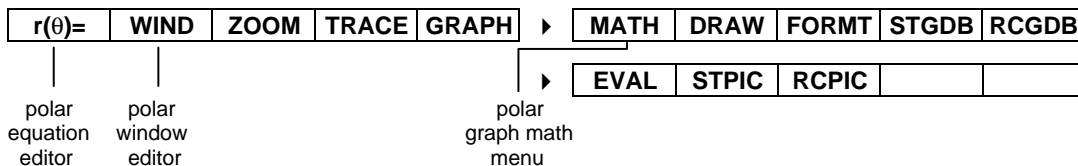
## Defining a Polar Graph

The steps for defining a polar graph are similar to the steps for defining a function graph. This chapter assumes that you are familiar with Chapter 5: Function Graphing and Chapter 6: Graph Tools. Chapter 8 details aspects of polar graphing that differ from function graphing.

### Setting Polar Graphing Mode

To display the mode screen, press  $\boxed{2\text{nd}} \boxed{[\text{MODE}]}$ . To graph polar equations, you must select **Pol** graphing mode before you enter equations, set the format, or edit window variable values. The TI-86 retains separate equation, format, and window data for each graphing mode.

### The GRAPH Menu $\boxed{[\text{GRAPH}]}$



Chapter 5 describes these GRAPH menu items: **GRAPH** and **FORMT**.

Chapter 6 describes these GRAPH menu items: **ZOOM**, **TRACE**, **DRAW**, **STGDB**, **RCGDB**, **EVAL**, **STPIC**, and **RCPIC**.

### Displaying the Polar Equation Editor

To display the polar equation editor, select  $r(\theta)=$  from the GRAPH menu in **Pol** graphing mode ( $\text{GRAPH}$   $\text{F1}$ ). The polar equation editor menu displayed on the bottom line is the same as the **Func** mode equation editor menu, except that  $\theta$  and  $r$  replace  $x$  and  $y$ .

In this editor, you can enter and display up to 99 polar equations,  $r1$  through  $r99$ , if sufficient memory is available. Equations are defined in terms of the independent variable  $\theta$ .

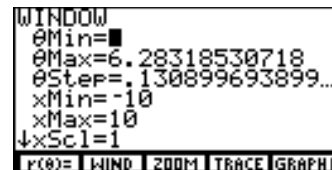
The default graph style is  $\setminus$  (line) in **Pol** graphing mode.  $\nabla$  (shade above) and  $\blacktriangledown$  (shade below) graph styles are not available in **Pol** graphing mode.



### Setting the Graph Screen Window Variables

To display the polar window editor, select **WIND** from the GRAPH menu ( $\text{GRAPH}$   $\text{F2}$ ). **Pol** graphing mode has the same window variables as **Func** graphing mode, except:

- ◆  $xRes$  is not available in **Pol** graphing mode.
- ◆  $\theta Min$ ,  $\theta Max$ , and  $\theta Step$  are available in **Pol** graphing mode.



The values shown in the picture to the right are the defaults in **Radian** mode.  $\downarrow$  indicates that  $yMin=-10$ ,  $yMax=10$ , and  $yScl=1$  are beyond the screen.

- $\theta Min=0$  Specifies the first  $\theta$  value to evaluate within the graph screen
- $\theta Max=6.28318530718$  Specifies the last  $\theta$  value to evaluate within the graph screen
- $\theta Step=.13089969389957$  Specifies the increment from one  $\theta$  value to the next  $\theta$  value

$\theta Max$  default is  $2\pi$ .

$\theta Step$  default is  $\pi/24$ .

**DrawLine** graph format typically displays a more meaningful polar graph than **DrawDot** graph format.

## Setting the Graph Format

To display the format screen in **Pol** graphing mode, select **FORMT** from the GRAPH menu (**GRAPH** **MORE** **F3**). Chapter 5 describes the format settings. Although the same settings are available for **Func**, **Pol**, and **Param** graphing modes, the TI-86 retains in memory separate format settings for each mode. In **Pol** graphing mode, **PolarGC** shows the cursor coordinates in terms of  $r$  and  $\theta$ , the variables that define the equations.

## Displaying the Graph

To plot the selected polar equations, you can select **GRAPH**, **TRACE**, **EVAL**, **RCGDB**, or a **ZOOM**, **MATH**, **DRAW**, or **RCPIC** operation, from the GRAPH menu. The TI-86 evaluates  $r$  for each value of  $\theta$  (from  $\theta\text{Min}$  to  $\theta\text{Max}$  in intervals of  $\theta\text{Step}$ ) and then plots each point. As the graph is plotted, the variables  $\theta$ ,  $r$ ,  $x$ , and  $y$  are updated.

# Using Graph Tools in Pol Graphing Mode

## The Free-Moving Cursor

The free-moving cursor in **Pol** graphing works the same as in **Func** graphing.

- ◆ In **RectGC** format, moving the cursor updates the values of  $x$  and  $y$ ; if **CoordOn** format is selected,  $x$  and  $y$  are displayed.
- ◆ In **PolarGC** format, moving the cursor updates  $x$ ,  $y$ ,  $r$ , and  $\theta$ ; if **CoordOn** format is selected,  $r$  and  $\theta$  are displayed.

### Tracing a Polar Equation

To begin a trace, select **TRACE** from the GRAPH menu (press  $\boxed{\text{GRAPH}}$   $\boxed{\text{F4}}$ ). The trace cursor appears on the first selected equation at  $\theta\text{Min}$ .

- ◆ In **RectGC** format, moving the trace cursor updates the values of  $\theta$ ,  $x$ , and  $y$ ; if **CoordOn** format is selected,  $\theta$ ,  $x$ , and  $y$  are displayed.
- ◆ In **PolarGC** format, moving the trace cursor updates  $x$ ,  $y$ ,  $r$ , and  $\theta$ ; if **CoordOn** format is selected,  $r$  and  $\theta$  are displayed.

To move the trace cursor...

Press:

Along the graph of the equation by increments or decrements of  $\theta\text{Step}$

$\boxed{\blacktriangleright}$  or  $\boxed{\blacktriangleleft}$

From one equation to another

$\boxed{\blacktriangledown}$  or  $\boxed{\blacktriangleup}$

*QuickZoom is available in Pol graphing; panning is not (Chapter 6).*

If you move the trace cursor beyond the top or bottom of the graph screen, the coordinate values at the bottom of the screen continue to change appropriately.

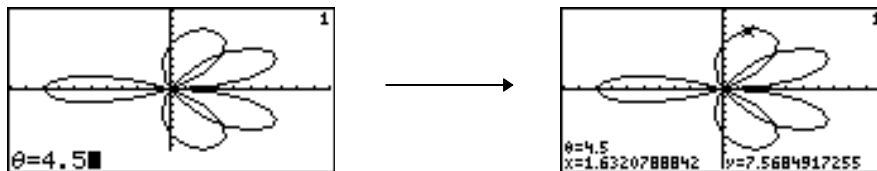
If you have graphed a family of curves,  $\boxed{\blacktriangledown}$  and  $\boxed{\blacktriangleup}$  move through each curve before moving to the next polar equation.

### Moving the Trace Cursor to a $\theta$ Value

To move the trace cursor to any valid  $\theta$  value on the current equation, enter the number. When you enter the first digit, a  $\theta=$  prompt is displayed in the bottom-left corner. The value you enter must be valid for the current graph screen. When you have completed the entry, press **[ENTER]** to reactivate the trace cursor.

*In the example,  $r1=8\sin(2.5\theta)$  is graphed.*

*Values for  $\theta$ ,  $x$ , and  $y$  are displayed on the graph to the right because **RectGC** graph format is selected.*



### Using Zoom Operations

The GRAPH ZOOM menu items, except **ZFIT**, work the same in **Pol** graphing as in **Func** graphing. In **Pol** graphing mode, **ZFIT** adjusts the graph screen in both the  $x$  and  $y$  directions.

The zoom operations affect only the  $x$  window variables (**xMin**, **xMax**, and **XscI**) and the  $y$  window variables (**yMin**, **yMax**, and **yScI**), except **ZSTO** and **ZRCL**, which also affect the  $\theta$  window variables ( **$\theta$ Min**,  **$\theta$ Max**, and  **$\theta$ Step**).

### The GRAPH MATH Menu

GRAPH MORE F1

MATH	DRAW	FORMT	STGDB	RCGDB
DIST	dy/dx	dr/dθ	ARC	TANLN

*The other GRAPH MATH menu items are the same as described in Chapter 6.*

**dr/dθ** Finds the numerical derivative (slope) of a function at a point

The distances calculated by **DIST** and **ARC** are distances in the rectangular coordinate plane. **dy/dx** and **dr/dθ** are independent of the **RectGC** or **PolarGC** format.

At a point where the derivative is undefined, **TANLN** will draw the line, but no result is displayed or stored in **Ans**.

### Evaluating an Equation for a Specified $\theta$

When the trace cursor is not active, the GRAPH menu item **Eval** evaluates selected polar equations directly on the graph for a given value of  $\theta$ . **eval** in a program or from the home screen returns a list of  $r$  values.

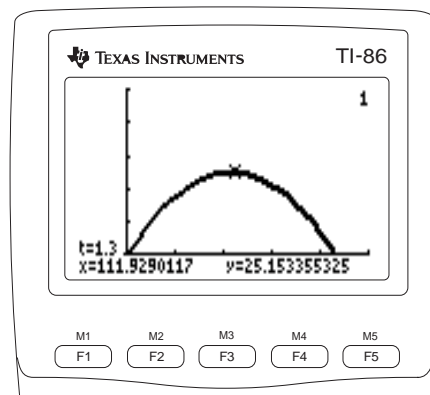
### Drawing on a Polar Graph

The GRAPH DRAW menu items work the same in **Pol** graphing as in **Func** graphing. DRAW instruction coordinates in **Pol** graphing mode are the  $x$ - and  $y$ -coordinates of the graph screen. **DrInV** is not available in **Pol** graphing mode.

# 9


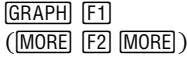
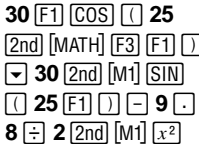
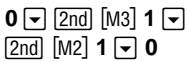
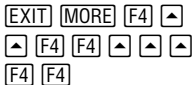
# Parametric Graphing

Preview: Parametric Graphing ..... 124  
Defining a Parametric Graph ..... 125  
Using Graph Tools in Param Graphing Mode ..... 128



## Preview: Parametric Graphing

Graph the parametric equation that describes the path of a ball kicked at an initial speed of 30 meters per second, at an initial angle of 25 degrees with the horizontal (from ground level). How far does the ball travel? When does it hit the ground? How high does it go?

- 1 Select **Param** mode from the mode screen. 
- 2 Display the equation editor and parametric equation editor menu. Deselect all equations and plots (if any are defined). 
- 3 Define the path of the ball as **xt1** and **yt1** in terms of **t**.  
Horizontal:  $xt1 = tv_0 \cos(\theta)$   
Vertical:  $yt1 = tv_0 \sin(\theta) - 1/2(gt)^2$   
Gravity constant:  $g = 9.8 \text{ m/sec}^2$  
- 4 Define the vertical component vector as **xt2** and **yt2** and define the horizontal component vector as **xt3** and **yt3**. 
- 5 Change the graph style of **xt3/yt3** to  $\overline{\hspace{1cm}}$  (thick). Change the graph style of **xt2/yt2** and **xt1/yt1** to  $\overline{\hspace{1cm}}$  (path). 

```
Normal Sci Eng
Float 012345678901
Radian Degree
RectPol PolarC
Func Pol Param DifEq
```

```
Plot1 Plot2 Plot3
\xt1=30t cos (25°)
\yt1=30t sin (25°)-9...
\xt2=
\yt2=
```

t	xt	yt	DELf	SELCt
0	F	I	FORMS	

```
Plot1 Plot2 Plot3
\xt1=30t sin (25°)-9...
\xt2=0
\yt2=0t1
\xt3=xt1
\yt3=0
```

t	xt	yt	DELf	SELCt
0	F	I	FORMS	

```
Plot1 Plot2 Plot3
\xt1=30t cos (25°)
\yt1=30t sin (25°)-9...
\xt2=0
\yt2=0t1
\xt3=xt1
```

MODE	WIND	ZOOM	TRACE	GRAPH
INSE	ALL	ALL	STYLE	

*In the example, ignore all forces except gravity. For initial velocity  $v_0$  and angle  $\theta$ , the position of the ball as a function of time has horizontal and vertical components.*



- 6 Enter these window variable values.

**tMin=0**      **xMin=-20**      **yMin=-5**  
**tMax=5**      **xMax=100**      **yMax=15**  
**tStep=.1**      **xScl=50**      **yScl=10**

2nd [M2] 0 ▾ 5 ▾  
 . 1 ▾ (←) 20 ▾  
 100 ▾ 50 ▾ (←) 5  
 ▾ 15 ▾ 10

```

WINDOW
tXMin=-20
xMax=100
xScl=50
yMin=-5
yMax=15
yScl=10
E(←)= WIND ZOOM TRACE GRAPH
  
```

- 7 Set **SimulG** and **AxesOff** graphing formats, so the path of the ball and the vectors will be plotted simultaneously on a clear graph screen.

MORE [F3] ▾ ▾ ▾  
 ▸ [ENTER] ▾ ▾ ▾  
 [ENTER]

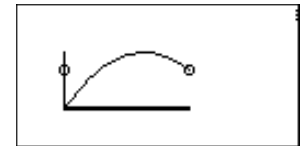
```

RectGC PolarGC
CoordOn CoordOff
DrawLine DrawDot
SeqS SimulG
GridOff GridOn
AxesOn AxesOff
LabelOff LabelOn
E(←)= WIND ZOOM TRACE GRAPH
  
```

To simulate the ball in flight, change the graph style of **xt1/yt1** to  $\dot{\phantom{x}}$  (animate).

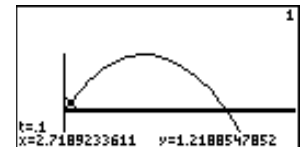
- 8 Plot the graph. The plotting action simultaneously shows the ball in flight and the vertical and horizontal component vectors of the motion.

[F5]



- 9 Trace the graph to obtain numerical results. Tracing begins at **tMin** and traces the path of the ball over time. The value displayed for **x** is distance; **y** is height; **t** is time.

[F4] ▸



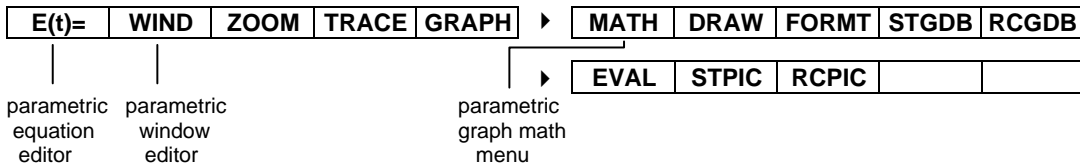
## Defining a Parametric Graph

The steps for defining a parametric graph are similar to the steps for defining a function graph. This chapter assumes that you are familiar with Chapter 5: Function Graphing and Chapter 6: Graph Tools. This chapter details those aspects of parametric graphing that differ from function graphing.

### Setting Parametric Graphing Mode

To display the mode screen, press  $\boxed{2\text{nd}} \boxed{[\text{MODE}]}$ . To graph parametric equations, you must select **Param** graphing mode before you enter equations, set the format, or edit window variable values. The TI-86 retains in memory separate equation, format, and window data for each graphing mode.

#### The GRAPH Menu $\boxed{[\text{GRAPH}]}$



Chapter 5 describes these GRAPH menu items: **GRAPH** and **FORMT**.

Chapter 6 describes these GRAPH menu items: **ZOOM**, **TRACE**, **DRAW**, **STGDB**, **RCGDB**, **EVAL**, **STPIC**, and **RCPIC**.

### Displaying the Parametric Equation Editor

To display the parametric equation editor, select **E(t)=** from the GRAPH menu in **Param** graphing mode ( $\boxed{[\text{GRAPH}]} \boxed{[\text{F1}]}$ ). The equation editor menu displayed on the bottom line is the same as the **Func**-mode equation editor menu, except that **t** and **xt** replace **x** and **y**, and **yt** displaces **INSf**.

In this editor, you can enter and display both the **x** and **y** components of up to 99 parametric equations, **xt1** and **yt1** through **xt99** and **yt99**, if sufficient memory is available. Each is defined in terms of the independent variable **t**.

Two components, **x** and **y**, define a single parametric equation. You must define both **xt** and **yt** for each equation. The default graph style is  $\setminus$  (line) in **Param** mode.  $\blacksquare$  (shade above) and  $\blacksquare$  (shade below) graph styles are not available in **Param** mode.

A common application of parametric graphs is graphing equations over time.



### Selecting and Deselecting a Parametric Equation

When a parametric equation is selected, the equals signs (=) of both **xt** and **yt** are highlighted. To change the selection status of a parametric equation, move the cursor onto either **xt** or **yt**, and then select **SELCT** from the equation editor menu. The status is changed for **xt** and **yt**.

### Deleting a Parametric Equation

To delete a parametric equation using **DELf**, move the cursor to either **xt** or **yt**, and then select **DELf** from the equation editor menu. Both components are deleted.

To delete a parametric equation using the MEM DELET menu (Chapter 17), you must select the **xt** component. If you select the **yt** component, the equation is retained in memory.

### Setting the Graph Screen Window Variables

To display the parametric window editor, select **WIND** from the GRAPH menu (**GRAPH** **F2**). **Param** graphing mode has the same window variables as **Func** graphing mode, except:

- ◆ **xRes** is not available in **Param** mode.
- ◆ **tMin**, **tMax**, and **tStep** are available in **Param** mode.

The values shown in the picture to the right are the defaults in **Radian** mode. ↓ indicates that **yMin=-10**, **yMax=10**, and **yScl=1** are beyond the screen.

```
WINDOW
tMin=
tMax=6.28318530718
tStep=.130899693899...
xMin=-10
xMax=10
↓xScl=1
EQ= WIND ZOOM TRACE GRAPH
```

<b>tMin=0</b>	Specifies the starting <b>t</b> value
<b>tMax=6.28318530718</b>	Specifies the ending <b>t</b> value
<b>tStep=.13089969389957</b>	Specifies the increment from one <b>t</b> value to the next

**tMax** default is  $2\pi$ .

**tStep** default is  $\pi/24$ .

**DrawLine** graph format typically displays a more meaningful parametric graph than **DrawDot** graphing format.

### Setting the Graph Format

To display the format screen in **Param** graphing mode, select **FORMAT** from the GRAPH menu ((GRAPH) MORE F3). Chapter 5 describes the format settings. The TI-86 retains in memory separate format settings for **Func**, **Pol**, **Param**, and **DifEq** graphing modes.

### Displaying the Graph

To plot the selected parametric equations, you can select **GRAPH**, **TRACE**, **EVAL**, **RCGDB**, or a **ZOOM**, **MATH**, **DRAW**, or **RPIC** operation. The TI-86 evaluates **x** and **y** for each value of **t** (from **tMin** to **tMax** in intervals of **tStep**) and then plots each point defined by **x** and **y**. As the graph is plotted, the variables **x**, **y**, and **t** are updated.

## Using Graph Tools in Param Graphing Mode

### The Free-Moving Cursor

The free-moving cursor in **Param** graphing works the same as in **Func** graphing.

- ◆ In **RectGC** format, moving the cursor updates the values of **x** and **y**; if **CoordOn** format is selected, **x** and **y** are displayed.
- ◆ In **PolarGC** format, moving the cursor updates **x**, **y**, **r**, and  $\theta$ ; if **CoordOn** format is selected, **r** and  $\theta$  are displayed.

### Tracing a Parametric Function

To begin a trace, select **TRACE** from the GRAPH menu ((GRAPH) F4). When you begin a trace, the trace cursor is on the first selected function at **tMin**.

- ◆ In **RectGC** format, moving the trace cursor updates the values of **x**, **y**, and **t**; if **CoordOn** format is selected, **t**, **x**, and **y** are displayed.

- ◆ In **PolarGC** format, moving the trace cursor updates  $x$ ,  $y$ ,  $r$ ,  $\theta$ , and  $t$ ; if **CoordOn** format is selected,  $r$ ,  $\theta$ , and  $t$  are displayed. The  $x$  and  $y$  (or  $r$  and  $\theta$ ) values are calculated from  $t$ .

To move the trace cursor...	Press:
Along the graph of the equation by increments or decrements of <b>tStep</b>	$\blacktriangleright$ or $\blacktriangleleft$
From one equation to another	$\blacktriangledown$ or $\blacktriangleup$

*QuickZoom is available in **Param** graphing; panning is not (Chapter 6).*

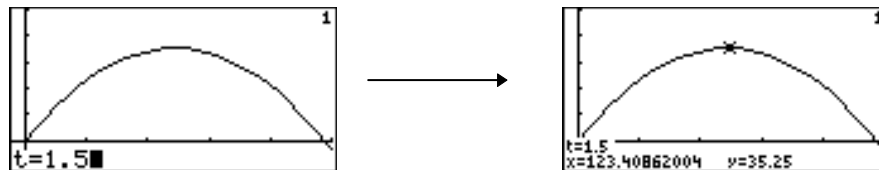
If you move the trace cursor beyond the top or bottom of the graph screen, the coordinate values at the bottom of the screen continue to change appropriately. If you have graphed a family of curves,  $\blacktriangledown$  and  $\blacktriangleup$  move through each curve before moving to the next parametric function.

### Moving the Trace Cursor to a $t$ Value

*You can enter an expression at the **t=** prompt.*

To move the trace cursor to any valid  $t$  value on the current equation, enter the number. When you enter the first digit, a **t=** prompt is displayed in the bottom-left corner. The value you enter must be valid for the current graph screen. When you have completed the entry, press **ENTER** to reactivate the trace cursor.

*In the example, the parametric equation is:  
 $xt1=95t \cos 30^\circ$   
 $yt1=95t \sin 30^\circ - 16t^2$   
 Also, **AxesOn** graph format is set.*



*(The example on page 124 is similar to this example.)*

### Using Zoom Operations

The GRAPH ZOOM menu items, except **ZFIT**, work the same in **Param** graphing as in **Func** graphing. In **Param** mode, **ZFIT** adjusts the graph screen in both the  $x$  and  $y$  directions.

The GRAPH ZOOM menu items affect only the **x** window variables (**xMin**, **xMax**, and **xScl**) and the **y** window variables (**yMin**, **yMax**, and **yScl**), except **ZSTO** and **ZRCL**, which also affect the **t** window variables (**tMin**, **tMax**, and **tStep**).

### The GRAPH MATH Menu

GRAPH MORE F1

MATH	DRAW	FORMT	STGDB	RCGDB				
DIST	dy/dx	dy/dt	dx/dt	ARC	▶	TANLN		

The other GRAPH MATH menu items are the same as described in Chapter 5.

**dy/dx** Returns the derivative of **yt** divided by the derivative of **xt**

**dy/dt** Returns the derivative of the **yt** equation at a point with respect to **t**

**dx/dt** Returns the derivative of the **xt** equation at a point with respect to **t**

The distances calculated by **DIST** and **ARC** are distances in the rectangular coordinate plane.

At a point where the derivative is undefined, **TANLN** will draw the line, but no result is displayed or stored in **Ans**.

### Evaluating an Equation for a Specified t

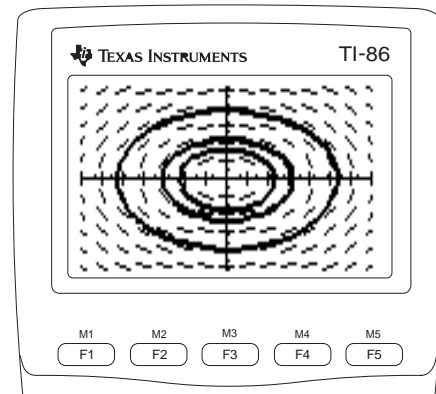
When the trace cursor is not active, the GRAPH menu item **EVAL** evaluates selected polar equations directly on the graph for a given value of **t**. **eval** in a program or from the home screen returns a list of x and y values in this form:  $\{xt1(t) yt1(t) xt2(t) yt2(t) \dots\}$ .

### Drawing on a Parametric Graph

The DRAW menu items work in **Param** graphing the same as in **Func** graphing. DRAW instruction coordinates in **Param** graphing are the **x**- and **y**-coordinate values of the graph screen.

# 10 Differential Equation Graphing

Defining a Differential Equation Graph..... 132  
Entering and Solving Differential Equations ..... 139  
Using Graph Tools in DifEq Graphing Mode ..... 144



*Chapters 8 and 9 each begin with an example; Chapter 10 has several differential equation examples throughout the chapter.*

## Defining a Differential Equation Graph

Most steps for defining a differential equation graph are similar to the steps for defining a function graph. This chapter assumes that you are familiar with Chapter 5: Function Graphing and Chapter 6: Graph Tools. This chapter details aspects of differential equation graphing that differ from function graphing.

Generally, **DifEq** graphing mode differs from other graphing modes in these ways.

- ◆ You must select the field format or accept the default before defining the equations (page 133).
- ◆ If an equation is higher than first order, you must convert it to an equivalent system of first-order differential equations, and then store the system in the equation editor (page 140 and page 142).
- ◆ When **FldOff** field format is selected, you must set initial conditions for each equation in the system (page 136).
- ◆ After you have selected the field format setting, you must select **AXES** from the GRAPH menu and enter axes information or accept the defaults (page 137).

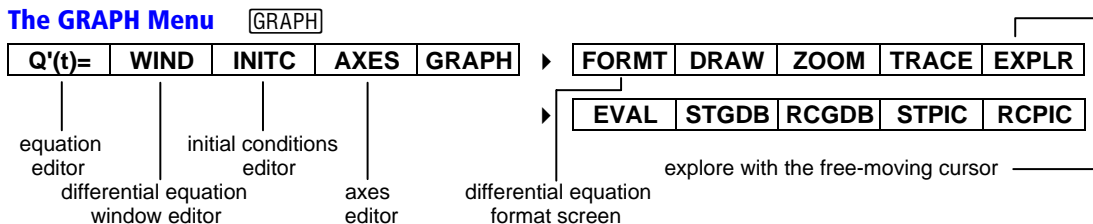
### Setting Differential Equation Graphing Mode

To display the mode screen, press  $\boxed{2\text{nd}}$  [MODE]. To graph differential equations, you must select **DifEq** graphing mode before you set the format, enter equations, or edit window variable values. The TI-86 retains in memory separate format, equation, and window data for each graphing mode.



Chapter 5 describes the GRAPH menu item GRAPH.

Chapter 6 describes these GRAPH menu items: DRAW, ZOOM, TRACE, EVAL, STGDB, RCGDB, STPIC, and RCPIC.



### Setting the Graph Format

To display the format screen in **DifEq** graphing mode, select **FORMT** from the GRAPH menu (GRAPH MORE F1).

- ◆ The **RK Euler** and **SlpFld DirFld FldOff** format settings are available only in **DifEq** mode.
- ◆ The **RectGC PolarGC**, **DrawLine DrawDot**, and **SeqG SimulG** format settings are not available in **DifEq** graphing mode.
- ◆ All other format settings are the same as described in Chapter 5.



The TI-86 retains independent format settings for each graphing mode.

### Solution Method Format

- RK** Uses the Runge-Kutta method to solve differential equations more accurately than the **Euler** solution method format, but not as fast
- Euler** Uses the Euler method to solve differential equations; requires a number of iterations between **tStep** values, so **EStep=** prompt replaces **difTol=** prompt on the window editor

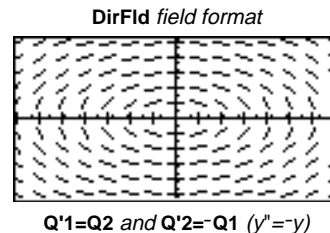
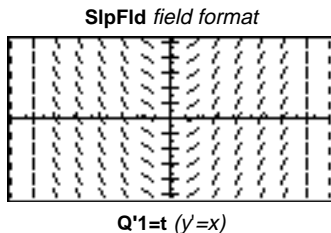
Field Format

- SlpFld** (slope field) Adds the slope field to the graph of only one first-order equation with **t** on the x-axis and a specified **Qn** equation on the y-axis
- DirFld** (direction field) Adds the direction field to the graph of only one second-order equation with **Qx#** on the x-axis and **Qy#** on the y-axis
- FldOff** (field off) Graphs all selected differential equations with **t** or **Q1** on the x-axis, **Q1** or **Q2** on the y-axis, and no field; initial conditions must be defined for all equations (page 136)

The examples below show the basic slope and direction fields; all unspecified settings and values are defaults. To replicate these examples, reset defaults, enter the specified information in **DifEq** graphing mode, and then press **GRAPH** **F5**.

Axes information is stored to **GDB** and **PIC** variables.

To remove menus from a graph, as shown in the examples, press **CLEAR**.



**Displaying the Differential Equation Editor**

To display the differential equation editor, select **Q'(t)=** from the **GRAPH** menu in **DifEq** graphing mode (**GRAPH** **F1**). The **DifEq** equation editor menu on the bottom line is the same as the **Func** mode equation editor menu, except that **t** and **Q** replace **x** and **y**.

In this editor, you can enter and display a system of up to nine first-order differential equations,  $Q'1$  through  $Q'9$ , if sufficient memory is available. Equations are defined in terms of the independent variable  $t$  and/or  $Q'$ .

You can refer to another differential equation variable in a **DifEq** equation, as in  $Q'2=Q1$ . However, you cannot enter a list in a **DifEq** equation.

When the TI-86 calculates a differential equation system, it references all equations in the equation editor, regardless of selection status, starting at  $Q'1$ . You must define  $Q'n$  equation variables consecutively, starting at  $Q'1$ . For example, if  $Q'1$  and  $Q'2$  are not defined, but you attempt to solve an equation defined in  $Q'3$ , the calculator returns an error.

The TI-86 allows you to analyze each equation independently. For example, you can enter  $Q'1=t$  and  $Q'2=t^2$  and analyze each equation independently.

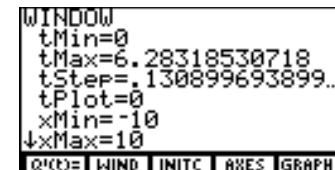
The TI-86 graphs only those selected equations that are appropriate for the specified axes.

- ◆ The default graph style is  $\blacksquare$  (thick) in **DifEq** mode.
- ◆  $\blacksquare$  (shade above),  $\blacksquare$  (shade below), and  $\cdot$  (dot) are not available in **DifEq** graphing mode.

### Setting the Graph Screen Window Variables

To display the differential equation window editor, select **WIND** from the **GRAPH** menu (**[GRAPH]** **[F2]**). **DifEq** has the same window variables as **Func** graphing mode, except:

- ◆ **xRes** is not available in **DifEq** mode.
- ◆ **tMin**, **tMax**, **tStep**, and **tPlot** are available in **DifEq** mode.
- ◆ **difTol** (RK) and **EStep** (Euler) are available in **DifEq** mode.



The values shown in the picture on page 135 are defaults in **Radian** mode. **x** and **y** settings correspond to the axes variables (page 137). ↓ indicates that **xScl=1**, **yMin=-10**, **yMax=10**, **yScl=1**, and **difTol=.001** (in **RK** format) or **EStep=1** (in **Euler** format) are beyond the screen.

**tMax** default is  $2\pi$ .

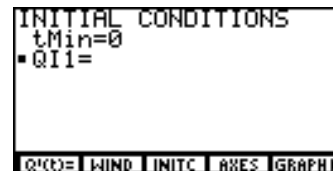
**tStep** default is  $\pi/24$ .

<b>tMin=0</b>	Specifies the <b>t</b> value at which to begin evaluating within a graph screen
<b>tMax=6.28318530718</b>	Specifies the last <b>t</b> value to evaluate within a graph screen
<b>tStep=.1308969389958</b>	Specifies the increment from one <b>t</b> value to the next <b>t</b> value
<b>tPlot=0</b>	Specifies the point at which plotting begins (ignored when <b>t</b> is an axis)
<b>difTol=.001</b> (in <b>RK</b> format)	Specifies tolerance to help select step size for solving; must be $\geq 1E-12$
<b>EStep=1</b> (in <b>Euler</b> format)	Specifies Euler iterations between <b>tStep</b> values; must be an integer $>0$ and $\leq 25$

### Setting the Initial Conditions

Initial conditions information is stored to **GDB** and **PIC** variables.

To display the initial conditions editor, select **INITC** from the **GRAPH** menu (**GRAPH** **F3**). On this editor, you can set the initial value at **t=tMin** for each first-order equation in the equation editor.



**tMin** is the first **t** value to evaluate. **QI1** is the initial value of **Qn**. A small square next to an initial condition variable indicates that a value is required for a defined differential equation.

You can enter an expression, list, or list name for initial conditions **tMin** and **QIn**. When you enter a list name, the elements are displayed when you press **ENTER**, **↓** or **↑**.

- ◆ If **SlpFld** or **DirFld** format is set, you need not specify initial conditions. The TI-86 returns the appropriate field with no specific solutions.
- ◆ If **FldOff** format is set, you **must** specify initial conditions.

## Setting the Axes

To display the axes editor, select **AXES** from the GRAPH menu in **DifEq** mode ( $\boxed{\text{GRAPH}}$   $\boxed{\text{F4}}$ ).

**x=** assigns a variable to the x-axis

**dTime=** specifies a point in time (real number)

**y=** assigns a variable to the y-axis

**fldRes=** (resolution) sets number of rows (1 through 25)

At the **x=** and **y=** prompts, you can enter the independent variable **t**, as well as **Q**, **Q'**, **Q<sub>n</sub>**, or **Q'<sub>n</sub>**, where *n* is an integer  $\geq 1$  and  $\leq 9$ . If you assign **t** to one axis and **Q<sub>n</sub>** or **Q'<sub>n</sub>** to the other axis, only the equation stored to **Q<sub>n</sub>** or **Q'<sub>n</sub>** is plotted; other differential equations in the equation editor are not plotted; their selection status is ignored. **dTime** is only valid for second-order equations with **t** in either equation.

The axes editor and defaults for each field format are shown below. When **SlpFld** field format is set, the x-axis is always **t**, so the AXES: SlpFld editor does not display **x=t**.

*Axes information is stored to GDB and PIC variables.*

When **SlpFld** format is set:

```
AXES: SlpFld
y=Q1
fldRes=15

Q'(t)= WIND INITC AXES GRAPH
Q
```

When **DirFld** format is set:

```
AXES: DirFld
x=Q1
y=Q2
dTime=0
fldRes=15

Q'(t)= WIND INITC AXES GRAPH
Q
```

When **FldOff** format is set:

```
AXES: FldOff
x=t
y=Q

Q'(t)= WIND INITC AXES GRAPH
Q t Q'
```

## Differential Equation Graphing Tips

- ◆ Since the TI-86 plots slope fields and direction fields before it plots equations, you can press  $\boxed{\text{ENTER}}$  to pause the graph and view the fields with no solutions plotted.
- ◆ If you do not specify initial conditions for the equations assigned to the axes, the TI-86 simply draws the field and stops. This gives you access to both the field format options and the interactive initial conditions simultaneously.

*Stat plot and screen drawings are not stored to fldPic.*

### The Built-In Variable fldPic

As the TI-86 plots a field, it stores the field and any displayed label, axes, or cursor coordinate information to the built-in variable **fldPic**.

These actions do not update **fldPic**.

- ◆ Switching the solving method format from **RK** to **Euler** or from **Euler** to **RK**
- ◆ Entering or editing any initial condition variable value (**Q11** through **Q19**)
- ◆ Editing a value for **difTol**, **EStep**, **tMin**, **tMax**, **tStep**, or **tPlot**
- ◆ Changing a graph style

These actions update **fldPic**.

- ◆ Editing an equation in the equation editor
- ◆ Re-assigning an axis, editing a **dTime** value, or editing a **fldRes** value
- ◆ Using a GRAPH ZOOM menu item
- ◆ Changing a format setting other than solving method format
- ◆ Editing a value for **xMin**, **xMax**, **xScl**, **yMin**, **yMax**, or **yScl**

### Displaying the Graph

To plot the differential equations, you can select **GRAPH**, **TRACE**, **EVAL**, or **STGDB**, or a **DRAW**, **ZOOM**, or **STPIC** operation, from the GRAPH menu. The TI-86 solves each equation from **tMin** to **tMax**. If **t** is not an axis, it plots each point beginning at **tPlot**; otherwise, it begins at **tMin**. As a graph is plotted, the variables **x**, **y**, **t**, and **Qn** are updated.

**tStep** affects trace resolution and graph appearance, but not the accuracy of the trace values. **tStep** does not determine the step size for solving; using the **RK** algorithm (Runge-Kutta 2-3) determines the step size. If the x-axis is **t**, setting  $\mathbf{tStep} < (\mathbf{tMax} - \mathbf{tMin}) / 126$  increases plotting time without increasing accuracy.

## Entering and Solving Differential Equations

In **Func** graphing mode, **x** is the independent variable and **y** is the equation variable. To avoid conflict between **Func** equations and **DifEq** equations on the TI-86, **t** is the independent variable and **Q'n** is the equation variable in **DifEq** graphing mode. Therefore, when you enter an equation in the differential equation editor, you must express it in terms of **t** and **Q'n**.

For example, to express the first-order differential equation  $y' = x^2$ , you would substitute  $t^2$  for  $x^2$  and  $Q'1$  for  $y'$ , and then enter  $Q'1=t^2$  in the equation editor.

### Graphing in SlpFld Format

- 1 Display the mode screen and set **DifEq** graphing mode.
- 2 Display the format screen and set **SlpFld** field format.
- 3 Display the equation editor and store the differential equation  $y' = x^2$ , substituting **Q'1** for  $y'$  and **t** for  $x$ . Clear any other equations.
- 4 Display the initial conditions editor and enter the initial conditions. A small square indicates that an initial condition is required.

2nd [MODE]  $\downarrow$   $\downarrow$   $\downarrow$   
 $\downarrow$   $\rightarrow$   $\rightarrow$   $\rightarrow$  ENTER

GRAPH MORE F1  $\downarrow$   
 $\downarrow$   $\downarrow$   $\downarrow$   $\downarrow$  ENTER

F1 F1  $x^2$

2nd [M3] 3



*In the example, the default window variable values are set initially.*

In **SlpFld** field format,  $x=t$  is always true;  $y=Q1$  and  $fldRes=15$  are the default axes settings.

- ⑤ Display the axes editor and enter the equation variable for which you want to solve. (Do not set  $y=Q$ .)
- ⑥ Accept or change **fldRes** (resolution).
- ⑦ Display the graph. With the default window variable values set, the slope fields for this graph are not very illustrative.
- ⑧ Change the window variables **xMin**, **xMax**, **yMin**, and **yMax**.
- ⑨ Select **TRACE** from the GRAPH menu to replot the graph and activate the trace cursor. Trace the solution. The trace cursor coordinates for **t** and **Q1** are displayed.

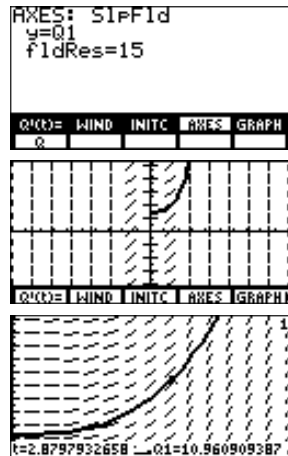
**F4** **F1** 1

**2nd** **[M5]**

**F2** **▼** **▼** **▼** **▼** 0  
**▼** 5 **▼** **▼** 0 **▼** 20

**MORE** **F4**

**▶** and **◀**



### Transforming an Equation into a First-Order System

On the TI-86, to enter a second-order or higher (up to ninth-order) differential equation, you must transform it to a system of first-order differential equations. For example, to enter the second-order differential equation  $y'' = -y$ , you must transform it to two first-order differential equations, as shown in the chart below.

Differentiate...	Define the variables as...	And then substitute:
$Q'1=y'$	$Q1=y$	$Q'1=Q2$ (since $Q'1=y'=Q2$ )
$Q'2=y''$	$Q2=y'$	$Q'2=-Q1$



## Graphing in DirFld Format

- 1 Display the mode screen and set **DifEq** graphing mode.
- 2 Display the format screen and set **DirFld** graphing format.
- 3 Display the equation editor and store the transformed system of differential equations for  $y''=-y$  to the equation editor, substituting **Q1** for  $y$  and **Q2** for  $y'$ .
- 4 Display the initial conditions editor and enter the initial conditions if you want a specific solution. To enter a list of initial conditions, use { and } from the LIST menu.
- 5 Display the axes editor and enter the two equation variables for which you want to solve. You must omit the prime mark ( ' ).
- 6 Accept or change **fldRes** (resolution).
- 7 Select **ZSTD** from the GRAPH ZOOM menu to set the standard window variable values and display the graph.
- 8 Clear the GRAPH menu from the screen.

**2nd** [MODE]  $\downarrow$   $\downarrow$   $\downarrow$   
 $\downarrow$   $\rightarrow$   $\rightarrow$   $\rightarrow$  **ENTER**

**GRAPH** **MORE** **F1**  $\downarrow$   
 $\downarrow$   $\downarrow$   $\downarrow$   $\downarrow$   $\rightarrow$  **ENTER**

**F1** **F2** **2**  $\downarrow$  **(-)** **F2** **1**

**2nd** [M3] **2nd** [LIST]  
**F1** **1**  $\downarrow$  **2**  $\downarrow$  **5** **F2**  
 $\downarrow$  **F1** **2nd** [ $\pi$ ]  $\downarrow$  **4**  
 $\downarrow$  **5**  $\downarrow$  **75** **F2**

**2nd** [M4] **F1** **1**  $\downarrow$   
**F1** **2**

**EXIT** **MORE** **F3** **F4**

**CLEAR**

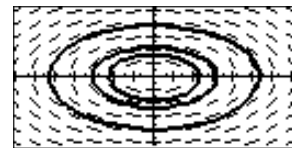
```
Func Pol Param DIRF1
Dec Bin Oct Hex
Rectl CylV SphereV
```

```
DirFld DIRF1 FldOfff
Q1= WIND INITC AXES GRAPH
```

```
Plot1 Plot2 Plot3
Q'1=Q2
Q'2=-Q1
MODE WIND INITC AXES GRAPH
t Q INSF DELE SELCT
```

```
INITIAL CONDITIONS
tMin=0
Q11=(1,2,5)
Q12=( $\pi$ ,4,5.75)
```

```
AXES: DirFld
x=Q1
y=Q2
dTime=0
fldRes=15
Q1= WIND INITC AXES GRAPH
Q
```



In **DifEq** graphing mode,  $t$  is the independent variable and  $Q^n$  is the dependent variable, where  $n \geq 1$  and  $\leq 9$ .

In the example, the default window variable values are set initially.

When **DirFld** field format is selected,  $x=Q1$ ,  $y=Q2$ ,  $dTime=0$ , and  $fldRes=15$  are the default axes settings. Since  $t$  is not part of the equation,  $dTime$  is ignored.

### Graphing a System of Equations in FldOff Format

For this example, you must transform the fourth-order differential equation  $y^{(4)} - y = e^{-x}$  into an equivalent system of first-order differential equations, as shown in the chart below.

Differentiate...	Define the variables as...	And then substitute:
$Q'1=y'$	$t=x$ $Q1=y$	$Q'1=Q2$ (since $Q'1=y'=Q2$ )
$Q'2=y''$	$Q2=y'$	$Q'2=Q3$
$Q'3=y'''$	$Q3=y''$	$Q'3=Q4$
$Q'4=y^{(4)}$	$Q4=y'''$	$Q'4=e^{-t}+Q1$ (since $Q'4=y^{(4)}=e^{-x}+y=e^{-t}+Q1$ )

- 1 Display the mode screen and set **DifEq** graphing mode.
- 2 Display the format screen and set **FldOff** field format.
- 3 Display the equation editor and store the transformed system of differential equations for  $y^{(4)}=e^{-x}+y$ , substituting as shown in the chart.
- 4 Deselect **Q'3**, **Q'2**, and **Q'1** to plot **Q'4= $e^{-t}+Q1$**  only.

$\boxed{2nd}$   $\boxed{[MODE]}$   $\boxed{\downarrow}$   $\boxed{\downarrow}$   
 $\boxed{\downarrow}$   $\boxed{\rightarrow}$   $\boxed{\rightarrow}$   $\boxed{\rightarrow}$   $\boxed{ENTER}$

$\boxed{GRAPH}$   $\boxed{MORE}$   $\boxed{F1}$   $\boxed{\downarrow}$   
 $\boxed{\downarrow}$   $\boxed{\downarrow}$   $\boxed{\downarrow}$   $\boxed{\rightarrow}$   $\boxed{\rightarrow}$   
 $\boxed{ENTER}$

$\boxed{F1}$   $\boxed{F2}$   $\boxed{2}$   $\boxed{\downarrow}$   $\boxed{F2}$   $\boxed{3}$   $\boxed{\downarrow}$   
 $\boxed{F2}$   $\boxed{4}$   $\boxed{\downarrow}$   $\boxed{2nd}$   $\boxed{e^{-x}}$   $\boxed{[ ]}$   
 $\boxed{(-)}$   $\boxed{F1}$   $\boxed{)}$   $\boxed{+}$   $\boxed{F2}$   $\boxed{1}$

$\boxed{\uparrow}$   $\boxed{F5}$   $\boxed{\uparrow}$   $\boxed{F5}$   $\boxed{\uparrow}$   $\boxed{F5}$

```
Func Pol Param DifEq
Deg Bin Oct Hex
Rectl CylU SphereV
```

```
SlpFld DirFld FldOff
MODE WIND INITC AXES GRAPH
```

```
Plot1 Plot2 Plot3
\Q'1=Q2
\Q'2=Q3
\Q'3=Q4
\Q'4=E^(-t)+Q1
MODE WIND INITC AXES GRAPH
t Q INSP DELF SELECT
```

*In DifEq graphing mode, t is the independent variable and Q'n is the equation variable, where n ≥ 1 and ≤ 9.*

- 5 Display the window editor and set the window variable values.

**2nd** **[M2]** **10**  
**.** **01** **0**  
**(-)** **4** **4**

```
WINDOW
tMin=0
tMax=10
tStep=.01
tPlot=0
xMin=0
xMax=10
xScl=1
yMin=-4
yMax=4
yScl=1
difTol=.001
MODE WIND INITC AXES GRAPH
```

- 6 Display the initial conditions editor and enter the initial conditions. A small square indicates that an initial condition is required.

**F3** **3** **(-)** **5** **.** **25**  
**7** **5**  
**(-)** **5** **.** **75**

```
INITIAL CONDITIONS
tMin=0
■ Q11=3
■ Q12=-5.25
■ Q13=7.5
■ Q14=-5.75
```

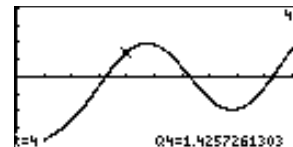
- 7 Display the axes editor. Enter the equation variables for which you want to solve.

**F4**

```
AXES: FldOff
x=t
y=Q
```

- 8 Display the graph. Explore the equation with the trace cursor.

**EXIT** **MORE** **F4**  
**▶** and **◀**



- 9 Enter a **t** value to move the trace cursor to the solution for that **t** value. The **t** and **Q4** coordinates are displayed.

**4** **ENTER**

When **FldOff** field format is selected, **x=t** and **y=Q** are the default axes settings.

t=4 Q4=1.4257261303

To paste to the home screen, you can select it from the CHAR MISC menu or from the CATALOG.

Due to TI-86 system requirements, you must express  $Q1(3)$  as  $Q'1(3)$  on the calculator.

### Solving a Differential Equation for a Specified Value

On the home screen in **DifEq** graphing mode, you can solve a differential equation stored to a specified independent variable value or expression. The syntax is:  $Q'n(\text{value})$ .

- ◆ The equation must be stored to a **DifEq** equation variable ( $Q'1$  through  $Q'9$ ).
- ◆ The initial conditions must be defined.
- ◆ The result sometimes varies, depending on the axes settings.

Plot1 Plot2 Plot3	
Q'1t	
INITIAL CONDITIONS	
tMin=0	
Q11=0	
AXES: S1F1d	
y=Q1	
f1dRes=15	
Q'1(3)	4.5

## Using Graph Tools in DifEq Graphing Mode

### The Free-Moving Cursor

The free-moving cursor works in **DifEq** mode as it does in **Func** graphing. The cursor coordinate values for  $x$  and  $y$  are displayed, and the variables are updated.

### Tracing a Differential Equation

To begin a trace, select **TRACE** from the GRAPH menu ( $\boxed{\text{GRAPH}}$   $\boxed{\text{MORE}}$   $\boxed{\text{F4}}$ ). The trace cursor appears on the first equation at or near **tPlot** (or **tMin**, if **t** is an axis).

The trace coordinates displayed at the bottom of the screen reflect the axes settings. For example, if  $x=t$  and  $y=Q1$ , then **t** and **Q1** are displayed. If **t** is not an axis, three trace values are displayed. If **t** is an axis, only **t** and the variable designated as the y-axis are displayed.

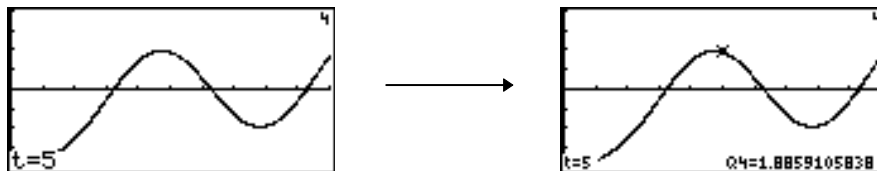
The trace cursor moves in increments or decrements of **tStep**. As you trace an equation, the coordinates are updated and displayed. If the cursor moves off the screen, the coordinate values displayed at the bottom of the screen continue to change appropriately.

*QuickZoom is available in DifEq graphing; panning is not (Chapter 6).*

### Moving the Trace Cursor to a t Value

To move the trace cursor to any valid  $t$  value on the current equation, enter the number. When you enter the first digit, a  $t=$  prompt is displayed in the bottom-left corner. The value you enter must be valid for the current graph screen. When you have completed the entry, press **[ENTER]** to reactivate the trace cursor.

Values for  $t$  and  $Q$  are displayed on the graph to the right because  $x=t$  and  $y=Q$  graph axes are selected.



### Drawing on a Differential Equation Graph

The GRAPH DRAW menu items work the same in **DifEq** graphing mode as in **Func** graphing. DRAW instruction coordinates are the  $x$ - and  $y$ -coordinates of the graph screen.

**DrEqu** is available only in **DifEq** mode. **DrInv** is not available in **DifEq** graphing mode.

### Drawing an Equation and Storing Solutions to Lists

To draw a solution on the current graph screen and store the results to specified list names, the syntax is:

**DrEqu**(*xAxisVariable*,*yAxisVariable*[,*xList*,*yList*,*tList*])

*xAxisVariable* and *yAxisVariable* specify the axes on which the drawing is based; they may differ from the current graph screen's axes settings.

**DrEqu**( does not store values to  $x$ ,  $y$ , or  $t$ .

In the example, the default window variable values are set. If necessary, select **ZSTD** from the GRAPH ZOOM menu.

If you select **FldOff** field format, you must enter initial conditions before you use **DrEqu**(.

$xList$ ,  $yList$ , and  $tList$  are optional list names to which you can store the solutions  $x$ ,  $y$ , and  $t$ . You then can display the lists on the home screen or in the list editor (Chapter 11).

Use the free-moving cursor to select initial conditions.

You cannot trace the drawing. However, you can plot  $xList$ ,  $yList$ , or  $tList$  as a stat plot after you draw the equation, and then trace them (Chapter 14). Also, you can fit statistical regression models to the lists (Chapter 14).

- ① Display the mode screen and set **DifEq** graphing mode.

2nd [MODE]  $\downarrow$   $\downarrow$   $\downarrow$   
 $\downarrow$   $\rightarrow$   $\rightarrow$   $\rightarrow$  [ENTER]

```
Func Pol Param UITER
Dec Bin Oct Hex
Rectl CylV SphereV
```

- ② Display the format screen and set **DirFld** field format.

[GRAPH] [MORE] [F1]  $\downarrow$   
 $\downarrow$   $\downarrow$   $\downarrow$   $\downarrow$   $\downarrow$   $\rightarrow$   
 [ENTER]

```
SlpFld UITER FldOff
Q'(t)= WIND INTC AXES GRAPH
```

- ③ Display the equation editor and store the equations **Q'1=Q2** and **Q'2=-Q1**. (Delete all other equations.)

[F1] [F2] 2  $\downarrow$  (-) [F2] 1

```
Plot1 Plot2 Plot3
NQ'1=Q2
NQ'2=-Q1
```

- ④ Remove the format screen, and then select **DrEqu** from the GRAPH DRAW menu. **DrEqu**( is pasted to the home screen.

[EXIT] [EXIT] [GRAPH]  
 [MORE] [F2] [F1]

```
DrEqu(
```

- ⑤ Assign variables to the  $x$ - and  $y$ -axes.

[ALPHA] [Q] 1  $\downarrow$   
 [ALPHA] [Q] 2  $\downarrow$

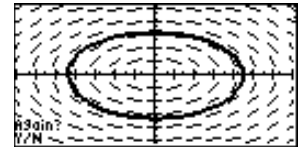
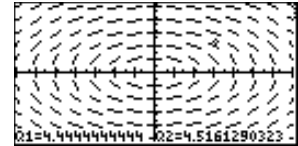
```
DrEqu(Q1,Q2,LX,LY,LT)
```

- ⑥ Specify list names to which to store the solution lists for  $x$ ,  $y$ , and  $t$ .

[ALPHA] [L] [ALPHA] [X]  
 , [ALPHA] [L] [ALPHA]  
 [Y] , [ALPHA] [L]  
 [ALPHA] [T]  $\downarrow$

In the example, since no initial conditions were set, the equation in Q'1 is not plotted.

- 7 Display the graph screen and plot the direction field. [ENTER]
- 8 Move the free-moving cursor to the initial condition coordinates you want. [▶] [▼] [◀] [▲]
- 9 Draw the solution. The solution lists for  $x$ ,  $y$ , and  $t$  are stored to **LX**, **LY**, and **LT**. The **Again?** prompt is displayed and ALPHA-lock is on for [Y] and [N] only.
  - ◆ To use **DrEqu**( again with new initial conditions, press [Y], [▶], [▼], [◀], or [▲].
  - ◆ To leave **DrEqu**( and display the GRAPH menu, press [N] or [EXIT].



### Using ZOOM Operations

The GRAPH ZOOM menu items, except **ZFIT**, work the same in **DifEq** graphing mode as in **Func** graphing mode. In **DifEq** graphing mode, **ZFIT** adjusts the graph screen in both the  $x$  direction and  $y$  direction.

The ZOOM menu items affect only the  $x$  (**xMin**, **xMax**, and **xScl**) and  $y$  (**yMin**, **yMax**, and **yScl**) window variables. The  $t$  window variables (**tMin**, **tMax**, **tStep**, and **tPlot**) are not affected, except with **ZSTD** and **ZRCL**. You may want to edit the  $t$  window variables to ensure that sufficient points are plotted. **ZSTD** sets **difTol**=**.001** and  $t$  and  $Q$  as the axes.

### Drawing Solutions Interactively with EXPLR

- ① Display the mode screen and set **DifEq** graphing mode.

**2nd** [MODE]  $\downarrow$   $\downarrow$   $\downarrow$   
 $\downarrow$   $\rightarrow$   $\rightarrow$   $\rightarrow$  [ENTER]

```
Func Pol Param DifEq
Dec Bin Oct Hex
Recall CylU SphereU
```

- ② Display the format screen and set **FldOff** field format.

[GRAPH] MORE [F1]  $\downarrow$   
 $\downarrow$   $\downarrow$   $\downarrow$   $\downarrow$   $\rightarrow$   $\rightarrow$   
 [ENTER]

```
SIrFld DirFld FldOff
Q'(t)= WIND INITC AXES GRAPH
```

- ③ Display the equation editor and store the equation **Q'1=.001Q1(100-Q1)**. (Delete all other equations.)

[F1] . 001 [F2] 1 [ ]  
 100 [ ] [F2] 1 [ ]

```
Plot1 Plot2 Plot3
Q'1=.001 Q1(100-Q1)
```

- ④ Set the axes to **x=t** and **y=Q1**.

**2nd** [M4]  $\downarrow$   $\rightarrow$  1

```
AXES: FldOff
x=t
y=Q1
```

- ⑤ Display the window editor and set the window variable values.

**2nd** [M2]  $\downarrow$  100  $\downarrow$   
 . 2  $\downarrow$   $\downarrow$   $\downarrow$   
 100  $\downarrow$   $\downarrow$   $\downarrow$  110

```
WINDOW
tMin=0
tMax=100
tStep=.2
tPlot=0
xMin=-10
xMax=100
xScl=1
yMin=-10
yMax=110
yScl=1
difTol=.001
Q'(t)= WIND INITC AXES GRAPH
```

- ⑥ Display the initial conditions editor and enter the initial condition.

[F3] 10

```
INITIAL CONDITIONS
tMin=0
Q1=10
```

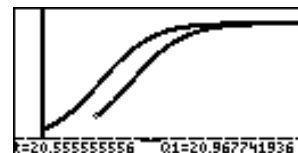
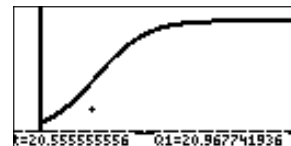


- 7 Select **EXPLR** from the GRAPH menu.
- 8 Move the free-moving cursor to the initial condition for which you want to solve.
- 9 Draw the solution to **Q1**, using the cursor coordinates  $(x,y)$  as initial condition  $(t, Q^1(t))$ .

[MORE] [F5]

[▶] [▼] [◀] [▲]

[ENTER]



To continue drawing more solutions, move the free-moving cursor and then press [ENTER].

To stop using **EXPLR**, press [EXIT].

If **SlpFld** or **DirFld** is set, the axes are set to specific solutions automatically.

- ◆ For **SlpFld**,  $x=t$  and  $y=Q1$  are set.
- ◆ For **DirFld**,  $x=Q1$  and  $y=Q2$  are set.

If the axes are set to a specific solution  $t$ ,  $Q_n$ , or  $Q^n$ , that solution is drawn.

If the axes are not set to a specific solution and  $t$  is one variable and  $Q$  is the other,  $Q1$  is drawn.

If both axes are set to a  $Q$  variable, executing **EXPLR** results in an error.

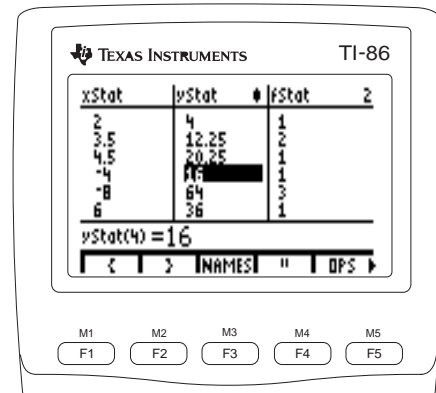
### Evaluating Differential Equations for a Specified $t$

When the trace cursor is not active, the GRAPH menu item **EVAL** evaluates currently selected differential equations  $Q_n$  for a specified value of  $t$ ,  $t_{\text{Min}} \leq t \leq t_{\text{Max}}$ . You can use it directly on the graph. In a program or from the home screen, **eval** returns a list of  $Q$  values.

When **DirFld** or **SlpFld** field format is set, you must specify initial conditions before using **EVAL**.

# 11 Lists

Lists on the TI-86 .....	152
Creating, Storing, and Displaying Lists .....	153
The List Editor .....	156
Using List Operations .....	159
Using Mathematical Functions with Lists .....	161
Attaching a Formula to a List Name .....	162



*The length and number of lists you can store in the TI-86 is limited only by memory capacity.*

*If you enter more than one list in an equation or expression, all lists must have the same number of elements.*

## Lists on the TI-86

A list is a set of real or complex elements, as in **{5,-20,13,9}**. On the TI-86, you can:

- ◆ Enter a list directly in an expression (page 153)
- ◆ Enter a list and store it to a list name (variable) (page 154)
- ◆ Enter a list name in the list editor (page 156), and then enter elements directly or use an attached formula to generate them automatically (page 161)
- ◆ Collect data with the Calculator-Based Laboratory™ (CBL 2™/CBL™) or Calculator-Based Ranger™ (CBR) and store it to a list name on the TI-86 (Chapter 18)
- ◆ Create lists dynamically using the LIST OPS menu item **seq** (page 159)

On the TI-86, you can use a list:

- ◆ As a set of values for an argument in a function to return a list of answers (Chapter 1)
- ◆ As part of an equation to graph a family of curves (Chapter 5)
- ◆ As a set of statistical data to analyze with statistical functions and plot on the graph screen (Chapter 14)

### The LIST Menu 2nd [LIST]

{	}	NAMES	EDIT	OPS
open brace	close brace	list names menu	list editor	list operations menu

When you enter a list, { (open brace) specifies the beginning and } (close brace) specifies the end. To paste { or } to the cursor location, select either from the LIST menu.

The LIST NAMES menu shown here has no user-created list names.

Chapter 14 describes **fStat**, **xStat**, and **yStat**.

### The LIST NAMES Menu $\boxed{2\text{nd}} \boxed{[\text{LIST}]} \boxed{[\text{F3}]}$

{	}	NAMES	EDIT	OPS
fStat	xStat	yStat		

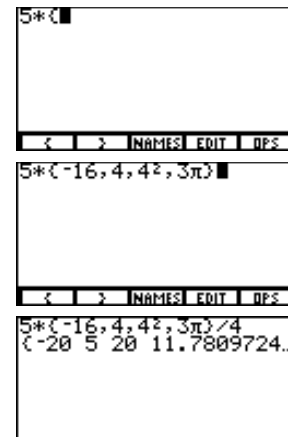
Each user-created list name is added to the LIST NAMES menu and VARS LIST screen. List names, including **fStat**, **xStat**, and **yStat**, are sorted in alphanumeric order in both places.

## Creating, Storing, and Displaying Lists

### Entering a List Directly in an Expression

To enter a list directly, the syntax is:  $\{element1, element2, \dots, element n\}$

- Enter any part of the expression that precedes the list.  $\boxed{5} \boxed{[x]}$
- Select { from the LIST menu to begin the list.  $\boxed{2\text{nd}} \boxed{[\text{LIST}]} \boxed{[\text{F1}]}$
- Enter each list element, separating each from the other with a comma. Each list element can be an expression.  $\boxed{(-)} \boxed{16} \boxed{[ ]} \boxed{4} \boxed{[ ]}$   
 $\boxed{4} \boxed{[x^2]} \boxed{[ ]} \boxed{3}$   
 $\boxed{2\text{nd}} \boxed{[\pi]}$
- Select } from the LIST menu to end the list.  $\boxed{[\text{F2}]}$
- Enter any part of the expression that follows the list.  $\boxed{[ ]} \boxed{4}$
- Evaluate the expression. Any elements that are expressions are evaluated first.  $\boxed{[\text{ENTER}]}$



An ellipsis (...) indicates that a list continues beyond the screen. Use  $\boxed{\rightarrow}$  and  $\boxed{\downarrow}$  to scroll the list.

You need not enter the close brace ( ) when you use  $\boxed{\text{STO}\blacktriangleright}$  to store a list name.

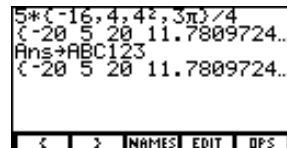
To delete a list name from memory, use the MEM DELETE:LIST screen (Chapter 17).

The TI-86 distinguishes between uppercase and lowercase letters in list names. For example, **ABC123**, **Abc123**, and **abc123** are three different list names.

## Creating a List Name by Storing a List

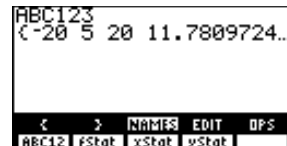
To store a list, the syntax is:  $\{element1, element2, \dots, element n\} \rightarrow listName$

- Enter a list directly. (To store a result expressed as a list and currently stored in **Ans**, as shown in the example, begin these steps at step 2.) (steps 2 through 5 above)
- Paste  $\rightarrow$  to the cursor location. ALPHA-lock is on.  $\boxed{\text{STO}\blacktriangleright}$
- Enter the list name. Either select a name from the LIST NAMES menu or directly enter a name one to eight characters long, starting with a letter.  $\boxed{[A][B][C]}$   
 $\boxed{\text{ALPHA}} \mathbf{1\ 2\ 3}$
- Store the list to the list name.  $\boxed{\text{ENTER}}$



## Displaying List Elements Stored to a List Name

- Enter the list name on the home screen; either select it from the LIST NAMES menu or enter the characters.  $\boxed{2\text{nd}} \boxed{[\text{LIST}]} \boxed{[F1]}$
- Display the list elements.  $\boxed{\text{ENTER}}$



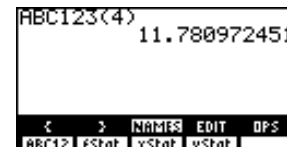
## Displaying or Using a Single List Element

To display or use a single list element, the syntax is: *listName(element#)*

*listName(element#)* is valid as part of an expression.

*element#* is  $\geq 1$  and  $\leq$  the dimension of the list.

- 1 Enter the list name; either select it from the LIST NAMES menu or enter the characters. 2nd [LIST] F3  
F1
- 2 In parentheses, enter the element's place number in the list. [ ( ] 4 ]
- 3 Display the list element. ENTER



## Storing a New Value to a List Element

To store a value to a current element or one element beyond the end of a list, the syntax is: *value*→*listName(element#)*

*value* can be an expression.

- 1 Enter the value to be stored in a current list element or one element beyond the end. 2nd [√] 18
- 2 Paste → to the cursor location. STO▶
- 3 Enter the list name; either select it from the LIST NAMES menu or enter the characters. F1
- 4 Enter the element's place number in parentheses. (In the example, **5** is one beyond the current dimension of **ABC123**.) ALPHA [ ( ] 5 ]
- 5 Enter the new value to the element number. ( $\sqrt{18}$  is evaluated and added as the fifth element.) ENTER



### Complex List Elements

A complex number can be a list element. If at least one list element is a complex number, all elements in the list are displayed as complex. ( $\sqrt{-4}$  results in a complex number.)

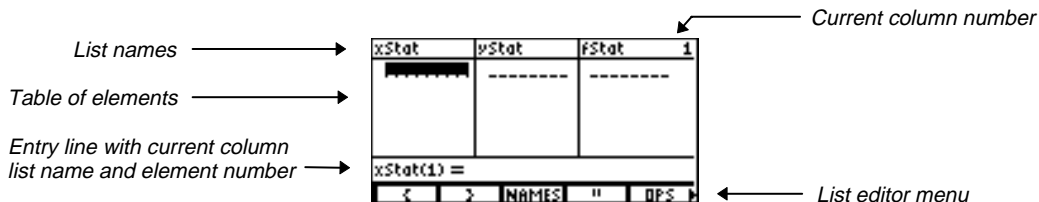
```
(1,2,√-4)
((1,0) (2,0) (0,2))
```

### The List Editor 2nd [LIST] F4

The list editor is a table where you can store, edit, and view up to 20 lists that are in memory. Also, you can create list names and attach formulas to lists in the list editor.

You also can press 2nd [STAT] F2 to display the list editor.

The list editor abbreviates list names and element values when necessary. The entry line displays entire list names and element values.



### The List Editor Menu 2nd [LIST] F4

{	}	NAMES	"	OPS	▶	▶REAL			
---	---	-------	---	-----	---	-------	--	--	--

The list editor menu items {, }, **NAMES**, and **OPS** are identical to the LIST menu items (page 152).

- " Designates the beginning and end of a formula to be attached to a list name
- ▶REAL Converts the current list to a list of real numbers

To use LIST OPS menu items (or any other functions or instructions) in the list editor, the cursor location must be appropriate for the result. For example, you can use the LIST OPS menu item **sortA** when a list name is highlighted but not when an element is highlighted.



After memory is reset, **xStat**, **yStat**, and **fStat** are stored to columns 1, 2, and 3. Resetting defaults does not affect the list editor.

To move from the list name in column 1 to the unnamed column, press  $\leftarrow \rightarrow$ .

### Creating a List Name in the Unnamed Column

- 1 Display the list editor.  $2^{\text{nd}}$  [LIST] [F4]
- 2 Move the cursor to the unnamed column (column 4). The **Name=** prompt is displayed in the entry line. ALPHA-lock is on.  $\leftarrow \rightarrow \rightarrow \rightarrow$
- 3 Enter the list name. The list name is displayed at the top of the current column. In the entry line, a list name prompt is displayed. The name becomes a LIST NAMES menu item and a VARS LIST screen item.  $[X][Y][Z]$  [ENTER]

yStat	fStat		4
-----	-----		
Name=X <del>Y</del> Z <del>0</del>			
ABC12	WY2	fStat	yStat
yStat	fStat	WY2	4
-----	-----		
WY2 =			
$\leftarrow$ $\rightarrow$ NAMES " " OPS $\rightarrow$			

### Inserting a List Name into the List Editor

- 1 Move the cursor to column 3.  $\leftarrow$
- 2 Insert a new, unnamed column. List names shift right, clearing column 3. The **Name=** prompt and LIST NAMES menu are displayed.  $2^{\text{nd}}$  [INS]
- 3 Select **ABC12** from the LIST NAMES menu to insert the list name **ABC123** into column 3. Elements stored to **ABC123** fill the column 3 table of elements. The full value of all **ABC123** elements is displayed in the entry line.  $F1$  [ENTER]

yStat		fStat	3
-----		-----	
Name=ABC123			
ABC12	WY2	fStat	yStat
yStat	ABC123	fStat	3
-----	-20	-----	
	5		
	20		
	11.78097		
	4.242641		
-----			
ABC123 = (-20, 5, 20, 11.7...			
$\leftarrow$ $\rightarrow$ NAMES " " OPS $\rightarrow$			

If all 20 columns have list names, you must remove a list name to make room for the unnamed column.

To cancel the list name insertion, press [CLEAR].

If a formula were attached to **ABC123**, the formula would be displayed in the entry line instead of the list shown in step 3 (page 162.)

To cancel any editing and restore the original element at the cursor, press **CLEAR** **ENTER**.

You can enter an expression as an element.

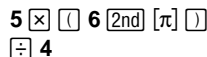
## Displaying and Editing a List Element

- 1 Move the cursor onto the fifth element of **ABC123**. In the entry line, the list name, the element number in parentheses, and the element's full value are displayed.



pStat	ABC123	fStat	2
-----	-20	-----	
	5		
	11.78097		
	23.56184		
	-----		
ABC123(5) = 4.24264068711...			
<	>	NAME3	" DP S

- 2 Switch to edit-element context and edit the element in the entry line.
- 3 Enter the edited element. Any expression is evaluated and the value is stored to the current element.



-----			
ABC123(5) = 5*(6π)/4			
<	>	NAME3	" DP S

	20		
	11.78097		
	23.56184		
	-----		
ABC123(6) =			
<	>	NAME3	" DP S

## Deleting Elements from a List

To delete a single element from a list, move the cursor onto the element and press **DEL**. The element is deleted. You can clear all elements from a list in any of three ways.

- ◆ In the list editor, press **▲** to move the cursor onto a list name and press **CLEAR** **ENTER**.
- ◆ In the list editor, move the cursor onto each element, and then press **DEL** one by one.
- ◆ On the home screen or in the program editor, enter **0→dimL listName** to set the dimension of *listName* to **0** (A to Z Reference).

## Removing a List from the List Editor

To remove a list from the list editor, move the cursor onto the list name and then press **DEL**. The list is not deleted from memory; it is only removed from the list editor.

You can remove all user-created lists from the list editor and restore list names **xStat**, **yStat**, and **fStat** to columns **1**, **2**, and **3** in either of two ways.

- ◆ Use **SetLEdit** with no arguments (page 161).
- ◆ Reset all memory (Chapter 17). Resetting defaults does not affect the list editor.

## Using List Operations

The **LIST OPS (Operations) Menu** 2nd [LIST] [F5]

{	}	NAMES	EDIT	OPS	
dimL	sortA	sortD	min	max	▶
					▶
					▶
					▶

sum	prod	seq	li>vc	vc>li
-----	------	-----	-------	-------

Fill	aug	cSum	DeltaI	Sortx
------	-----	------	--------	-------

Sorty	Select	SetLE	Form	
-------	--------	-------	------	--

For all LIST OPS menu items except **Fill**( and sometimes **dimL**, a directly entered list (*{element1,element2,...}*) is valid for the list argument.

**SortA** and **SortD** sort complex lists based on magnitude (modulus).

**dimL** list

*#ofElements* → **dimL** listName

*#ofElements* → **dimL** listName

**sortA** list

**sortD** list

Returns the dimension of (or number of elements in) *list*

Creates *listName* as a list that is *#ofElements* in length; each element is a **0**

Redimensions an existing *listName*; previously entered elements within the new dimension remain; each new list element is a **0**; each element in the old list that is outside the new dimension is deleted

Sorts *list* elements in ascending order, from low to high values

Sorts *list* elements in descending order, from high to low values

For a complex list, **min** or **max** returns the smallest or largest magnitude (modulus).

**min**(*list*)

Returns the smallest element of a real or complex *list*

**max**(*list*)

Returns the largest element of a real or complex *list*

**sum** *list*

Returns the sum of all the elements of a real or complex *list*, adding from the last element to the first

**prod** *list*

Returns the product of all the elements of a real or complex *list*

**seq**(*expression,variable,begin,end[,step]*)

Returns a list in which each element is the result of the evaluation of *expression* with regard to *variable* for the values ranging from *begin* to *end* in intervals of *step* (*step* can be negative)

**li** $\rightarrow$ **vc** *list*

Converts a real or complex *list* to a vector

**li** $\rightarrow$ **vc** {*element1,element2,...*}

**vc** $\rightarrow$ **li** *vector*

Converts a real or complex *vector* to a list

**vc** $\rightarrow$ **li** [*element1,element2,...*]

**Fill**(*number,listName*)

Stores a real or complex *number* to every element of *listName*

**aug**(*listA,listB*)

(augment) Concatenates the real or complex elements of *listA* and *listB*

**cSum**(*list*)

Returns a list of the cumulative sums of real or complex *list* elements, starting with the first element and proceeding to the last

**Delta**list(*list*)

Returns a list containing the differences between consecutive elements for all elements in a real or complex *list*

**Sortx** [*ListName,ListName,frequencyListName*]

In ascending order of **x** elements, sorts *xListName* , sorts **x** and **y** data pairs, and optionally, their frequencies, in *xListName*, *yListName* , and *frequencyListName*; **xStat** and **yStat** are defaults

**Sorty** [*xListName,ListName,frequencyListName*]

In ascending order of **y** elements, sorts *xListName* , sorts **x** and **y** data pairs, and optionally, their frequencies, in *xListName*, *yListName* , and *frequencyListName*; **xStat** and **yStat** are defaults

Selecting **Delta**l from the menu pastes **Delta**list( to the cursor location.

For **Sortx** and **Sorty**, both lists must have the same number of elements.

Selecting **SetLE** from the menu pastes **SetLEdit** to the cursor location.

You can create new list names as **SetLEdit** arguments.

**Select**(*xListName*,  
*yListName*)

Selects one or more specific data points from a scatter plot or xyLine plot (only), then stores the selected data points to *xListName* and *yListName* (Chapter 14)

**SetLEdit** [*column1ListName*,  
*column2ListName*,...,  
*column20ListName*]

Sets up the list editor; **SetLEdit** with one to 20 *ListNames* loads them in the specified order; **SetLEdit** with no arguments removes all current list names from the list editor and enters the default lists **xStat**, **yStat**, and **fStat** to columns 1, 2, and 3

**Form**("formula",*listName*)

Attaches *formula* to *listName*; *formula* resolves to a list, which is dynamically stored and updated in *listName* (page 162)

## Using Mathematical Functions with Lists

You can use a list as a single argument for many TI-86 functions; the result is a list. The function must be valid for every element in the list; however, when graphing, undefined points do not result in an error.

When you use lists for two or more arguments in the same function, all lists must have the same number of elements (equal dimension). Here are some examples of a list as a single argument.

$\{1,2,3\}+10$  returns  $\{11\ 12\ 13\}$

$\{5,10,15\}*\{2,4,6\}$  returns  $\{10\ 40\ 90\}$

$3+\{1,7,(2,1)\}$  returns  $\{(4,0)\ (10,0)\ (5,1)\}$

$\sqrt{\{4,16,36,64\}}$  returns  $\{2\ 4\ 6\ 8\}$

$\sin\{7,5\}$  returns  $\{.656986598719\ -.958924274663\}$

$\{1,15,36\}<19$  returns  $\{1\ 1\ 0\}$

You cannot edit an element of a list created from an attached formula unless you first detach the formula from the list name.

When you include more than one list name in an attached formula, each list must have the same dimension.

Begin these steps on a blank line on the home screen.

## Attaching a Formula to a List Name

You can attach a formula to a list name so that the formula generates a list that is stored and dynamically updated in the list name.

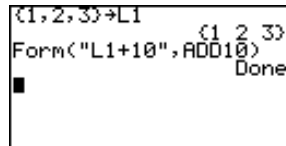
- ◆ When you edit an element of a list that is referenced in the formula, the corresponding element in the list to which the formula is attached is updated.
- ◆ When you edit the formula itself, all elements in the list to which the formula is attached are updated.

To attach a formula to a list name on the home screen or in the program editor, the syntax is:

**Form**("formula",listName)

When you enter a new list name as the second argument for **Form**( , the list name is created and stored in the LIST NAMES menu and VARS LIST screen upon execution.

- 1 Store elements to a list name. 2nd [LIST] [F1] 1 [.] 2 [.]  
3 [F2] [STO▶] [L] [ALPHA]  
1 [ENTER]
- 2 Select **Form** from the LIST OPS menu; **Form**( is pasted to the cursor location. [F5] [MORE] [MORE] [MORE]  
[F4]
- 3 Enter a formula in quotation marks. 2nd [STRNG] [F1] [ALPHA]  
[L] 1 [+] 10 [F1]
- 4 Enter a comma and then the list name to which you want to attach the formula. [,] [ALPHA] [ALPHA] [A]  
[D] [D] [ALPHA] 10 [.]
- 5 Attach the formula to the list name. [ENTER]



To view a formula attached to a list name, use the list editor (page 157).

## Comparing an Attached List with a Regular List

To see the differences between an attached list and a regular list, follow these steps. The example below builds on the example above for attaching a formula to a list. Notice that the formula in step 1 below is not attached to **LX** because it is not set off by quotation marks.

- 1 Generate a regular list by storing the expression **L1+10** to the list name **LX**.

[ALPHA] [L] 1 [+]   
 10 [STO▶] [L] [X]   
 [ENTER]

L1+10→LX	(11 12 13)
----------	------------

- 2 Change the second element in **L1** to **-8** and display the edited list.

[(-) 8 [STO▶] [L]   
 [ALPHA] 1 [↑] 2 [↓]   
 [2nd] [.] [ALPHA]   
 [L] 1 [ENTER]

L1+10→LX	(11 12 13)
-8→L1(2):L1	(1 -8 3)

- 3 Compare the elements of the regular list **LX** with **ADD10**, to which the formula **L1+10** is attached. Notice that element 2 of **LX** is unchanged. Meanwhile, element 2 of **ADD10** has been recalculated, since element 2 of **L1** has been edited.

[2nd] [LIST] [F3]   
 [F2] [ENTER] [F4]   
 [ENTER]

ADD10	(11 2 13)
LX	(11 12 13)

< > NAMES EDIT OPS   
 REC12 ADD10 L1 LX fStat

If other list names are stored on the LIST NAMES menu, pressing [F1] and [F3] may not paste **ADD10** and **LX** to the home screen as shown.

## Using the List Editor to Attach a Formula

- 1 Display the list editor.
- 2 Highlight the list name to which you want to attach the formula.
- 3 Enter the formula in quotation marks.

[2nd] [LIST] [F4]

↑ ↓

[F4] 4 [X] [F3] [F2]

[2nd] [F4]

xStat	yStat	fStat
-2	-----	-----
5		
6		
1		

yStat = "4\*xStat"   
 < > NAMES " OPS   
 fStat xStat yStat

In the example, only **fStat**, **xStat**, and **yStat** are on the LIST NAMES menu and **xStat**={-2,9,6,1,-7}.

The attached formula must be set off by quotation marks.

The list editor displays a formula-lock symbol next to each list name that has a formula attached to it.

- 4 Attach the formula and generate the list. ENTER
- ◆ The TI-86 calculates each list element.
- ◆ A lock symbol is displayed next to the list name to which the formula is attached.

xStat	yStat	fStat
-2	3	-----
0	2	-----
1	4	-----
-7	28	-----
-----		
yStat(1) = -8		
<span style="border: 1px solid black; padding: 2px;">&lt;</span> <span style="border: 1px solid black; padding: 2px;">&gt;</span> <span style="border: 1px solid black; padding: 2px;">NAMES</span> <span style="border: 1px solid black; padding: 2px;">"</span> <span style="border: 1px solid black; padding: 2px;">DPS</span>		

To edit an attached formula, press ENTER in step 3, and then edit the formula.

### Using the List Editor With Attached-Formula Lists

When you edit an element of a list referenced in an attached formula, the TI-86 updates the corresponding element in the list to which the formula is attached.

xStat	yStat	fStat
5	10	-----
10	20	-----
15	30	-----
20	40	-----
-----		
xStat(1) = -33		
<span style="border: 1px solid black; padding: 2px;">&lt;</span> <span style="border: 1px solid black; padding: 2px;">&gt;</span> <span style="border: 1px solid black; padding: 2px;">NAMES</span> <span style="border: 1px solid black; padding: 2px;">"</span> <span style="border: 1px solid black; padding: 2px;">DPS</span>		

xStat	yStat	fStat
-33	66	-----
10	20	-----
15	30	-----
20	40	-----
-----		
xStat(2) = 10		
<span style="border: 1px solid black; padding: 2px;">&lt;</span> <span style="border: 1px solid black; padding: 2px;">&gt;</span> <span style="border: 1px solid black; padding: 2px;">NAMES</span> <span style="border: 1px solid black; padding: 2px;">"</span> <span style="border: 1px solid black; padding: 2px;">DPS</span>		

When you edit or enter elements of a displayed list in any of the three current list editor columns while an attached-formula list also is displayed, the TI-86 takes slightly longer to execute the edit or entry. To reduce this effect, move lists with formulas off the current three-column display, either by scrolling columns to the left or right or by rearranging the list editor.

### Executing and Displaying Attached Formulas

An attached formula must resolve to a list upon execution. Some examples of formulas that resolve to a list are " $5 \times xStat$ ", " $seq(x,x,1,10)$ ", and " $\{3,5,-8,4\}^2/10$ ". Execution of the formula occurs when you attempt to display the list to which the formula is attached. Also, the formula is executed whenever a list referenced by the formula is modified — whether on the home screen, in the list editor, or in a program.



You can successfully attach to a list a formula that does not yet resolve to a list. For example, you can attach "**5\*xStat**" to the list name **BY5** with no elements stored to **xStat**. However, if you attempt to display **BY5** when **xStat** has no elements, an error occurs.

When you attach such a formula to a list name in the list editor, the formula is successfully attached, but an error occurs. This is because the list editor attempts to execute the formula immediately after attaching it to the list name.

To view the list editor again, you must return to the home screen and either enter something to cause the formula to resolve to a list or remove the attached-formula list from the list editor using the LIST OPS menu item **SetLE** (page 161).

### Handling Errors Related to Attached Formulas

*All elements of a list referenced by an attached formula must be valid for the attached formula.*

On the home screen, you can attach to a list a formula that references another list that has no elements (dimension is **0**; page 161). However, you cannot display the attached-formula list in the list editor or on the home screen until you enter at least one element to the list that the formula references.

**Tip:** If an error menu is returned when you attempt to display an attached-formula list in the list editor, you can select **GOTO**, write down the formula that is attached to the list name, and then press **CLEAR** **ENTER** to detach (clear) the formula. Then you can use the list editor to find the source of the error. After making the appropriate changes, you can re-attach the formula to the list name.

If you do not want to clear the formula, you can select **QUIT**, display the referenced list on the home screen, and find and edit the source of the error. To edit an element of a list on the home screen, store the new value to *listName(element#)* (page 155).

### Detaching a Formula from a List Name

You can detach a formula in any of five ways.

- ◆ Use **dimL** to change the dimension of the list (page 159).
- ◆ Use *value*→*listName*(*element*→) to store *value* to an attached-formula list element.
- ◆ Use ""→*listName*, where *listName* is the attached-formula list.
- ◆ In the list editor, move the cursor onto the name of the attached-formula list, and then press **[ENTER]** **[CLEAR]** **[ENTER]**. All list elements remain, but the formula is detached and the lock symbol disappears.
- ◆ In the list editor, move the cursor onto an element of the attached-formula list. Press **[ENTER]**, edit the element, and then press **[ENTER]**. The element changes, the formula is detached, and the lock symbol disappears. All other list elements remain.

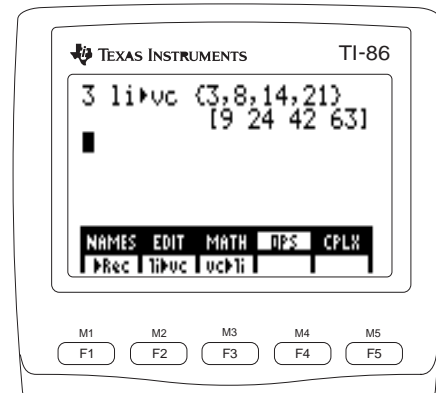
### Editing an Element of a Attached-Formula List

As described above, one way to detach a formula from a list name is to edit an element of the attached-formula list. The TI-86 protects against inadvertently detaching the formula from the list name when you move the cursor onto one of the elements.

Because of the protection feature, you must press **[ENTER]** before you can edit an element of an attached-formula list. The protection feature prevents you from deleting an element of an attached-formula list. To delete an element of a attached-formula list, you must first detach the formula in any of the ways described above.

# 12 Vectors

Vectors on the TI-86 .....	168
Creating, Storing, and Displaying Vectors.....	169
Using Mathematical Functions with Vectors.....	176



## Vectors on the TI-86

A vector is a one-dimensional array, arranged in either one row or one column. The vector elements can be real or complex. You can create, display, and edit vectors on the home screen or in the vector editor. When you create a vector, the elements are stored to the vector name.

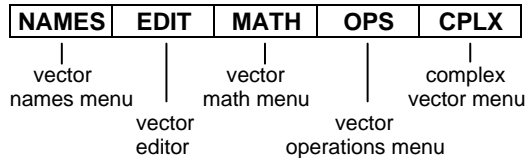
The TI-86 vector editor displays a vector vertically. On the home screen, a vector is entered and displayed horizontally. When you use a vector in an expression, the TI-86 automatically interprets the vector in the form (row vector or column vector) that is appropriate for the expression. For example, a column vector is appropriate for the expression  $matrix * vector$ .

On the TI-86, you can store up to 255 elements to a vector in rectangular form. You can use two- or three-element vectors to define magnitude and direction in a two- or three-dimensional space. You can express two- or three-element vectors in different forms, depending on the type of vector.

To express a...	You enter:	And the TI-86 returns:
Two-element rectangular vector	$[x,y]$	$[x\ y]$
Two-element cylindrical vector	$[r\angle\theta]$	$[r\angle\theta]$
Two-element spherical vector	$[r\angle\theta]$	$[r\angle\theta]$
Three-element rectangular vector	$[x,y,z]$	$[x\ y\ z]$
Three-element cylindrical vector	$[r\angle\theta,z]$	$[r\angle\theta\ z]$
Three-element spherical vector	$[r\angle\theta\angle\phi]$	$[r\angle\theta\angle\phi]$

## Creating, Storing, and Displaying Vectors

### The VECTR (Vector) Menu $\boxed{2\text{nd}} \boxed{[\text{VECTR}]}$

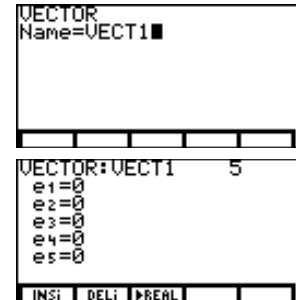


### The VECTR NAMES Menu $\boxed{2\text{nd}} \boxed{[\text{VECTR}]} \boxed{[\text{F1}]}$

The VECTR NAMES menu contains all currently stored vector names in alphanumeric order. To paste a vector name to the current cursor location, select it from the menu.

### Creating a Vector in the Vector Editor $\boxed{2\text{nd}} \boxed{[\text{VECTR}]} \boxed{[\text{F2}]}$

- ① Display the vector **Name=** prompt screen.  $\boxed{2\text{nd}} \boxed{[\text{VECTR}]} \boxed{[\text{F2}]}$
- ② ALPHA-lock is on. The VECTR NAMES menu is displayed. Enter a name from one to eight characters long, starting with a letter.  $\boxed{[\text{V}]} \boxed{[\text{E}]} \boxed{[\text{C}]} \boxed{[\text{T}]}$   
 $\boxed{[\text{ALPHA}]} \boxed{1}$
- ③ Display the vector editor. The vector editor menu also is displayed.  $\boxed{[\text{ENTER}]}$
- ④ Accept or change the vector *elements* dimension with an integer  $\geq 1$  and  $\leq 255$ . The vector is displayed; all elements are **0**.  $\boxed{5} \boxed{[\text{ENTER}]}$



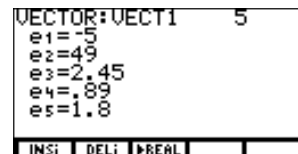
The TI-86 distinguishes between uppercase and lowercase letters in vector names. For example, **VECT1**, **Vect1**, and **vect1** are three different vector names.

↓ or ↑ in the first column indicates additional vector elements.

You can enter an expression at a vector element prompt.

- 5 Enter each vector element value at each vector element prompt. You can enter expressions. To move to the next prompt, press **ENTER** or **↓**. The vector elements are stored to **VECT1**, which becomes a VECTR NAMES menu item.

**(-) 5 ↓ 49**  
**↓ 2 . 45 ↓**  
**□ 89 ↓ 1 □ 8**



**The Vector Editor Menu** **2nd** [VECTR] **F2** *vectorName***ENTER**

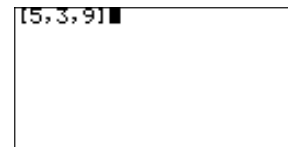
INSi	DELi	▶REAL		
------	------	-------	--	--

- INSi** Inserts a blank element (**e<sub>n</sub>=**) prompt at the cursor location; shifts current elements down
- DELi** Deletes the element from the cursor location and from the vector; shifts elements up
- ▶REAL** Converts the displayed complex number vector to a real number vector

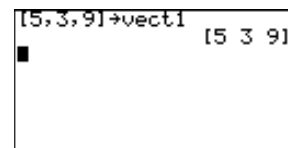
**Creating a Vector on the Home Screen**

- 1 Define the beginning of the vector with **[**.
- 2 Enter each vector element, separating each from the next with a comma.
- 3 Define the end of the vector with **]**.
- 4 Store the vector to a vector name from one to eight characters long, starting with a letter. The vector is displayed horizontally and the vector name becomes a VECTR NAMES menu item.

**2nd** [**]**  
**5 □ 3 □ 9**  
**2nd** [**]**



**STO▶** **2nd** [**alpha**] [**V**]  
**[E]** **[C]** **[T]** **ALPHA**  
**ALPHA** **1** **ENTER**



To delete a vector name from memory, use the MEM DELETE:VECTR screen (Chapter 17).

### Creating a Complex Vector

If any element of a vector is complex, all elements of the vector are displayed as complex. For example, when you enter the vector  $[1, 2, (3, 1)]$ , the TI-86 displays  $[(1, 0) (2, 0) (3, 1)]$ .

To create a complex vector from two real vectors, the syntax is:

*realVector*+(0,1)*imaginaryVector*→*complexVectorName*

*realVector* contains the real part of each element and *imaginaryVector* contains the imaginary part.

### Displaying a Vector

To display a vector, paste the vector name to the home screen, and then press  $\boxed{\text{ENTER}}$ .

To display a specific element of *vectorName* on the home screen or in a program, the syntax is:

*vectorName*(*element*#)

Real two- and three-element vector results are displayed according to the current vector mode setting: **RectV**, **CylV**, or **SphereV** (Chapter 1). You can select a vector conversion instruction from the VECTR OPS menu to override the mode setting (page 173).

Complex vectors are displayed in rectangular form only.

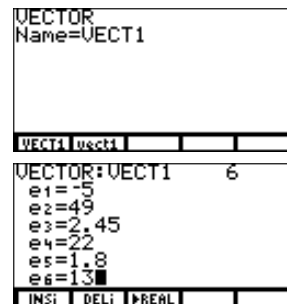
When you execute the expression, the answer is displayed as a vector.

### Using a Vector in an Expression

- ◆ You can enter the vector directly (for example,  $35-[5,10,15]$ ).
- ◆ You can use  $\boxed{\text{ALPHA}}$  and  $\boxed{2\text{nd}} \boxed{[\text{alpha}]}$  to enter a vector name's individual characters.
- ◆ You can select the vector name from the VECTR NAMES menu ( $\boxed{2\text{nd}} \boxed{[\text{VECTR}]} \boxed{[\text{F1}]}$ ).
- ◆ You can select the vector name from the VARS VECTR screen ( $\boxed{2\text{nd}} \boxed{[\text{CATLG-VARS}]} \boxed{[\text{MORE}]} \boxed{[\text{F1}]}$ ).

### Editing Vector Dimension and Elements

- 1 Display the vector **Name=** prompt screen.  $\boxed{2\text{nd}} \boxed{[\text{VECTR}]} \boxed{[\text{F2}]}$
- 2 Enter the vector name. Either select it from the VECTR NAMES menu or enter the characters.  $\boxed{[\text{F1}]}$
- 3 Display the vector editor.  $\boxed{[\text{ENTER}]}$
- 4 Change or accept the vector dimension.  $\boxed{6} \boxed{[\text{ENTER}]}$
- 5 Move the cursor to any element and edit it.  $\boxed{\downarrow} \boxed{\downarrow} \boxed{\downarrow} \boxed{22}$   
Continue moving the cursor to other elements.  $\boxed{\downarrow} \boxed{\downarrow} \boxed{13}$
- 6 Save the changes and exit the vector editor.  $\boxed{[\text{EXIT}]}$



You can use  $\boxed{[\text{CLEAR}]}$ ,  $\boxed{[\text{DEL}]}$ , and  $\boxed{2\text{nd}} \boxed{[\text{INS}]}$  to edit matrix elements. You also can overwrite existing characters.

To use  $\boxed{[\text{STO} \rightarrow]}$  to change an element value on the home screen, the syntax is:  
*value*  $\rightarrow$  *vectorName*(*element#*)



### The VECTR MATH Menu 2nd [VECTR] F3

NAMES	EDIT	MATH	OPS	CPLX
cross	unitV	norm	dot	

**cross**(*vectorA*,*vectorB*) Returns the cross product of *vectorA* and *vectorB*, both of which are real or complex two-element or three-element vectors; expressed with variables, **cross**([*a,b,c*],[*d,e,f*]) returns [**bf-ce cd-af ae-bd**]

**unitV** *vector* Returns a unit vector where each element of a real or complex *vector* is divided by the vector norm

**norm** *vector* Returns the Frobenius norm ( $\sqrt{\sum(\text{real}^2 + \text{imaginary}^2)}$ ) where the sum is over all elements of a real or complex *vector*

**dot**(*vectorA*,*vectorB*) Returns the dot product of *vectorA* and *vectorB*, both of which are real or complex vectors; expressed with variables, **dot**([*a,b,c*],[*d,e,f*]) returns **ad+be+cf**

### The VECTR OPS (Operations) Menu 2nd [VECTR] F4

NAMES	EDIT	MATH	OPS	CPLX
dim	Fill	►Pol	►Cyl	►Sph
►				
		►Rec	li►vc	vcl►li

**dim** *vector* Returns the dimension of (or number of elements in) *vector*

*#ofElements*►**dim***vectorName* Creates a new *vectorName* of the specified length (*#ofElements*); each element is 0

*#ofElements*►**dim***vectorName* Redimensions *vectorName* to the specified length (*#ofElements*)

**Fill**(*number*,*vectorName*) Stores a real or complex *number* to every element in *vectorName*

Press STO► to enter the → symbol after *#ofElements*.

For the conversion functions below, the three-element vector conversion equations for cylindrical form  $[r \ \theta \ z]$  are:

$$x = r \cos\theta \quad y = r \sin\theta \quad z = z$$

The three-element vector conversion equations for spherical form  $[r \ \theta \ \phi]$  are:

$$x = r \cos\theta \sin\phi \quad y = r \sin\theta \sin\phi \quad z = r \cos\phi$$

<i>vector</i> → <b>Pol</b>	Displays a 2-element <i>vector</i> in polar form $[r \angle \theta]$
<i>vector</i> → <b>Cyl</b>	Displays a 2- or 3-element <i>vector</i> as a cylindrical vector $[r \angle \theta \ \mathbf{0}]$ or $[r \angle \theta \ z]$
<i>vector</i> → <b>Sph</b>	Displays a 2- or 3-element <i>vector</i> as a spherical vector $[r \angle \theta \ \mathbf{0}]$ or $[r \angle \theta \ \phi]$
<i>complexVector</i> → <b>Rec</b>	Displays a 2- or 3-element <i>complexVector</i> in rectangular form $[x \ y]$ or $[x \ y \ z]$
<b>li</b> → <i>vc</i> <i>list</i>	Converts a real or complex <i>list</i> into a vector
<b>vc</b> → <i>li</i> <i>vector</i>	Converts a real or complex <i>vector</i> into a list

*Complex elements are valid only for li→vc and vc→li.*

**The VECTR CPLX (Complex) Menu**     $\boxed{2\text{nd}}$  [VECTR]  $\boxed{F5}$ 

NAMES	EDIT	MATH	OPS	CPLX
conj	real	imag	abs	angle

- conj** *complexVector* Returns a vector in which each element is the complex conjugate of the corresponding element of a *complexVector*
- real** *complexVector* Returns a real vector in which each element is the real portion of the corresponding element of a *complexVector*
- imag** *complexVector* Returns a real vector in which each element is the imaginary portion of the corresponding element of a *complexVector*
- abs** *Vector* Returns a real vector in which each element is either the absolute value of the corresponding element of a real *vector* or the magnitude (modulus) of the corresponding element of a *complexVector*
- angle** *complexVector* Returns a real vector in which each element is either **0** if the element of *complexVector* is real or the polar angle if the element of *complexVector* is complex; polar angles are calculated as  $\tan^{-1}(\text{complex}/\text{real})$  adjusted by  $+\pi$  in the second quadrant and by  $-\pi$  in the third quadrant

## Using Mathematical Functions with Vectors

To add or subtract two vectors, the dimension of *vectorA* must equal the dimension of *vectorB*.

You cannot multiply two vectors or divide one vector by another vector.

**==** and **≠** are on the TEST menu.

**round**, **iPart**, **fPart**, and **int** are on the MATH NUM menu.

*vectorA*+*vectorB*

*vectorA*-*vectorB*

*vector*\**number* or  
*number*\**vector*

*matrix*\**vector*

*vector*/*number*

-*vector*

*vectorA*==*vectorB*

*vectorA*≠*vectorB*

**round**(*vector*[,*#ofDecimals*])

**iPart** *vector*

**fPart** *vector*

**int** *vector*

Adds each *vectorA* element to the corresponding *vectorB* element; returns a vector of the sums

Subtracts each *vectorB* element from the corresponding *vectorA* element; returns a vector of the differences

Returns a vector that is the product of a real or complex *number* times each element in a real or complex *vector*

Returns a vector that is the product of each *vector* element times each *matrix* element; *matrix* column dimension and *vector* dimension must be equal

Returns a vector that is the quotient of each real or complex *vector* element divided by a real or complex *number*

(negation) Changes the sign of each *vector* element

Returns **1** if every corresponding element comparison is true; returns **0** if any is false

Returns **1** if at least one corresponding element comparison is false

Rounds each *vector* element to 12 digits, or rounds to specified *#ofDecimals*

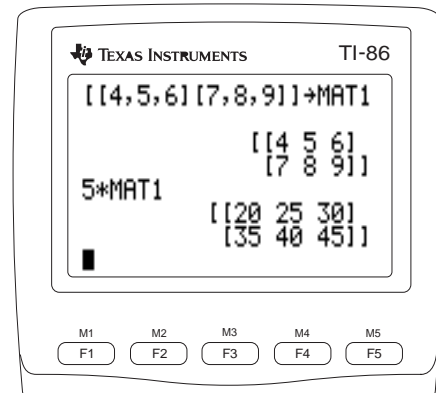
Returns the integer part of each real or complex *vector* element

Returns the fractional part of each real or complex *vector* element

Returns the greatest integer of each real or complex *vector* element

# 13 Matrices

Matrices on the TI-86 ..... 178  
Creating, Storing, and Displaying Matrices..... 178  
Using Mathematical Functions with Matrices..... 185

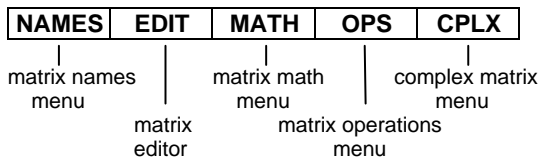


## Matrices on the TI-86

A matrix is a two-dimensional array, arranged in rows and columns. The matrix elements can be real or complex. You can create, display, and edit matrices on the home screen or in the matrix editor. When you create a matrix, the elements are stored to the matrix name.

### Creating, Storing, and Displaying Matrices

#### The **MATRIX** (Matrix) Menu $\text{[2nd] [MATRIX]}$

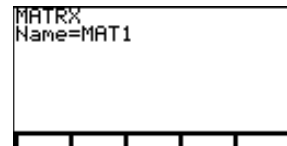


#### The **MATRIX NAMES** Menu $\text{[2nd] [MATRIX] [F1]}$

The **MATRIX NAMES** menu contains all currently stored matrix names in alphanumeric order. To paste a matrix name to the current cursor location, select it from the menu.

#### Creating a Matrix in the Matrix Editor $\text{[2nd] [MATRIX] [F2]}$

- 1 Display the matrix **Name=** prompt screen.  $\text{[2nd] [MATRIX] [F2]}$
- 2 ALPHA-lock is on. The **MATRIX NAMES** menu is displayed. Enter a name from one to eight characters long, starting with a letter.  $\text{[M] [A] [T] [ALPHA] 1}$



*The TI-86 distinguishes between uppercase and lowercase letters in matrix names. For example, **MAT1** and **mat1** are two different vector names.*

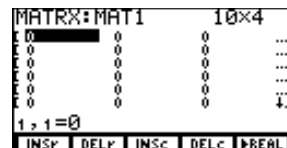
An ellipsis (...) at either end of matrix rows indicates additional columns.

↓ or ↑ in the last column indicates additional rows.

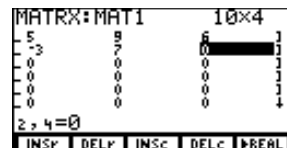
- ③ Display the matrix editor and the matrix editor menu.
- ④ Accept or change the matrix dimensions (row × column) in the top-right corner of the screen, ( $1 \leq \text{row} \leq 255$  and  $1 \leq \text{column} \leq 255$ ); maximum combination is subject to memory availability. The matrix is displayed; all elements are 0.
- ⑤ Enter each matrix element value at the element prompt (1,1= for row 1, column 1). You can enter expressions. To move to the next element, press **ENTER**. To move to the next row, press **↓**.

**ENTER**

**10** **ENTER** **4** **ENTER**



**(←)** **4** **ENTER** **5**  
**ENTER** **9** **ENTER** **6**  
**ENTER** **1** **ENTER**  
**(←)** **3** **ENTER** **7**  
**ENTER** and so on



**The Matrix Editor Menu** **2nd** **[MATRIX]** **F2** *matrixName* **ENTER**

<b>INSr</b>	<b>DELr</b>	<b>INSc</b>	<b>DELC</b>	<b>▶REAL</b>
-------------	-------------	-------------	-------------	--------------

- INSr** Inserts a row at the cursor location; shifts subsequent rows down
- DELr** Deletes row at the cursor location; shifts subsequent rows up
- INSc** Inserts a column at the cursor location; shifts subsequent columns to the right
- DELC** Deletes the column at the cursor location; shifts subsequent columns to the left
- ▶REAL** Converts the displayed complex number matrix to a real number matrix

## Creating a Matrix on the Home Screen

- 1 Define the start of the matrix with [, and then define the start of the first row with another [. Enter each element for the row, separating them with commas. Define the end of the first row with ].

`2nd` `[` `[` `2nd` `[`  
`2` `,` `4` `,` `6` `,`  
`8` `2nd` `]`

```
[[2,4,6,8]
```

- 2 Define the start of each subsequent row with [. Enter the row elements, separating each from the next with a comma. Define the end of each row with ]. Then define the end of the matrix with ].

`2nd` `[` `(-)` `1` `,`  
`(-)` `3` `,` `(-)` `5` `,`  
`(-)` `7` `2nd` `[` `]`  
`2nd` `]`

```
[[2,4,6,8][-1,-3,-5,-7]]
```

- 3 Store the matrix to a matrix name. Either enter a name from one to eight characters long, starting with a letter, or select a name from the MATRX NAMES menu. The matrix is displayed. If newly created, the matrix name becomes a MATRX NAMES menu item.

`STO` `2nd` `[alpha]`  
`[M]` `[A]` `[T]`  
`ALPHA` `ALPHA` `1`  
`ENTER`

```
[[2,4,6,8][-1,-3,-5,-7]]→mat1
[[2 4 6 8]
 [-1 -3 -5 -7]]
```

The close bracket is not necessary when it precedes `STO`.

To delete a matrix name from memory, use the MEM DELETE:MATRX screen (Chapter 17).

## Creating a Complex Matrix

If any matrix element is complex, all elements of the matrix are displayed as complex. For example, when you enter the matrix `[[1,2][5,(3,1)]]`, the TI-86 displays `[[1,0)(2,0)][5,0)(3,1)]]`. To create a complex matrix from two real matrices with the same dimensions, the syntax is: *realMatrix*+**(0,1)***imaginaryMatrix*→*complexMatrixName*

*realMatrix* contains the real part of each element and *imaginaryMatrix* contains the imaginary part of each element.



To view elements beyond the current screen, use  $\downarrow$ ,  $\leftarrow$ ,  $\uparrow$ , and  $\rightarrow$ .

### Displaying Matrix Elements, Rows, and Submatrices

To display an existing matrix on the home screen, enter the matrix name's individual characters or select it from the MATRX NAMES menu, and then press  $\text{ENTER}$ . The full value of each element is displayed. Elements with very large values may be expressed exponentially.

To display specific elements of *matrixName*, the syntax is:  
*matrixName*(row,column)

To display a row of *matrixName*, the syntax is:  
*matrixName*(row)

To display a submatrix of *matrixName*, the syntax is:  
*matrixName*(beginRow,beginColumn,endRow,endColumn)

```

[[[-4 5 9 6]
 [1 -3 7 0]
 [0 0 0 0]
 [0 0 0 0]
 [0 0 0 0]
 [0 0 0 0]]↓

```

```

MAT1(2,2)                -3

```

```

MAT1(2)                   [1 -3 7 0]

```

```

MAT1(1,2,2,3)            [[5 9]
                          [-3 7]]

```

### Using a Matrix in an Expression

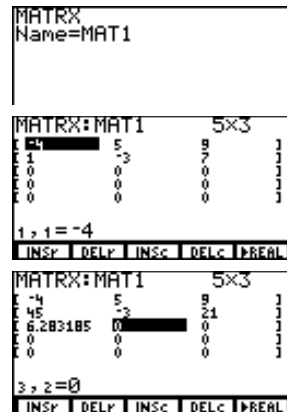
- ◆ You can enter the matrix directly (for example,  $5*[[2,3][3,5]]$ ).
- ◆ You can use  $\text{ALPHA}$  and  $\text{2nd}$  [alpha] to enter a matrix name's individual characters (for example,  $\text{MAT1} * \mathbf{3}$ ).
- ◆ You can select the matrix name from the MATRX NAMES menu ( $\text{2nd}$  [MATRX] [F1]).
- ◆ You can select the matrix name from the VARS MATRX screen ( $\text{2nd}$  [CATLG-VARS] [MORE] [F2]).

When you execute the expression, the answer is displayed as a matrix.

## Editing Matrices in the Matrix Editor

- 1 Display the matrix **Name=** prompt screen.  $\boxed{2\text{nd}} \boxed{[\text{MATRX}]} \boxed{[\text{F2}]}$
- 2 Enter the matrix name. Either select it from the **MATRX NAMES** menu or enter the characters.  $\boxed{[\text{M}]}\boxed{[\text{A}]}\boxed{[\text{T}]}$   
 $\boxed{[\text{ALPHA}]} \boxed{1}$
- 3 Display the matrix editor.  $\boxed{[\text{ENTER}]}$
- 4 Edit or accept the row dimension, and then edit or accept the column dimension.  $\boxed{5} \boxed{[\text{DEL}]} \boxed{[\text{ENTER}]}$   
 $\boxed{3} \boxed{[\text{ENTER}]}$
- 5 Move the cursor to any element and edit it.  $\boxed{\downarrow} \boxed{45} \boxed{[\text{ENTER}]} \boxed{\rightarrow}$   
 $\boxed{21} \boxed{[\text{ENTER}]} \boxed{2} \boxed{2\text{nd}}$   
 $\boxed{[\pi]} \boxed{[\text{ENTER}]}$
- 6 Save the changes and leave the matrix editor.  $\boxed{[\text{EXIT}]}$

You can use  $\boxed{[\text{CLEAR}]}$ ,  $\boxed{[\text{DEL}]}$ , and  $\boxed{2\text{nd}} \boxed{[\text{INS}]}$  to edit matrix elements. You also can overwrite existing characters.



## Editing Matrices on the Home Screen

To change a matrix element value, the syntax is:

*value*  $\rightarrow$  *matrixName*(*row,column*)

To change the values of an entire row of elements, the syntax is:

[*valueA,valueB,...,value n*]  $\rightarrow$  *matrixName*(*row*)

To change the values of part of a row, beginning at a specified column, the syntax is:

[*valueA,valueB,...,value n*]  $\rightarrow$  *matrixName*(*row,beginColumn*)

To change the values of a submatrix within *matrixName*, the syntax is:

[[*valueA,...,value n*] ... [*valueA,...,value n*]]  $\rightarrow$  *matrixName*(*beginRow,beginColumn*)

The **MATRIX MATH Menu** 2nd [MATRIX] F3

NAMES	EDIT	MATH	OPS	CPLX						
det	T	norm	eigVl	eigVc	▶	rnorm	cnorm	LU	cond	

<b>det</b> <i>squareMatrix</i>	Returns the determinant of <i>squareMatrix</i>
<i>matrix</i> <sup>T</sup>	Returns a transposed matrix; each element's ( <i>row,column</i> ) coordinates switch
<b>norm</b> <i>matrix</i>	Returns the Frobenius norm ( $\sqrt{\sum(\text{real}^2 + \text{imaginary}^2)}$ ) where the sum is over all elements of a real or complex <i>matrix</i>
<b>eigVl</b> <i>squareMatrix</i>	Returns a list of the normalized eigenvalues of a real or complex <i>squareMatrix</i>
<b>eigVc</b> <i>squareMatrix</i>	Returns a matrix containing the eigenvectors for a real or complex <i>squareMatrix</i> ; each column corresponds to an eigenvalue
<b>rnorm</b> <i>matrix</i>	(row norm) Returns the largest of the sums of the absolute values of the elements (magnitudes of complex elements) in each row of <i>matrix</i>
<b>cnorm</b> <i>Matrix</i>	(column norm) Returns the largest of the sums of the absolute values of the elements (magnitudes of complex elements) in each column of <i>matrix</i>
<b>LU</b> ( <i>matrix</i> , <i>lMatrixName</i> , <i>uMatrixName</i> , <i>pMatrixName</i> )	Calculates the Crout LU (lower-upper) decomposition of a real or complex <i>matrix</i> ; stores the lower triangular matrix to <i>lMatrixName</i> , the upper triangular matrix to <i>uMatrixName</i> , and the permutation matrix (which describes the row swaps done during calculation) in <i>pMatrixName</i>
<b>cond</b> <i>squareMatrix</i>	Calculates <b>cnorm</b> <i>squareMatrix</i> * <b>cnorm</b> <i>squareMatrix</i> <sup>-1</sup> ; the closer the product is to 1, the more stable <i>squareMatrix</i> can be expected to be in matrix functions

The MATRX OPS (Operations) Menu 2nd [MATRX] F4

NAMES	EDIT	MATH	OPS	CPLX	
dim	Fill	ident	ref	rref	▶ <span style="border: 1px solid black; padding: 2px;">aug</span> <span style="border: 1px solid black; padding: 2px;">rSwap</span> <span style="border: 1px solid black; padding: 2px;">rAdd</span> <span style="border: 1px solid black; padding: 2px;">multR</span> <span style="border: 1px solid black; padding: 2px;">mRAdd</span>
					▶ <span style="border: 1px solid black; padding: 2px;">randM</span> <span style="border: 1px solid black; padding: 2px;"></span> <span style="border: 1px solid black; padding: 2px;"></span> <span style="border: 1px solid black; padding: 2px;"></span> <span style="border: 1px solid black; padding: 2px;"></span>

Press STO▶ to enter the  $\rightarrow$  symbol after the close brace.

<b>dim</b> <i>matrix</i>	Returns the dimensions of <i>matrix</i> as a list {rows columns}
<b>{rows,columns}</b> → <b>dim</b> <i>matrixName</i>	Creates a new <i>matrixName</i> of the specified dimensions; each element is 0
<b>{rows,columns}</b> → <b>dim</b> <i>matrixName</i>	Redimensions <i>matrixName</i> to the specified dimensions
<b>Fill</b> ( <i>number,matrixName</i> )	Stores a real or complex <i>number</i> to each <i>matrixName</i> element
<b>ident</b> <i>dimension</i>	Returns the square identity matrix of <i>dimension</i> × <i>dimension</i>
<b>ref</b> <i>matrix</i>	Returns the row-echelon form of <i>matrix</i>
<b>rref</b> <i>matrix</i>	Returns the reduced row-echelon form of <i>matrix</i>
<b>aug</b> ( <i>matrixA,matrixB</i> )	Concatenates <i>matrixA</i> and <i>matrixB</i>
<b>aug</b> ( <i>matrix,vector</i> )	Concatenates <i>matrix</i> and <i>vector</i>
<b>rSwap</b> ( <i>matrix,rowA,rowB</i> )	Returns a matrix after swapping <i>rowA</i> and <i>rowB</i> of <i>matrix</i>
<b>rAdd</b> ( <i>matrix,rowA,rowB</i> )	Returns <i>matrix</i> with ( <i>rowA+rowB</i> ) of <i>matrix</i> stored in <i>rowB</i>
<b>multR</b> ( <i>number,matrix,row</i> )	Returns <i>matrix</i> with ( <i>row*number</i> ) stored in <i>row</i>
<b>mRAdd</b> ( <i>number,matrix,rowA,rowB</i> )	Returns <i>matrix</i> with (( <i>rowA*number</i> )+ <i>rowB</i> ) stored in <i>rowB</i>
<b>randM</b> ( <i>rows,columns</i> )	Creates a matrix of specified dimensions with random elements

When you use **aug**(, the number of rows in *matrixA* must equal the number of rows in *matrixB* or the number of elements in *vector*.

Elements of matrices created with **randM**( are integers  $\geq -9$  and  $\leq 9$ .

### The **MATRIX CPLX (Complex ) Menu** 2nd [MATRIX] [F5]

NAMES	EDIT	MATH	OPS	CPLX
conj	real	imag	abs	angle

<b>conj</b> <i>complexMatrix</i>	Returns a matrix in which each element is the complex conjugate of the corresponding element of a <i>complexMatrix</i>
<b>real</b> <i>complexMatrix</i>	Returns a real matrix in which each element is the real portion of the corresponding element of a <i>complexMatrix</i>
<b>imag</b> <i>complexMatrix</i>	Returns a real matrix in which each element is the imaginary portion of the corresponding element of a <i>complexMatrix</i>
<b>abs</b> <i>matrix</i>	Returns a real matrix in which each element is either the absolute value of the corresponding element of a real <i>matrix</i> or the magnitude (modulus) of the corresponding element of a complex <i>matrix</i>
<b>angle</b> <i>complexMatrix</i>	Returns a real matrix in which each element is either <b>0</b> if the element of <i>complexMatrix</i> is real or the polar angle if the element of <i>complexMatrix</i> is complex; the polar angles are calculated as $\tan^{-1}(\text{imaginary} / \text{real})$ adjusted by $+\pi$ in the second quadrant and by $-\pi$ in the third quadrant

## Using Mathematical Functions with Matrices

To add or subtract two matrices, the dimensions of *matrixA* must equal the dimensions of *matrixB*.

<i>matrixA</i> + <i>matrixB</i>	Adds each <i>matrixA</i> element to the corresponding <i>matrixB</i> element; returns a matrix of the sums
<i>matrixA</i> - <i>matrixB</i>	Subtracts each <i>matrixB</i> element from the corresponding <i>matrixA</i> element; returns a matrix of the differences

To multiply two matrices, the column dimension of *matrixA* must equal the row dimension of *matrixB*.

To enter  $x^{-1}$ , press  $\boxed{2\text{nd}} \boxed{[x^{-1}]}$ . Do not use  $\boxed{\text{x-VAR}} \boxed{\wedge} \boxed{(-)} \boxed{1}$ .

$e^x$ , **sin**, and **cos** do not return the exponential, sine, or cosine of each matrix element.

To make relational comparisons, *matrixA* and *matrixB* must have equal dimensions.

**==** and **≠** are on the TEST menu.

**round**, **iPart**, **fPart**, and **int** are on the MATH NUM menu.

*matrixA*\**matrixB* or  
*matrixB*\**matrixA*

*matrix*\**number* or  
*number*\**matrix*

*matrix*\**vector*

-*matrix*

*squareMatrix*<sup>-1</sup>

*matrix*<sup>2</sup>

*squareMatrix*<sup>power</sup>

**e<sup>x</sup>** *squareMatrix*

**sin** *squareMatrix*

**cos** *squareMatrix*

*matrixA*==*matrixB*

*matrixA*≠*matrixB*

**round**(*matrix*[,*#ofDecimals*])

**iPart** *matrix*

**fPart** *matrix*

**int** *matrix*

Multiplies *matrixA* and *matrixB*; returns a square matrix of the products

Returns a matrix that is the product of a real or complex *number* times each element in a real or complex *matrix*

Returns a vector that is the product of each *vector* element times each *matrix* element; the *matrix* column dimension and *vector* dimension must be equal

(negation) Changes the sign of each element in *matrix*

Returns the inverse of *squareMatrix* (not the inverse of each element)

Squares a square matrix

Raises a *squareMatrix* to the designated *power*

Returns the square matrix exponential of a real *squareMatrix*

Returns the square matrix sine of a real *squareMatrix*

Returns the square matrix cosine of a real *squareMatrix*

Returns **1** if every corresponding element comparison is true; returns **0** if any is false

Returns **1** if at least one corresponding element comparison is false

Rounds each *matrix* element to 12 digits or to specified *#ofDecimals*

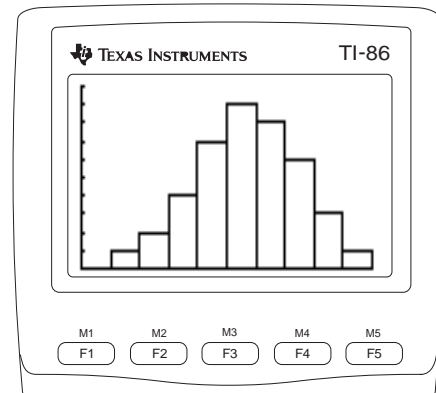
Returns the integer part of each element of a real or complex *matrix*

Returns the fractional part of each element of a real or complex *matrix*

Returns the greatest integer of each element of a real or complex *matrix*

# 14 Statistics

Statistical Analysis on the TI-86 .....	188
Setting Up a Statistical Analysis.....	188
Results of a Statistical Analysis.....	192
Plotting Statistical Data .....	194
The STAT DRAW Menu .....	199
Forecasting a Statistical Data Value .....	199



## Statistical Analysis on the TI-86

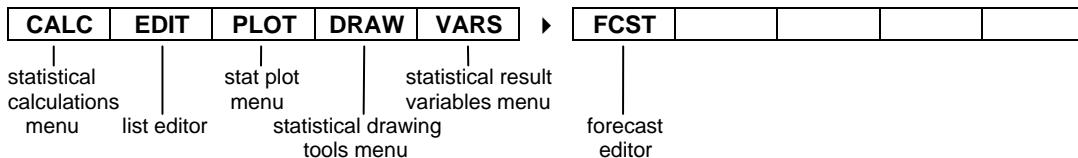
With the TI-86, you can analyze one-variable and two-variable statistical data, which are stored in lists. One-variable data has one measured variable. Two-variable data has pairs comprising an independent variable and a dependent variable.

When analyzing either kind of data, you can specify a frequency of occurrence for the independent variable values. These specified frequencies must be real numbers  $\geq 0$ .

## Setting Up a Statistical Analysis

- ❶ Enter the statistical data into one or more lists (Chapter 11).
- ❷ Calculate the statistical variables or fit a model to the data.
- ❸ Plot the data.
- ❹ Graph the regression equation for the plotted data.

### The STAT (Statistics) Menu 2nd STAT



The same list editor is displayed, whether you press 2nd STAT F2 or 2nd LIST F4. For a description of the list editor, see Chapter 11.



## Entering Statistical Data

Data for statistical analysis is stored in lists, which you can create and edit in the list editor (Chapter 11), on the home screen (Chapter 11), or in a program (Chapter 16). The TI-86 has three built-in list names for statistics, **xStat** (x-variable list), **yStat** (y-variable list), and **fStat** (frequency list). TI-86 statistical functions use these lists as defaults.

### The LIST NAMES Menu 2nd [STAT] F2 F3

{	}	NAMES	EDIT	OPS
fStat	xStat	yStat		

The LIST NAMES menu shown here has no user-created list names.

Editing an element of **xStat** or **yStat** clears any values stored to statistical result variables.

- fStat** An automatically updated list of the frequency values used in the last statistical computation requiring a frequency; default is a list where each element is 1
- xStat** An automatically updated list of the data from the x-list used in the last statistical analysis
- yStat** An automatically updated list of the data from the y-list used in the last statistical analysis

### The STAT CALC (Calculations) Menu 2nd [STAT] F1

CALC	EDIT	PLOT	DRAW	VARS	
OneVa	TwoVa	LinR	LnR	ExpR	▶ <span style="border: 1px solid black; padding: 0 2px;">PwrR</span> <span style="border: 1px solid black; padding: 0 2px;">SinR</span> <span style="border: 1px solid black; padding: 0 2px;">LgstR</span> <span style="border: 1px solid black; padding: 0 2px;">P2Reg</span> <span style="border: 1px solid black; padding: 0 2px;">P3Reg</span>
					▶ <span style="border: 1px solid black; padding: 0 2px;">P4Reg</span> <span style="border: 1px solid black; padding: 0 2px;">StReg</span> <span style="border: 1px solid black; padding: 0 2px;"></span> <span style="border: 1px solid black; padding: 0 2px;"></span> <span style="border: 1px solid black; padding: 0 2px;"></span>

The STAT CALC functions store the results to statistical result variables (page 193).

The syntax description for each STAT CALC menu item follows this section.

- OneVa** (one variable) Analyzes data with one measured variable
- TwoVa** (two variable) Analyzes paired data

For regression analysis, the statistical results are calculated using a least-squares fit.

<b>LinR</b>	(linear regression) Fits the model equation $y=a+bx$ to the data; displays values for <b>a</b> (slope) and <b>b</b> (y-intercept)
<b>LnR</b>	(logarithmic regression) Fits the model equation $y=a+b \ln x$ to the data using transformed values $\ln(x)$ and $y$ ; displays values for <b>a</b> and <b>b</b>
<b>ExpR</b>	(exponential regression) Fits the model equation $y=ab^x$ to the data using transformed values $x$ and $\ln(y)$ ; displays values for <b>a</b> and <b>b</b> ; elements in the x-list and y-list elements must be integers
<b>PwrR</b>	(power regression) Fits the model equation $y=ax^b$ to the data using transformed values $\ln(x)$ and $\ln(y)$ ; displays values for <b>a</b> and <b>b</b>
<b>SinR</b>	(sinusoidal regression) Fits the model equation $y=a*\sin(bx+c)+d$ to the data; displays values for <b>a</b> , <b>b</b> , <b>c</b> , and <b>d</b> ; <b>SinR</b> requires at least four data points; it also requires at least two data points per cycle to avoid aliased frequency estimates
<b>LgstR</b>	(logistic regression) Fits the model equation $y=a/(1+be^{cx})+d$ to the data; displays <b>a</b> , <b>b</b> , <b>c</b> , and <b>d</b>
<b>P2Reg</b>	(quadratic regression) Fits the second-degree polynomial $y=ax^2+bx+c$ to the data; displays values for <b>a</b> , <b>b</b> , and <b>c</b> ; for three data points, the equation is a polynomial fit; for four or more, it is a polynomial regression; <b>P2Reg</b> requires at least three data points
<b>P3Reg</b>	(cubic regression) Fits the third-degree polynomial $y=ax^3+bx^2+cx+d$ to the data; displays values for <b>a</b> , <b>b</b> , <b>c</b> , and <b>d</b> ; for four points, the equation is a polynomial fit; for five or more, it is a polynomial regression; <b>P3Reg</b> requires at least four data points
<b>P4Reg</b>	(quartic regression) Fits the fourth-degree polynomial $y=ax^4+bx^3+cx^2+dx+e$ to the data; displays values for <b>a</b> , <b>b</b> , <b>c</b> , <b>d</b> , and <b>e</b> ; for five points, the equation is a polynomial fit; for six or more, it is a polynomial regression; <b>P4Reg</b> requires at least five data points
<b>StReg</b>	(store regression equation) Pastes <b>StReg</b> ( to the home screen; enter a <i>variable</i> and press <u>ENTER</u> ]; the current regression equation is stored to <i>variable</i>

**SinR** and **LgstR** are calculated using an iterative least-squares fit.

When you select **OneVa** or **TwoVa**, the abbreviation **OneVar** or **TwoVar** is displayed.

For **PwrR** and **ExpR**, the elements of *xList* and *yList* must be integers  $\geq 1$ .

Default for *iterations* is 64.

For **OneVa**, the syntax is:

**OneVar** [*xList*,*frequencyList*]

For **TwoVa**, the syntax is:

**TwoVar** [*xList*,*yList*,*frequencyList* ]

For **LinR**, **LnR**, **ExpR**, **PwrR**, **P2Reg**, **P3Reg**, and **P4Reg**, the syntax is:

**TwoVar** [*xList*,*yList*,*frequencyList*]

For **SinR**, the syntax is:

**SinR** [*iterations*,*xList*,*yList*,*period*,*equationVariable*]

*iterations* is the number of iterations to go through; higher values for iterations produce a better fit, but take longer to calculate. *period* is an initial guess at which to begin calculation.

For **LgstR**, the syntax is:

**LgstR** [*iterations*,*xList*,*yList*,*frequencyList*,*equationVariable*]

To copy the contents **RegEq** to any variable after calculating the regression, the syntax is:

**StReg**(*variable*)

### Automatic Regression Equation Storage

**LinR**, **LnR**, **ExpR**, **PwrR**, **SinR**, **LgstR**, **P2Reg**, **P3Reg**, and **P4Reg** are regression models. Each regression model has an optional argument, *equationVariable*, for which you can specify an equation variable, such as **y1**. Upon execution, the regression equation is stored automatically to the specified equation variable, and the function is selected.

Regardless of whether you specify *equationVariable*, the regression equation always is stored to the result variable **RegEq**, which is an item on the STAT VARS menu. The regression equation displays the actual result values.

**PRegC** is the only statistical result variable calculated for a polynomial regression.

One- and two-variable statistical functions share the result variables.

The statistical variables are calculated and stored as shown in the table on the next page.

You can use ALPHA keys, alpha keys, and the CHAR GREEK menu to enter some result variables.

The result for a polynomial regression, sinusoidal regression, or logistic regression is stored in **PRegC** (polynomial/regression coefficients). **PRegC** is a list containing the coefficients for an equation. For example, for **P3Reg**, the result **PRegC**={3 5 -2 7} would represent  $y=3x^3+5x^2-2x+7$ .

## Results of a Statistical Analysis

When you perform a statistical analysis, the calculated results are stored in the result variables and the data from the lists used in the analysis are stored to **xStat**, **yStat**, and **fStat**. If you edit a list or change the type of analysis, all statistical variables are cleared.

### The STAT VARS (Statistical Variables) Menu

[2nd] [STAT] [F5]

CALC	EDIT	PLOT	DRAW	VARs
$\bar{x}$	$\sigma_x$	<b>Sx</b>	$\bar{y}$	$\sigma_y$

▶	<b>Sy</b>	$\Sigma x$	$\Sigma x^2$	$\Sigma y$	$\Sigma y^2$
▶	$\Sigma xy$	<b>RegEq</b>	<b>corr</b>	<b>a</b>	<b>b</b>
▶	<b>n</b>	<b>minX</b>	<b>maxX</b>	<b>minY</b>	<b>maxY</b>
▶	<b>Med</b>	<b>PRegC</b>	<b>Qrtl1</b>	<b>Qrtl3</b>	<b>tolMe</b>

To paste a result variable to the cursor location, either select the variable from the STAT VARS menu or select the variable from the VARS STAT selection screen.

- ◆ To use a result variable in an expression, paste it to the appropriate cursor location.
- ◆ To display the value of a result variable, paste it to the home screen and press [ENTER].
- ◆ To store results to another variable after a calculation, paste the result variable to the home screen, press [STO▶], enter a new variable, and then press [ENTER].

These words are abbreviated in the table:

pop = population

std dev = standard deviation

coeff = coefficient

int = intercept

reg eq = regression equation

pts = points

min = minimum

max = maximum

Result Variables	1-Var Stats	2-Var Stats	Other	Result Variables	1-Var Stats	2-Var Stats	Other
mean of x values	$\bar{x}$	$\bar{x}$		correlation coeff			<b>corr</b>
pop std dev of x	$\sigma_x$	$\sigma_x$		y-intercept of reg eq			<b>a</b>
sample std dev of x	<b>Sx</b>	<b>Sx</b>		slope of reg eq			<b>b</b>
mean of y values		$\bar{y}$		regression/fit coeff			<b>a, b</b>
pop std dev of y		$\sigma_y$		number of data pts	<b>n</b>	<b>n</b>	
sample std dev of y		<b>Sy</b>		min of x values	<b>minX</b>	<b>minX</b>	
sum of x values	$\Sigma x$	$\Sigma x$		max of x values	<b>maxX</b>	<b>maxX</b>	
sum of $x^2$ values	$\Sigma x^2$	$\Sigma x^2$		min of y values		<b>minY</b>	
sum of y values		$\Sigma y$		max of y values		<b>maxY</b>	
sum of $y^2$ values		$\Sigma y^2$		median	<b>Med</b>		
sum of $x * y$		$\Sigma xy$		1st quartile			<b>Qrt11</b>
regression equation			<b>RegEq</b>	3rd quartile			<b>Qrt13</b>
polynomial, <b>LgstR</b> , and <b>SinR</b> coeff's			<b>a</b> (y-int) <b>b</b> (slope)	polynomial <b>LgstR</b> , and <b>SinR</b> reg coeff's			<b>PRegC</b>

The first quartile (**Qrt11**) is the median of the points between **minX** and **Med** (median). The third quartile (**Qrt13**) is the median of the points between **Med** and **maxX**.

When you calculate a logistic regression, **1** is stored to **tolMet** (**tolMe**) if the TI-86 internal tolerance was met before the calculator arrived at a result; if not met, **0** is stored to **tolMet**.



When you display a stat plot editor, the STAT PLOT menu remains so that you can easily switch to another stat plot.

In this guidebook, brackets ( [ and ] ) with syntax specify arguments as optional. Do not enter brackets, except with vectors and matrices.

You need not turn on a stat plot to change the settings.

You also can use STAT PLOT menu items **PIOn** or **PIOff** to turn on or turn off stat plots.

## The STAT PLOT Menu $\boxed{2nd}$ [STAT] $\boxed{F3}$

PLOT1	PLOT2	PLOT3	PIOn	PIOff
-------	-------	-------	------	-------

**PLOT1** Displays the stat plot editor for **Plot1**

**PLOT2** Displays the stat plot editor for **Plot2**

**PLOT3** Displays the stat plot editor for **Plot3**

**PIOn [1,2,3]** Turns on all plots (if you enter no arguments) or turns on specified plots only

**PIOff [1,2,3]** Turns off all plots (if you enter no arguments) or turns off specified plots only

To turn on or turn off all three stat plots, select **PIOn** or **PIOff** from the STAT PLOT menu. **PIOn** or **PIOff** is pasted to the home screen. Press  $\boxed{ENTER}$ . All stat plots are now on or off.

## Setting Up a Stat Plot

To set up a stat plot, select **PLOT1**, **PLOT2**, or **PLOT3** from the STAT PLOT menu. The stat plot editor for the selected stat plot is displayed. Each stat plot type has a unique stat plot editor. The screen to the right shows the stat plot editor for the default  $\boxed{L1}$  (scatter plot). If you select another plot type, some prompts may change.



## Turning On and Turning Off a Stat Plot

When you display a stat plot editor, the cursor is on the **On** option.

- ◆ To turn on the stat plot, press  $\boxed{ENTER}$ .
- ◆ To turn off the stat plot, press  $\boxed{\blacktriangleright}$   $\boxed{ENTER}$ .

To display the PLOT TYPE menu, move the cursor onto the plot type icon at the Type= prompt.

When you select a plot type, the appearance of the stat plot editor may change.

### The PLOT TYPE Menu (Selecting a Plot Type)

<b>PLOT1</b>	<b>PLOT2</b>	<b>PLOT3</b>	<b>PIOn</b>	<b>PIOff</b>
<b>SCAT</b>	<b>xyLINE</b>	<b>MBOX</b>	<b>HIST</b>	<b>BOX</b>

At this prompt...	Enter this information:	Default is:	Displayed menu is:
<b>Xlist Name=</b>	independent-data list name	<b>xStat</b>	LIST NAMES menu
<b>Ylist Name=</b>	dependent-data list name	<b>yStat</b>	LIST NAMES menu
<b>Freq=</b>	frequency list name (or 1)	<b>fStat</b> (default value: 1)	LIST NAMES menu
<b>Mark=</b>	plot mark ( $\square$ or $+$ or $\bullet$ )	$\square$ (none for <b>HIST</b> or <b>BOX</b> )	PLOT MARK menu

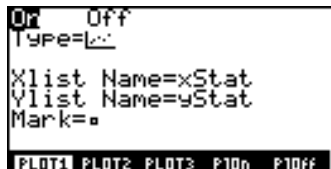
- ◆ Any list you enter at the **Xlist Name=** prompt is stored to the list name **xStat**.
- ◆ Any list you enter at the **Ylist Name=** prompt is stored to the list name **yStat**.
- ◆ Any list you enter at the **Freq=** prompt is stored to **fStat**.

### Plot Type Characteristics

Stat plots are displayed on the graph screen (**GRAPH** **F5**), as defined by the window variable values (Chapter 5). Some graph tools apply to stat plots.

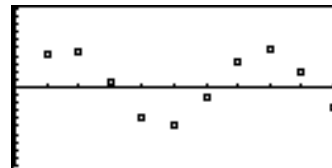
In these stat plot examples, all functions are deselected. Also, menus are cleared from the screen with **CLEAR**.

**SCAT** (scatter plot) plots the data points from **Xlist Name** and **Ylist Name** as coordinate pairs, representing each point with a box ( $\square$ ), cross ( $+$ ), or dot ( $\bullet$ ) mark type. **Xlist Name** and **Ylist Name** must be the same length. **Xlist Name** and **Ylist Name** can be the same list.





For the example:  
**xStat**={1 2 3 4 5 6 7 8 9 10}  
**yStat**=5 sin(**xStat**)

Window variable values:  
**xMin**=0      **yMin**=-10  
**xMax**=10     **yMax**=10



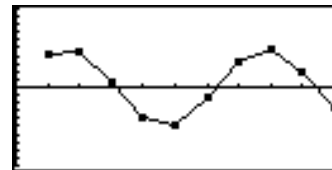



 **xyLINE** is a scatter plot in which the data points are plotted and connected in order of appearance in **Xlist Name** and **Ylist Name**. You may want to use **SortA** or **SortD** from the LIST OPS menu (Chapter 11) to sort the lists before you plot them.

```
On Off
Type=
Xlist Name=xStat
Ylist Name=yStat
Mark=•
PLOT1 PLOT2 PLOT3 P1On P1Off
```

For the example:  
**xStat**={1 2 3 4 5 6 7 8 9 10}  
**yStat**=5 sin(xStat)

Window variable values:  
**xMin**=0      **yMin**=-10  
**xMax**=10     **yMax**=10

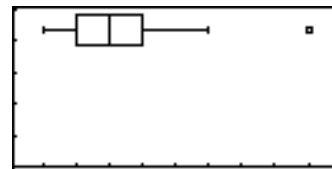


 **MBOX** (modified box plot) plots one-variable data, like the regular box plot, except that the points are  $1.5 \times$  Interquartile Range beyond the quartiles. (The Interquartile Range is defined as the difference between the third quartile **Q<sub>3</sub>** and the first quartile **Q<sub>1</sub>**.) These points are plotted individually beyond the whisker, using the **Mark** ( $\square$  or  $+$  or  $\bullet$ ) you select.

```
On Off
Type=MBOX
Xlist Name=xStat
Freq=1
Mark=•
PLOT1 PLOT2 PLOT3 P1On P1Off
```

For the example:  
**xStat**={1 2 2 2.5 3 3.3 4 4 2 6 9}

Window variable values are set by selecting **ZDATA** from the GRAPH ZOOM menu.




*Whiskers are the lines protruding from the sides of the box.*

You can trace these points, which are called outliers. When outliers exist, the end of each whisker will display an **x=** prompt. When no outliers exist, **xMin** and **xMax** are the prompts for the end of each whisker. **Q<sub>1</sub>**, **Med** (median), and **Q<sub>3</sub>** define the box.

Modified box plots are plotted with respect to **xMin** and **xMax**, but ignore **yMin** and **yMax**. When two modified box plots are plotted, the first one plots at the top of the screen and the

second plots in the middle. When three are plotted, the first one plots at the top, the second in the middle, and the third at the bottom.

 **HIST** (histogram) plots one-variable data. The **xScl** window variable value determines the width of each bar, beginning at **xMin**. **ZDATA** (GRAPH ZOOM menu) adjusts **xMin**, **xMax**, **yMin**, and **yMax** to include all values, and also adjusts **xScl**.  $(xMax - xMin) / xScl \leq 47$  must be true. A value that occurs on the edge of a bar is counted in the bar to the right.

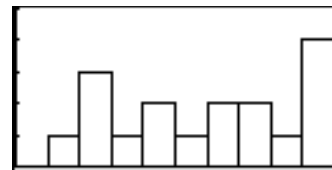
```

Off Off
Type=Hn
Xlist Name=xStat
Freq=1
PLOT1 PLOT2 PLOT3 P10n P10ff


```

For the example:  
**xStat**={1 2 2 2 3 8 9 5 6 6 7 7  
 4 4 9 9 9}

Window variable values:  
**xMin**=0      **yMin**=0  
**xMax**=10     **yMax**=5



*Whiskers are the lines protruding from the sides of the box.*

 **BOX** (regular box plot) plots one-variable data. The whiskers on the plot extend from the minimum data point in the set (**xMin**) to the first quartile (**Q<sub>1</sub>**) and from the third quartile (**Q<sub>3</sub>**) to the maximum point (**xMax**). The box is defined by **Q<sub>1</sub>**, **Med** (median), and **Q<sub>3</sub>**.

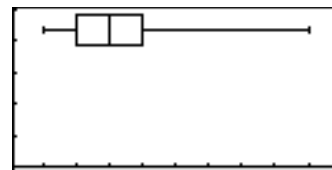
```

Off Off
Type=On
Xlist Name=xStat
Freq=1
PLOT1 PLOT2 PLOT3 P10n P10ff

```

For the example:  
**xStat**={1 2 2 2.5 3 3.3 4 4 2 6 9}

Window variable values are set by selecting **ZDATA** from the GRAPH ZOOM menu.



Box plots are plotted with respect to **xMin** and **xMax**, but ignore **yMin** and **yMax**. When two box plots are plotted, the first one plots at the top of the screen and the second plots in the

middle. When three are plotted, the first one plots at the top, the second in the middle, and the third at the bottom.

## The STAT DRAW Menu 2nd [STAT] F4

CALC	EDIT	PLOT	DRAW	VAR		DRREG	CLDRW	DrawF	STPIC	RPCIC
HIST	SCAT	xyLINE	BOX	MBOX	▶					

When you select any of the first five STAT DRAW menu items, the TI-86 plots the data stored in the lists **xStat** and **yStat**.

<b>HIST</b>	Draws a histogram of one-variable data
<b>SCAT</b>	Draws a scatter plot of the data points
<b>xyLINE</b>	Draws the data points and a line connecting each point to the next point
<b>BOX</b>	Draws a box plot of the data points
<b>MBOX</b>	Draws a modified box plot of the data points
<b>DRREG</b>	(draw regression equation) Draws the current regression equation
<b>CLDRW</b>	(clear drawings) Displays the current graph with no drawings
<b>DrawF</b>	<i>expression</i> (draw function) Plots <i>expression</i> as a drawing
<b>STPIC</b>	(store picture) Displays the picture variable <b>Name=</b> prompt; enter a valid variable name, starting with a letter, and then press <span style="border: 1px solid black; padding: 1px;">ENTER</span> to store the current picture
<b>RPCIC</b>	(recall picture) Displays the picture variable <b>Name=</b> prompt and menu; select or enter a valid variable name, and then press <span style="border: 1px solid black; padding: 1px;">ENTER</span> ; the stored picture is redrawn

## Forecasting a Statistical Data Value

Using the forecast editor, you can forecast an x-value or y-value based on the current regression equation. To use the forecast editor, a regression equation must be stored to **RegEq**.

- ❶ Enter stat data in the list editor. The screen to the right shows all **fStat** elements as **1**, but you need not enter them. **1** is the default for all **fStat** elements. However, if other elements are stored to **fStat**, you must clear them.
- ❷ Display the home screen.
- ❸ Execute a linear regression for **xStat** and **yStat**. The statistical results are displayed.
- ❹ Remove the STAT CALC menu to display all results, including **n**.
- ❺ Display the forecast editor. The current regression model is displayed on the top line.
- ❻ Enter **x=3**, and then move the cursor to the **y=** prompt.
- ❼ Select **SOLVE** from the forecast editor menu to solve for **y** at **x=3**. A small square indicates the solution. You can continue to use the forecast editor with other values for **x** or **y**.

*Values entered at forecast editor prompts must be real numbers or expressions that evaluate to real numbers.*

*If the most recent calculation was a polynomial regression, you can only forecast the y value.*

2nd [STAT] F2  
 □ 1 ▾ □ 1  
 ▾ 2 ▾ 4 ▾ 5  
 ▾ ▸ 1 ▾ 2  
 ▾ 3 ▾ 4 ▾ 2

[EXIT]

2nd [STAT] F1  
 F3 [ENTER]

[EXIT]

[MORE] F1

3 ▾

[F5]

xStat	yStat	fStat
1	1	1
1	1	1
1	1	1
1	1	1
1	1	1

fStat(6) =

< > NAMES " OPS

LinReg
y=a+bx
a=1.65548262
b=.305130075
corr=.54274108
n=5

CALC EDIT PLOT DRAW VARS

FORECAST:LinReg
x=3
y=

SOLVE

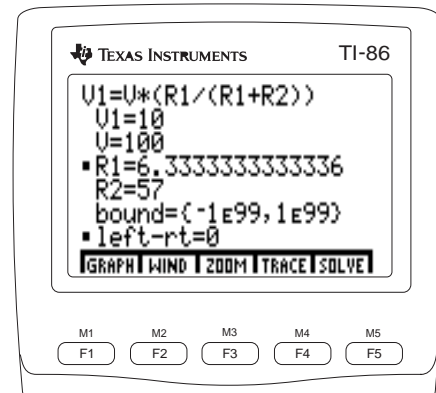
FORECAST:LinReg
x=3
y=2.5708728422076

SOLVE

When you use **FCST**, the values of **x**, **y**, and **Ans** are not updated. To store the **x** value or **y** value, move the cursor onto the variable to be stored, press **[STO▶]**, enter a valid variable name at the **Sto** prompt, and then press **[ENTER]**.

# 15 Equation Solving

Preview: The Equation Solver .....	202
Entering an Equation in the Equation-Entry Editor .....	203
Setting Up the Interactive-Solver Editor .....	204
Solving for the Unknown Variable .....	206
Graphing the Solution.....	207
Solver Graph Tools.....	207
The Simultaneous Equation Solver .....	208
The Polynomial Root-Finder .....	211



## Preview: The Equation Solver 2nd SOLVER

With the equation solver, you can enter an expression or equation, store values to all but one variable in the expression or equation, and then solve for the unknown variable. These steps introduce the solver. For details, read this chapter.

The VARS EQU menu is a menu version of the VARS EQU screen (Chapter 2).

The example uses a formula for a voltage divider.

R1 and R2 represent resistors.

V and V1 represent voltage.

To solve for the unknown variable in an equation on the home screen or in the program editor, select **Solver** from the CATALOG (A to Z Reference).

- 1 Display the equation-entry editor. The VARS EQU menu is displayed on the bottom of the screen.
- 2 Enter an equation. When you press ENTER, the interactive-solver editor and solver menu are displayed.
- 3 Enter values for each variable, except the unknown variable **R1**. Some variables may have values stored to them already.
- 4 Move the cursor to the variable for which you want to solve. You may enter a guess.
- 5 Solve the equation for the variable. Small squares mark both the solution variable and the equation **left-rt=0** (the left side of the equation minus the right side of the equation). If you edit a value or leave the screen, the squares disappear.

2nd SOLVER  
ALPHA [V] 1 ALPHA  
[=] ALPHA [V] [ ]  
ALPHA [R] 1 [ ] [ ]  
ALPHA [R] 1 [+]  
ALPHA [R] 2 [ ] [ ]  
ENTER

10 ▾ 100 ▾ ▾ 57

▲

F5

```
eqn:U1=U(R1/(R1+R2))
```

```
U1=U(R1/(R1+R2))
U1=
U=
R1=
R2=
bound=C-1E99,1E99
```

GRAPH WIND ZOOM TRACE SOLVE

```
U1=U(R1/(R1+R2))
U1=10
U=100
R1=
R2=57
bound=C-1E99,1E99
```

GRAPH WIND ZOOM TRACE SOLVE

```
U1=U(R1/(R1+R2))
U1=10
U=100
R1=6.33333333333336
R2=57
bound=C-1E99,1E99
left-rt=0
```

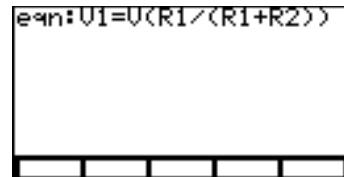
GRAPH WIND ZOOM TRACE SOLVE

## Entering an Equation in the Equation-Entry Editor

The equation solver uses two editors: the equation-entry editor, where you enter and edit the equation you want to solve, and the interactive-solver editor, where you enter known variable values, select the variable for which you want to solve, and display the solution.

To display the equation-entry editor, press  $\boxed{2\text{nd}}$  [SOLVER]. In this editor, you can:

- ◆ Enter an equation directly.
- ◆ Enter a defined equation variable's individual characters or select it from the VARS EQU menu.
- ◆ Recall the contents of a defined equation variable.



The equation can have more than one variable to the left of the equal sign, as in

$$A+B=C+\sin D.$$

As you enter or edit the equation, the TI-86 automatically stores it to the variable **eqn**.

You can display other menus in the equation-entry editor.

The VARS EQU menu is a menu version of the VARS EQU screen (Chapter 2). The items are all variables to which an equation is stored. This includes all selected and deselected equation variables defined in the equation editors of all four graphing modes (Chapters 5, 8, 9, and 10). The menu items are in alphanumeric order.

An ellipsis (...) indicates that an entered equation continues beyond the screen. To move directly to the start of the equation, press  $\boxed{2\text{nd}}$   $\boxed{\leftarrow}$ ; to move directly to the end, press  $\boxed{2\text{nd}}$   $\boxed{\rightarrow}$ .

- ◆ If you select an equation variable from the menu, the variable is pasted to the cursor location, overwriting characters for the length of the variable name.
- ◆ If you press  $\boxed{2\text{nd}}$  [RCL], select an equation variable from the menu, and then press  $\boxed{\text{ENTER}}$ , the variable contents are inserted at the cursor location.

If you enter an equation variable, the TI-86 automatically converts it to the equation **exp=equationVariable**. If you enter an expression directly, the TI-86 automatically converts the expression to the equation **exp=expression**.

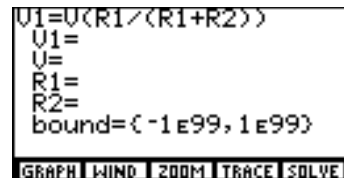
In the example, the equation  $V1=V(R1/(R1+R2))$  was entered in the equation-entry editor.

If you entered an expression for **eqn**, then **exp=** is the first variable prompt on the interactive-solver editor.

## Setting Up the Interactive-Solver Editor

After you have stored an equation to **eqn** in the equation-entry editor, press **ENTER** to display the interactive-solver editor.

The equation is displayed across the top of the editor. Each variable in the equation is displayed as a prompt. Values already stored to variables are displayed; undefined variables are blank. The solver menu is displayed on the bottom of the editor (page 206).



**bound={-1E99,1E99}** is a list containing the default lower bound (**-1E99**) and the default upper bound (**1E99**). You can edit the bounds (below).

### Entering Variable Values

To solve for an unknown variable, you must define every other variable in the equation. When you enter or edit a variable value in the interactive-solver editor, the new value is stored to the variable in memory. For any variable, you may enter an expression, which is evaluated when you press **ENTER**, **↓**, **↑**, or **EXIT**. Expressions must resolve to real numbers at each step of the calculation.

### Controlling the Solution with Bounds and a Guess

The solver seeks a solution only within the specified bounds. Whenever you display the interactive-solver editor, the default **bound={-1E99,1E99}** is displayed. These are the maximum bounds for the TI-86.



The TI-86 solves equations through an iterative process. To control that process, you can enter lower bounds and upper bounds that are close to the solution, and enter a guess within those bounds in the prompt for the unknown variable.

Controlling the process with specific bounds and a guess helps the TI-86 in two ways.

- ◆ It finds a solution more quickly.
- ◆ It is more likely to find the solution you want when an equation has multiple solutions.

To set more precise bounds at the **bound=** prompt, the syntax is:

**bound={lowerBound,upperBound}**

At the prompt for the unknown variable, you may enter a guess or a list of two guesses. If you do not enter a guess, the TI-86 uses  $(lowerBound+upperBound)/2$  as a guess.

On the solver graph (page 207), you can guess a solution by moving the free-moving cursor or trace cursor to a point on the graph between *lowerBound* and *upperBound*. To solve for the unknown variable using the new guess, select **SOLVE** from the solver graph menu. The solution is displayed on the interactive-solver editor.

### Editing the Equation

To edit the equation stored to **eqn** when the interactive-solver editor is displayed, press  $\square$  until the cursor is on the equation. The equation-entry editor is displayed. The TI-86 automatically stores the edited equation to **eqn** as you edit.

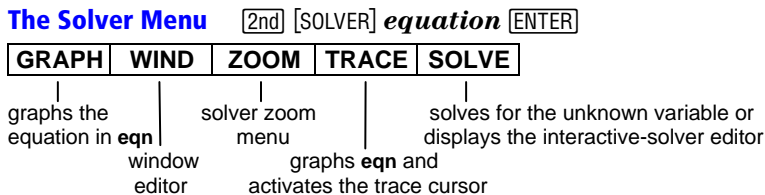
If you store an equation to **eqn** by recalling the contents of an equation variable, such as **y1**, and then edit the equation stored to **eqn**, the original equation (in **y1**, for example) is not changed. Likewise, subsequently editing the contents of the equation variable (**y1**, for example) does not change **eqn**.

*lowerBound < upperBound  
must be true.*

*You can enter a list variable  
at the **bound=** prompt if a  
valid two-element list is  
stored to it.*

*If you exit the equation  
solver, any equation stored to  
eqn is displayed when you  
return to the equation solver.*

You can display other menus in the interactive-solver editor



To display the window editor, select **WIND** from the solver menu.

When you select **GRAPH** or **WIND** from the solver menu, **EDIT** replaces the item you selected on the menu. To return to the interactive-solver editor from the graph or window editor, select **EDIT**.

## Solving for the Unknown Variable

After you have stored all known variable values, set the bounds, and entered a guess (optional), move the cursor to the prompt for the unknown variable.

An ellipsis (...) indicates that the variable value continues beyond the screen. To scroll the value, press  $\boxed{\rightarrow}$  and  $\boxed{\downarrow}$ .

The squares disappear when you edit any value.

After solving, you can edit a variable value or edit the equation, and then solve for the same variable or another variable in the equation.

To solve, select **SOLVE** from the solver menu ( $\boxed{[F5]}$ ).

- ◆ A small square marks the variable for which you solved. The solution value is displayed.
- ◆ A small square also marks the **left-rt=** prompt. The value at this prompt is the value of the left side of the equation minus the value of the right side of the equation, evaluated at the new value of the variable for which you solved. If the solution is precise, **left-rt=0** is displayed.

```

U1=U(R1/(R1+R2))
U1=10
U=100
▪ R1=6.33333333333336
R2=57
bound=(-1e99,1e99)
▪ left-rt=0
GRAPH WIND ZOOM TRACE SOLVE
  
```

Some equations have more than one solution. To look for additional solutions, you can enter a new guess or set new bounds, and then solve for the same variable.

The graph to the right plots the solution from the example on page 202. The window variable values are:  $xMin=-10$   
 $yMin=-50$   
 $xMax=50$   $yMax=50$

## Graphing the Solution

When you select **GRAPH** from the solver menu ( $\boxed{F1}$ ), the solver graph is displayed with the free-moving cursor.

- ◆ The vertical axis represents the result of the left side of the equation minus the right side of the equation (left-right) at each independent variable value.
- ◆ The horizontal axis represents the independent variable for which you solved the equation.



On the graph, solutions exist for the equation where  $left-rt=0$ , which is where the graph intersects the x-axis. The solver graph:

- ◆ Uses the current window and format settings (Chapter 5).
- ◆ Does not graph the solution according to the current graphing mode.
- ◆ Always graphs a solution as a function graph.
- ◆ Does not graph selected functions or turned on stat plots along with the solution.

## Solver Graph Tools

You can explore the graph of a solution with the free-moving cursor, as you would on any other graph. When you do, the coordinate values for the variable (the x-axis) and the value  $left-rt$  (the y-axis) are updated.

To activate the trace cursor, select **TRACE** from the solver menu. Panning, QuickZoom, and entering a specific value (Chapter 6) are available with the trace cursor on the solver graph.

To return to the solver menu from a trace, press  $\boxed{EXIT}$ .

*You can use the free-moving cursor or trace cursor to select a guess on the graph.*

### The Solver ZOOM Menu $\boxed{2\text{nd}} \text{ [SOLVER]} \textit{ equation} \text{ [ENTER]} \text{ [F3]}$

GRAPH	WIND	ZOOM	TRACE	SOLVE
BOX	ZIN	ZOUT	ZFACT	ZSTD

Chapter 6 and the A to Z Reference describe these features in detail.

- BOX** Draws a box to redefine the viewing window (Chapter 6)
- ZIN** Magnifies the graph around the cursor by factors of **xFact** and **yFact** (Chapter 6)
- ZOUT** Displays more of the graph around the cursor by factors of **xFact** and **yFact** (Chapter 6)
- ZFACT** Displays the ZOOM FACTORS screen (Chapter 6)
- ZSTD** Displays the graph in standard dimensions; resets the default window variable values for **Func** graphing mode

### The Simultaneous Equation Solver $\boxed{2\text{nd}} \text{ [SIMULT]}$

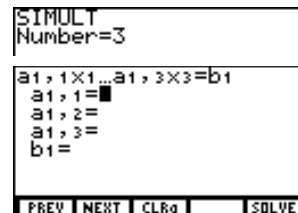
The simultaneous equation solver solves systems of up to 30 linear equations with 30 unknowns.

#### Entering Equations to Solve Simultaneously

- 1 Display the SIMULT number screen.  $\boxed{2\text{nd}} \text{ [SIMULT]}$
- 2 Enter an integer  $\geq 2$  and  $\leq 30$  for the number of equations. The coefficients-entry editor for the first equation (for a system of  $n$  equations and  $n$  unknowns) is displayed. The SIMULT ENTRY menu also is displayed.  $\mathbf{3} \text{ [ENTER]}$

The SIMULT coefficients are not variables.

You can display other menus in the coefficients-entry screen.



To move from the coefficients-entry editor for one equation to the editor for another equation, select **PREV** or **NEXT**.

To move among coefficients, press  $\leftarrow$ ,  $\rightarrow$ , or **ENTER**. From the last or first coefficient, these keys move to the next or previous coefficients-entry screen, if possible.

Ellipses indicate that a value continues beyond the screen. Press  $\uparrow$  and  $\downarrow$  to scroll the value.

- Enter a real or complex value (or an expression that resolves to one) for each coefficient in the equation and for  $b_1$ , which is the solution to that equation.

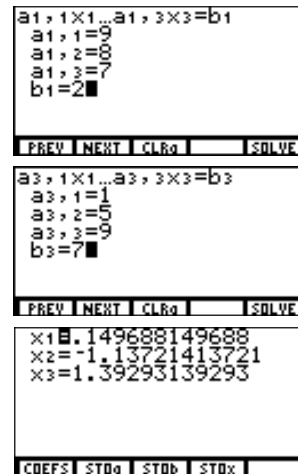
9  $\downarrow$  8  $\downarrow$  7  $\downarrow$  2

- Display the coefficients-entry screen for the second and third equation, and enter values for them.

$\downarrow$  (or **ENTER** or **F2**) 5  $\downarrow$  (-) 6  $\downarrow$  (-)  
4  $\downarrow$  2  
 $\downarrow$  1  $\downarrow$  5  $\downarrow$  9  $\downarrow$  7

- Solve the equations. The results of the polynomial are calculated and displayed on the result screen. Results are not stored to variables and cannot be edited. The **SIMULT RESULT** menu is displayed.

**F5**



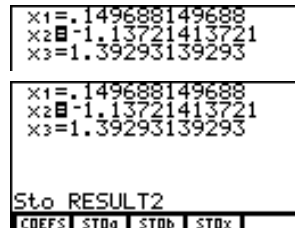
### Storing Equation Coefficients and Results to Variables

- ◆ To store coefficients  $a_{1,1}$ ;  $a_{1,2}$ ;...; $a_{n,n}$  to an  $n \times n$  matrix, select **STOa**.
- ◆ To store solutions  $b_1, b_2, \dots, b_n$  to a vector of dimension  $n$ , select **STOb**.
- ◆ To store the results  $x_1, x_2, \dots, x_n$  to a vector of dimension  $n$ , select **STOx**.

To store a single value on the coefficients-entry screen or result screen, follow these steps.

To switch to the coefficients-entry screen, select **COEFS** from the SIMULT RESULT menu.

- 1 Move the cursor to the = sign next to the coefficient or result you want to store. [ ]
- 2 Display the variable **Name=** prompt. ALPHA-lock is on. [STO▶]
- 3 Enter the variable to which you want to store the value. [R][E][S][U][L]  
[T] [ALPHA] 2
- 4 Store the value. The variable name becomes an item on the VARS REAL screen or VARS CPLX screen. [ENTER]



To solve equations simultaneously on the home screen or in a program, select **simult** from the CATALOG.

To return to the coefficients-entry screen, where you can edit coefficients and calculate new solutions, select **COEFS** from the SIMULT RESULT menu.

## The Polynomial Root-Finder 2nd POLY

The root finder solves up to 30th-order real or complex polynomials.

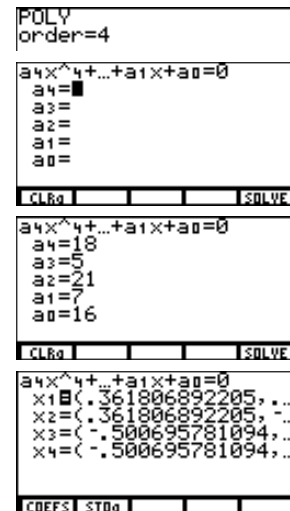
### Entering and Solving a Polynomial

The POLY coefficients are not variables.

You can display other menus in the coefficients-entry editor.

Ellipses indicate that a value continues beyond the screen. Press ▶ and ◀ to scroll the value.

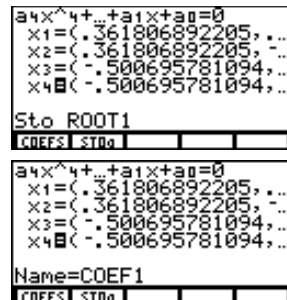
- 1 Display the POLY order screen. 2nd POLY
- 2 Enter an integer between 2 and 30. The coefficients-entry editor is displayed with the equation across the top, the coefficient prompts along the left side, and the POLY ENTRY menu on the bottom. 4 ENTER
- 3 Enter a real or complex value (or an expression that resolves to one) for each coefficient. 18 ▼ 5 ▼ 21  
7 ▼ 16  
To clear all coefficients, select **CLRa** from the POLY ENTRY menu.
- 4 Solve the equation. The roots of the polynomial are calculated and displayed. Results are not stored to variables and you cannot edit them. Also, the POLY RESULT menu is displayed. Results can be complex numbers. F5



To switch to the coefficients-entry screen, select **COEFS** from the POLY RESULT menu.

### Storing a Polynomial Coefficient or Root to a Variable

- 1 Move the cursor to the = sign next to the coefficient or root value you want to store. [▼] [▼] [▼]
- 2 Display the **Sto** prompt. ALPHA-lock is on. [STO▶]
- 3 Enter the variable to which you want to store the value. [R] [O] [O] [T]  
[ALPHA] 1
- 4 Store the value. [ENTER]
- 5 Display the **Name=** prompt for the coefficients list name. ALPHA-lock is on. [F2]
- 6 Enter the list variable name to which you want to store the coefficients. [C] [O] [E] [F]  
[ALPHA] 1
- 7 Store the polynomial coefficient values. [ENTER]



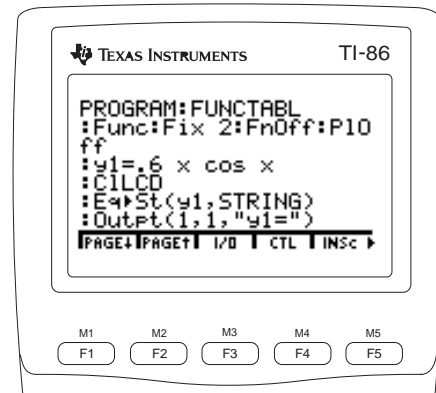
To find roots on the home screen or in a program, select **poly** from the CATALOG.

To return to the coefficients-entry screen, where you can edit coefficients and calculate new solutions, select **COEFS** from the POLY RESULT menu.



# 16 Programming

Writing a Program on the TI-86 .....	214
Running a Program .....	221
Working with Programs .....	223
Running an Assembly Language Program .....	225
Entering and Storing a String.....	226

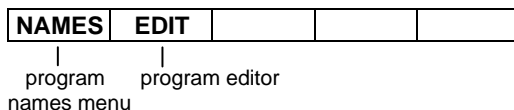


## Writing a Program on the TI-86

A program is a set of expressions, instructions, or both, which you enter or download. Expressions and instructions in the program are executed when you run the program.

You can use most TI-86 features in a program. Programs can retrieve and update all variables stored to memory. Also, the program editor menu has input/output commands, such as **Input** and **Disp**, and program control commands, such as **If**, **Then**, **For**, and **While**.

### The PRGM Menu



### Creating a Program in the Program Editor

To begin writing a program, select **EDIT** from the PRGM menu ([PRGM] [F2]). The program **Name=** prompt and PRGM NAMES menu are displayed. ALPHA-lock is on. Enter a program name from one to eight characters long, beginning with a letter. To edit an existing program, you can select the name from the PRGM NAMES menu.



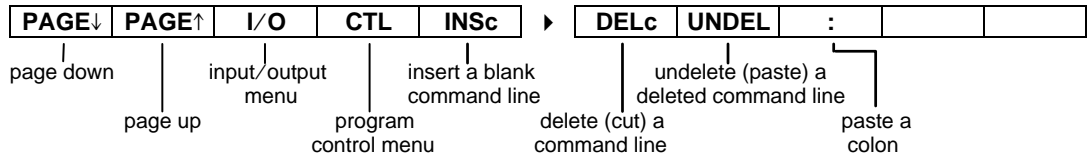
*The TI-86 distinguishes between uppercase and lowercase letters in program names. For example, **ABC**, **Abc**, and **abc** would be three different program names.*

After you enter a program name, press **ENTER**. The program editor and program editor menu are displayed. The program name is displayed at the top of the screen. The cursor is on the first command line, which begins with a colon. The TI-86 automatically places a colon at the beginning of each command line.

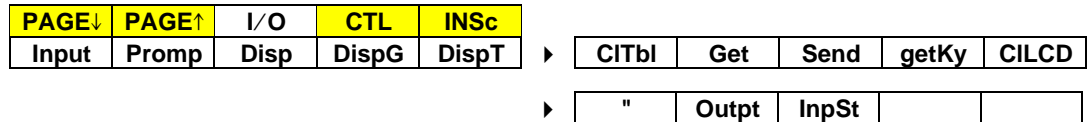


As you write the program, the commands are stored to the program name.

**The Program Editor Menu** **PRGM** **F2** *programName* **ENTER**



**The PRGM I/O (Input/Output) Menu** **PRGM** **F2** *programName* **ENTER** **F3**



*The PRGM I/O menu items are instructions. The actions they perform occur as the program runs.*

To see examples that show how to use PRGM I/O menu items in programs, refer to the A to Z Reference.

If you enter an expression for *variable* at an **Input** or **Prompt** prompt, it is evaluated and stored.

For **Input** and **Prompt**, built-in variables such as **y1** and **r1** are not valid as *variable*.

To halt the program temporarily after **Disp** or **DispG** and examine what the program is displaying, enter **Pause** on the next command line (page 219).

<b>Input</b>	Displays the current graph and lets you use the free-moving cursor
<b>Input</b> <i>variable</i>	Pauses a program, displays ? as a prompt, and then stores your response to <i>variable</i>
<b>Input</b> <i>promptString,variable</i>	Pauses a program, displays <i>promptString</i> or <i>string</i> (up to 21 characters) as a prompt, and then stores your response to <i>variable</i>
<b>Input</b> "string", <i>variable</i>	Although using <b>Get</b> ( is preferred on the TI-86, you can use <b>Input</b> to receive <i>variable</i> from a CBL 2/CBL, CBR, or TI-86 (TI-85 compatible)
<b>Input</b> "CBLGET", <i>variable</i>	Displays each <i>variable</i> with ? to prompt you to enter a value for that <i>variable</i>
<b>Prompt</b> <i>variableA</i> [, <i>variableB,variableC,...</i> ]	Displays the home screen
<b>Disp</b>	Displays each <i>value</i>
<b>Disp</b> <i>valueA,valueB,...</i>	Displays the value stored to each <i>variable</i>
<b>Disp</b> <i>variableA,variableB,...</i>	Displays each <i>text</i> string on the left side of the current display line
<b>Disp</b> "textA","textB",...	Displays the current graph
<b>DispG</b>	Displays the current table and temporarily halts the program
<b>DispT</b>	Clears the current table if <b>Indpnt: Ask</b> is set (Chapter 7)
<b>CITbl</b>	Gets data from a CBL 2/CBL, CBR, or another TI-86 and stores it to <i>variable</i>
<b>Get</b> ( <i>variable</i> )	Sends the contents of <i>listName</i> to a CBL 2/CBL or CBR
<b>Send</b> ( <i>listName</i> )	Returns a number corresponding to the last key pressed, according to the key code diagram (page 217); if no key was pressed, returns <b>0</b>
<b>getKy</b>	Clears the home screen (LCD stands for liquid crystal display)
<b>CILCD</b>	

"string"

Specifies the beginning and end of a *string*

**Output**(row,column,"string")

Displays *string*, *stringName*, *value*, or a value stored to *variable* beginning at the specified *row* and *column* on the display

**Output**(row,column,stringName)

**Output**(row,column,value)

**Output**(row,column,variable)

**Output**("CBLSEND",listName)

Although using **Send**( is preferred on the TI-86, you can use **Output**( to send *listName* to a CBL 2/CBL or CBR (for TI-85 compatibility)

**InpSt** *promptString,variable*

Pauses a program, displays *promptString* or **?**, and waits for a response; stores the response to *variable* always as a string; omit quotation marks from your response

**InpSt** *variable*

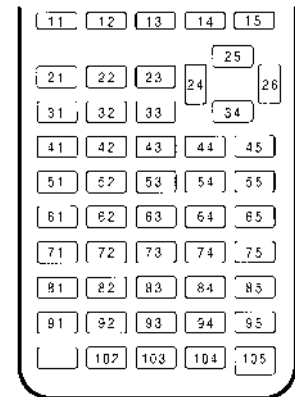
## The TI-86 Key Code Diagram

When **getKey** is encountered in a program, it returns a number corresponding to the last key pressed, according to the key code diagram to the right. If no key has been pressed, **getKey** returns **0**. Use **getKey** inside loops to transfer control, such as when you create a video game.

This program returns the key code of each key you press.

```
:Float
:0→A
:Lbl TOP
:getKey→A
:If A>0
:Disp A
:Goto TOP
```

To break (interrupt) the program, press **ON** and then press **F5**.



The PRGM CTL Menu [PRGM] [F2] *programName* [ENTER] [F4]

PAGE↓	PAGE↑	I/O	CTL	INSc	
If	Then	Else	For	End	▶ While Repea Menu Lbl Goto
					▶ IS> DS< Pause Retur Stop
					▶ DelVa GrStl LCust

To see examples that show how to use PRGM CTL menu items in programs, refer to the A to Z Reference.

**If, While, and Repeat**  
instructions can be nested.

**If** *condition*

If *condition* is false (evaluates to 0), the next program command is skipped; if *condition* is true (evaluates to a nonzero value), the program continues on to the next command

**Then**

Following **If**, executes a group of commands if *condition* is true

**Else**

Following **If** and **Then**, executes a group of commands if *condition* is false

**For** (*loops can be nested*).

**For**(*variable,begin,end*  
[,*step*])

Starting at *begin*, repeats a group of commands by an optional real *step* until *variable* > *end*; default *step* is 1

**End**

Identifies the end of a group of program commands; **For**(, **While**, **Repeat**, and **Else** groups must end with **End**; **Then** groups without an associated **Else** instruction also must end with **End**








**While** *condition*

Repeats a group of commands while *condition* is true; *condition* is tested when the **While** instruction is encountered; typically, the expression that defines *condition* is a relational test (Chapter 3)

**Repeat** *condition*

Repeats a group of commands until *condition* is true; *condition* is



	tested when the <b>End</b> instruction is encountered
<b>Menu</b> ( <i>item#</i> ," <i>title1</i> ", <i>label1</i> [, <i>item#</i> , " <i>title2</i> ", <i>label2</i> ,...])	Sets up branching within a program as selected from menu keys <b>[F1]</b> through <b>[F5]</b> ; when encountered, displays the first of up to 3 menu groups (up to 15 <i>titles</i> ); when you select a <i>title</i> , the program branches to the <i>label</i> that the <i>title</i> represents; <i>item#</i> is an integer $\geq 1$ and $\leq 15$ that specifies <i>title</i> 's menu placement; <i>title</i> is a text string from one to eight characters long (may be abbreviated in the menu)
<b>Lbl</b> <i>label</i>	Assigns a <i>label</i> to a program command; label can be one to eight characters long, starting with a letter
<b>Goto</b> <i>label</i>	Transfers control to the program branch labeled with <i>label</i>
<b>IS&gt;</b> ( <i>variable</i> , <i>value</i> )	Adds 1 to <i>variable</i> ; if the answer is $> value$ , the next command is skipped; if the answer is $\leq value$ , the next command is executed; <i>variable</i> cannot be a built-in variable
<b>DS&lt;</b> ( <i>variable</i> , <i>value</i> )	Subtracts 1 from <i>variable</i> ; if the answer is $< value$ , the next command is skipped; if the answer is $\geq value$ , the next command is executed; <i>variable</i> cannot be a built-in variable
<b>Pause</b>	Halts the program so that you can examine results, including displayed graphs and tables; to resume the program, press <b>[ENTER]</b>
<b>Pause</b> <i>value</i>	Displays <i>value</i> on the home screen so that you can scroll large values, such as lists, vectors, or matrices; to resume, press <b>[ENTER]</b>
<b>Return</b>	Exits a subroutine (page 224) and returns to the calling program, even if encountered within nested loops; within the main program, stops the program and returns to the home screen (an implied <b>Return</b> exits each subroutine upon completion and returns to the calling program)
<b>Stop</b>	Stops a program and returns to the home screen

<b>DelVar</b> ( <i>variable</i> )	Deletes from memory <i>variable</i> (except program names) and its contents
<b>GrStl</b> ( <i>function#</i> , <i>graphStyle#</i> )	Specifies the graph style represented by <i>graphStyle#</i> for the function represented by <i>function#</i> ; <i>function#</i> is the number part of an equation variable, such as the <b>5</b> in <b>y5</b> ; <i>graphStyle#</i> is an integer $\geq 1$ and $\leq 7$ , where <b>1</b> =  (line), <b>2</b> =  (thick), <b>3</b> =  (shade above), <b>4</b> =  (shade below), <b>5</b> =  (path), <b>6</b> =  (animate), and <b>7</b> =  (dotted)
<b>*LCust</b> ( <i>item#</i> ," <i>title</i> " [, <i>item#</i> ," <i>title</i> ",...])	Loads (defines) the TI-86 custom menu, which is displayed when you press <b>CUSTOM</b> ; <i>item#</i> is an integer $\geq 1$ and $\leq 15$ ; <i>title</i> is a string with one to eight characters (may be abbreviated in the menu)

*A command line that is longer than the screen is wide automatically continues at the beginning of the next line.*

### Entering a Command Line

You can enter on a command line any instruction or expression that you could execute on the home screen. In the program editor, each new command line begins with a colon. To enter more than one instruction or expression on a single command line, separate each with a colon.

To move the cursor down to the next new command line, press **ENTER**. You cannot move to the next new command line by pressing . However, you can return to existing command lines to edit them by pressing .

### Menus and Screens in the Program Editor

*All CATALOG items are valid in the program editor.*

TI-86 menus and screens may be altered when displayed in the program editor. Menu items that are invalid for a program are omitted from menus. Menus that are not valid in a program, such as the LINK menu or MEM menu, are not displayed at all.

When you select a setting from a screen such as the mode screen or graph format screen, the setting you select is pasted to the cursor location on the command line.



Variables to which you typically store values from an editor, such as the window variables, become items on program-only menus, such as the GRAPH WIND menu. When you select them, they are pasted to the cursor location on the command line.

## Running a Program

- 1 Paste the program name to the home screen. Either select it from the PRGM NAMES menu ( $\text{PRGM}$   $\text{F1}$ ) or enter individual characters.
- 2 Press  $\text{ENTER}$ . The program begins to run.

Each result updates the last-answer variable **Ans** (Chapter 1). The TI-86 reports errors as the program runs. Commands executed during a program do not update the previous-entry storage area ENTRY (Chapter 1).

The example program below is shown as it would appear on a TI-86 screen. The program:

- ◆ Creates a table by evaluating a function, its first derivative, and its second derivative at intervals in the graphing window
- ◆ Displays the graph of the function and its derivatives in three different graph styles, activates the trace cursor, and pauses to allow you to trace the function

*To resume the program after a pause, press  $\text{ENTER}$ .*

PROGRAM:FUNCTABL	The name of the program
:Func:Fix 2:FnoFF:P10ff	Set graphing and decimal modes (mode screen); turn off functions (GRAPH VARS menu) and plots (STAT PLOT menu)
:y1=.6 x cos x	Define the function (assignment statement)
:C1LCD	Clear the home screen (PRGM I/O menu)
:EqSt(y1,STRING)	Convert <b>y1</b> into the string variable <b>STRING</b> (STRNG menu)
:Outpt(1,1,"y1=")	Display <b>y1=</b> at row 1, column 1 (PRGM I/O menu)
:Outpt(1,4,STRING)	Display value stored to <b>STRING</b> at row 1, col. 4 (PRGM I/O menu)
:Outpt(8,1,"PRESS ENTER")	Display <b>PRESS ENTER</b> at line 8, column 1 (PRGM I/O menu)
:Pause	Pause the program (PRGM CTL menu)
:C1LCD	Clear the home screen (PRGM I/O menu)
:y2=der1(y1,x,x)	Define <b>y2</b> as the first derivative of <b>y1</b> (CALC menu)
:y3=der2(y1,x,x)	Define <b>y3</b> as the second derivative of <b>y1</b> (CALC menu)
:DispT	Display the table (PRGM I/O menu)
:GrSt1(1,1):GrSt1(2,2)	Set graph styles for <b>y1</b> , <b>y2</b> , and <b>y3</b> (PRGM CTL menu)
):GrSt1(3,7)	
:2→xRes	Store <b>2</b> to the window variable <b>xRes</b> (GRAPH WIND menu)
:ZTrig	Set the viewing window variables (GRAPH ZOOM menu)
:Trace	Display the graph, activate trace cursor, and pause (GRAPH menu)

### Breaking (Interrupting) a Program

To break (interrupt) the program, press **[ON]**. The ERROR 06 BREAK menu is displayed.

- ◆ To display the program editor where the interruption occurred, select **GOTO** (**[F1]**).
- ◆ To return to the home screen, select **QUIT** (**[F5]**).

## Working with Programs

### Managing Memory and Deleting a Program

To check whether adequate memory is available for a program you want to enter or download, display the Check RAM screen (**2nd** **[MEM]** **[F1]**; Chapter 17). To increase available memory, consider deleting selected items or data types from memory (Chapter 17).

### Editing a Program

After you write a program, you can display it in the program editor and edit any command line.

*The program editor does not display a ↓ to indicate that command lines continue beyond the screen.*

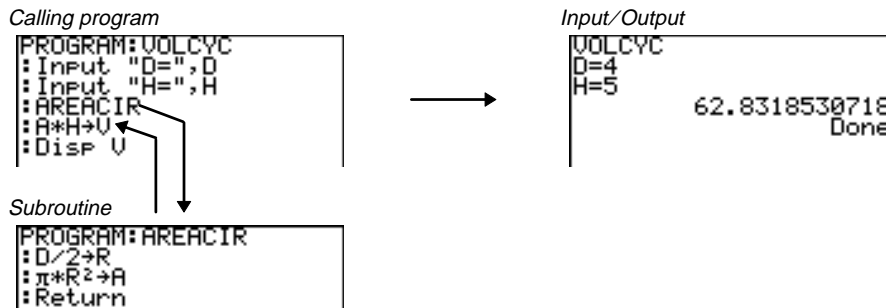
- ❶ Display the program editor (**PRGM** **[F2]**). The PRGM NAMES menu also is displayed.
- ❷ Enter the name of the program you want to edit. Either select the name from the PRGM NAMES menu or enter the individual characters.
- ❸ Edit the program command lines.
  - ◆ Move the cursor to the appropriate location, and then delete, overwrite, or insert characters.
  - ◆ Press **CLEAR** to clear the entire command line, except for the leading colon, and then enter a new program command.
  - ◆ Select program editor menu items **INSc** (**[F5]**) and **DELc** (**MORE** **[F1]**) to insert and delete command lines.

### Calling a Program from Another Program

On the TI-86, any stored program can be called from another program as a subroutine. In the program editor, enter the subroutine program name on a command line by itself.

- ◆ Press **[PRGM]** to display the PRGM NAMES menu, and then select the program name.
- ◆ Use ALPHA keys and alpha keys to enter the program name's individual characters.

When the program name is encountered as the calling program runs, the next command executed is the first command in the subroutine. It returns to the next command in the calling program when it encounters **Return** (or implied **Return**) at the end of a subroutine.



*label* used with **Goto** and **Lbl** is local to the program where it is located. *label* in one program is not recognized by another program. You cannot use **Goto** to branch to a *label* in another program.

## Copying a Program to Another Program Name

- ❶ Display a new or existing program in the program editor.
- ❷ Move the cursor to the command line on which you want to copy a program.
- ❸ Display the **Rcl** prompt ( $\boxed{2nd}$  [RCL]).
- ❹ Enter the name of the program you want to copy. Either select the name from the PRGM NAMES menu or enter individual characters.
- ❺ Press  $\boxed{ENTER}$ . The contents of the recalled program name are inserted into the other program at the cursor location.

## Using and Deleting Variables within a Single Program

If you want to use variables within a program but do not need them after the program is run, you can use **DelVar** within the program to delete the variables from memory.

The program segment to the right uses the variables A and B as counters and then deletes them from memory.

```

:3→B
:For (A,1,100,1)
:B+A→B
:End
:Disp A
:Disp B
:DelVar(A)
:DelVar(B)

```

## Running an Assembly Language Program

An assembly language program is a program that runs much faster and has greater control of the calculator than the regular programs described in this chapter. You can download and run TI-created assembly language programs to add features to your TI-86 that are not built in. For example, you can download the TI-83 finance or inferential statistics features to use on your TI-86.

TI assembly language programs and other programs are available on TI's World Wide Web site:  
**<http://www.ti.com/calc>**

When you download an assembly language program, it is stored among the other programs as a PRGM NAMES menu item. You can:

- ◆ Transmit it using the TI-86 communication link (Chapter 18).
- ◆ Delete it using the MEM DELETE:PRGM screen (Chapter 17).
- ◆ Call it from another program as a subroutine (page 224).

To run an *assemblyProgramName*, the syntax is: **Asm(*assemblyProgramName*)**

If you write an assembly language program, use the two instructions below from the CATALOG.

**AsmComp**(*AsciiAssemblyPrgmName*, *HexAssemblyPrgmName*)      Compiles an assembly language program written in ASCII and stores the hex version

**AsmPrgm**      Identifies an assembly language program; must be entered as the first line of an assembly language program

## Entering and Storing a String

A string is a sequence of characters that you enclose within quotation marks.

- ◆ A string defines characters to be displayed in a program.
- ◆ A string accepts input from the keyboard in a program.

*You do not use quotation marks to enter a string name. In concatenation, you can substitute *stringName* for any "string".*

To enter a string directly, the syntax is:

*"string"*

To concatenate (join together) two or more strings, use  $\oplus$ . The syntax is:

*"stringA"+"stringB"+"stringC"+...*

## The STRNG (String) Menu 2nd [STRNG]

"	sub	Ingh	EqSt	StEq
---	-----	------	------	------

" also marks the start and end of a formula to be attached to a list; it is also an item on the list editor menu (Chapter 11).

"string"

**sub**("string",begin,length)

**sub**(stringName,begin,length)

**Ingh** "string" or **Ingh** stringName

**EqSt**(equationVariable,stringName)

**StEq**(stringName,equationVariable)

Marks the start and end of *string*

Returns a subset of "string" or *stringName*, starting at *begin* character place and *length* characters long

Returns the number of characters in "string" or *stringName*

Converts *equationVariable* contents to *stringName*

Converts *stringName* to *equationVariable*

## Creating a String

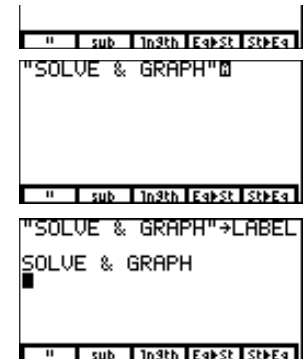
Begin these steps on a blank line on the home screen or in the program editor.

- 1 Display the STRNG menu.
- 2 Enter the open quotation mark, then the string **SOLVE & GRAPH**, and then the close quotation mark.
- 3 Store the string to the string variable name **LABEL**.

2nd [STRNG]

[F1] [ALPHA] [ALPHA]  
 [S] [O] [L] [V] [E] [L]  
2nd [CHAR] [F1] [F3] [L]  
 [G] [R] [A] [P] [H]  
2nd [STRNG] [F1]

[ALPHA] [STO▶]  
 [L] [A] [B] [E] [L]  
 [ENTER]



To evaluate the contents of a string, you must use **StEq** to convert it to an equation.

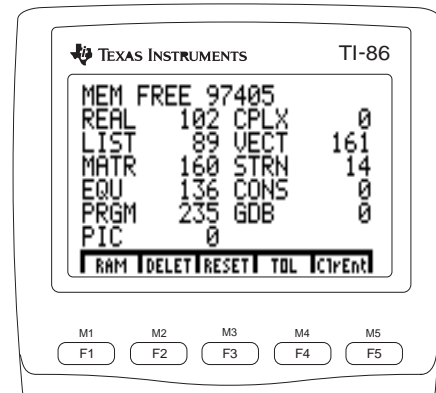




# 17

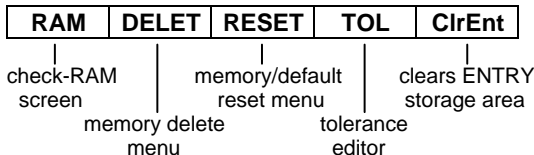
## Memory Management

Checking Available Memory .....	230
Deleting Items from Memory .....	231
Resetting the TI-86 .....	232



## Checking Available Memory

### The MEM (Memory) Menu [2nd] [MEM]



For information on TOL (the tolerance editor), refer to the Appendix.

### Checking Memory Usage [2nd] [MEM] [F1]

When all memory is cleared and all defaults are set, the standard TI-86 has **98,224** bytes of available random-access memory (RAM). As you store information to RAM, you can monitor memory allocation on the Check RAM screen.

MEM FREE reports the total number of bytes available in RAM. Conversely, all other numbers on the screen report the number of bytes that each data type currently occupies. For example, if you were to store a 50-byte matrix in memory, the **MATR** total would increase to **50** bytes, while the **MEM FREE** total would decrease by 50 to **98174** bytes.

MEM FREE	98224		
REAL	19	CPLX	0
LIST	39	VECT	0
MATR	0	STRN	0
EQU	0	CONS	0
PRGM	18	GDB	0
PIC	0		
RAM	DELET	RESET	TOL ClrEnt

To display the number of bytes that a specific variable occupies, display the DELETE screen for that data type (page 231). Scroll the screen, if necessary.

## Deleting Items from Memory

### The MEM DELET (Delete) Menu [2nd] [MEM] [F2]

ALL	REAL	CPLX	LIST	VECTR	▶	MATRX	STRNG	EQU	CONS	PRGM
						▶	GDB	PIC		

To delete a parametric equation, delete the **xt** component.

In the example, the equation  $y5=x^3-x^2+4x-1$  is deleted.

To move directly to the first item beginning with any letter, enter that letter; ALPHA-lock is on.

Each MEM DELET menu item displays the deletion screen for that data type. For example, when you select **LIST**, the MEM DELETE:LIST screen is displayed. Use the DELETE screens to delete any user-created variable and the information stored to it.

- 1 Select **DELET** from the MEM menu to display the MEM DELET menu. [2nd] [MEM] [F2]
- 2 Select the data type of the item you want to delete. To scroll down to the next six items or up to the previous six items, select **PAGE↓** or **PAGE↑**. [MORE] [F3]
- 3 Move the selection cursor (▶) to the item you want to delete (**y5**). The uppercase items are in alphanumeric order, followed by the lowercase items in alphanumeric order. ▼ ▼ ▼
- 4 Delete the item. To delete other items on the screen, repeat steps 3 and 4. [ENTER]

DELETE: EQU	
▶ y1	14 EQU
y2	14 EQU
y3	14 EQU
y4	14 EQU
y5	14 EQU
y6	33 EQU
PAGE↓ PAGE↑	

DELETE: EQU	
y1	14 EQU
y2	14 EQU
y3	14 EQU
▶ y4	14 EQU
PAGE↓ PAGE↑	

## Resetting the TI-86

The **MEM RESET (Reset) Menu** 2nd [MEM] F3

RAM	DELET	RESET	TOL	ClrEnt
ALL	MEM	DFLTS		

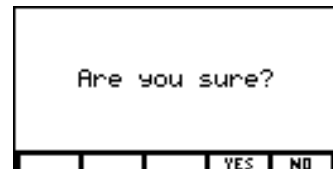
Before resetting all memory, consider deleting selected information to increase memory capacity (page 231).

- ALL** When confirmed, all data is cleared and memory is reset; both messages are displayed
- MEM** When confirmed, clears all stored data from memory; **Mem Cleared** is displayed
- DFLTS** When confirmed, resets all defaults; **Defaults Set** is displayed

When you select and confirm **ALL** or **DFLTS**, the default contrast is reset; to adjust it, use 2nd ▲ or 2nd ▼ (Chapter 1).

When you select **ALL**, **MEM**, or **DFLTS**, a confirmation menu is displayed.

- ◆ To confirm the selected reset, select **YES** (press F4).
- ◆ To cancel the selected reset, select **NO** (press F5).



**ClrEnt (Clear Entry)** 2nd [MEM] F5

The TI-86 retains as many previous entries as possible in ENTRY, up to a capacity of 128 bytes.

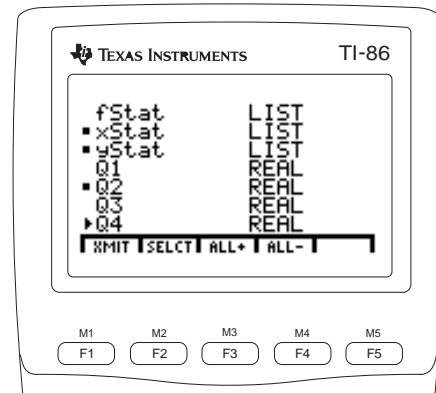
To clear the ENTRY storage area of all entries, execute **ClrEnt** on a blank line on the home screen (2nd [MEM] F5 ENTER).



# 18

## The TI-86 Communication Link

TI-86 Linking Options .....	234
Connecting the TI-86 to Another Device .....	235
Selecting Data to Send.....	236
Preparing the Receiving Device .....	240
Transmitting Data .....	240
Receiving Transmitted Data.....	240



## TI-86 Linking Options

Using the unit-to-unit cable included with the TI-86, you can transmit data between the TI-86 and several other devices.

### Linking Two TI-86s

You can link two TI-86 units and select the data types to be transmitted, including programs. You can back up the entire memory of a TI-86 onto another TI-86.

### Linking a TI-86 and a TI-85

You can select the data types, including programs, to transfer from a TI-85 to a TI-86. You can send most variables and programs from a TI-86 to a TI-85 using **SND85** (page 239), except lists, vectors, or matrices that exceed TI-85 capacity.

When you run a TI-85 program on a TI-86, the TI-85 **PrtScrn** program instruction is not valid. Also, the EOS implied multiplication on the TI-86 differs from the TI-85 (Appendix). For example, the TI-85 interprets **sin 2x** as **sin (2x)**; the TI-86 interprets **sin 2x** as **(sin 2)x**.

### Linking a TI-86 and a CBL 2/CBL or CBR System

The Calculator-Based Laboratory™ (CBL 2™/CBL™) and Calculator-Based Ranger™ (CBR™) systems are optional TI accessories that collect data from physical occurrences, such as science experiments. The CBL 2/CBL and CBR store data to lists, which you can transmit to a TI-86 and analyze. You can transmit list names to a CBL 2/CBL or CBR from a TI-86.

### Linking a TI-86 and a PC or Macintosh

TI-86 TI-GRAPH LINK™ is an optional system that links a TI-86 with an IBM®-compatible or Macintosh® computer.

### Downloading Programs from the Internet

If you have TI-GRAPH LINK and internet services, you can download programs from TI's World Wide Web site at:

**<http://www.ti.com/calc>**

You can download various programs from TI's web site, including assembly language programs that add features such as TI-83 finance and inferential statistics. The site also links to many other TI-86 web sites maintained by user groups, high schools, universities, and individuals.

## Connecting the TI-86 to Another Device

Before you begin to transmit data to or from the TI-86, connect it to the other device.

- 1 Firmly insert one end of the unit-to-unit cable into the port on the bottom edge of the calculator.
- 2 Firmly insert the other end of the cable into the other device (or PC adapter).

**The LINK Menu** [2nd] [LINK]

SEND	RECV	SND85		
------	------	-------	--	--

menu of data types to send  
 receive mode (waiting)  
 menu of data types to send to a TI-85

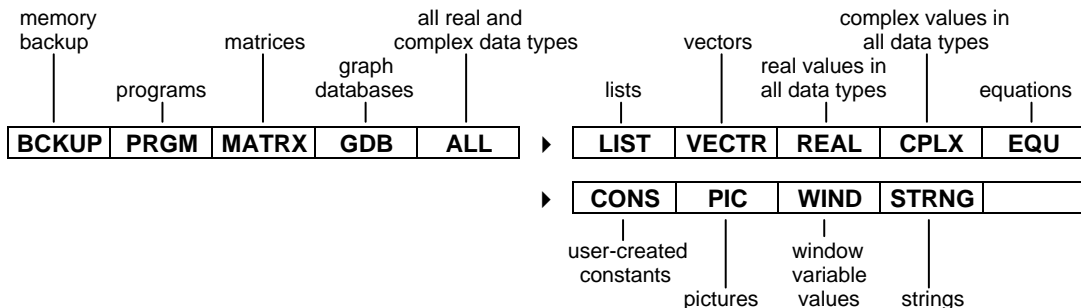
The link menus are not available in the program editor.

**Selecting Data to Send**

To list the variables for a specific data type on a selection screen, select the data type from the LINK SEND menu. When you select **BCKUP**, the message **Memory Backup** is displayed.

The CBL 2/CBL, CBR, and TI-86 TI-GRAPH LINK have built-in Silent Link, which eliminates the need for you to set up the devices to send or receive.

**The LINK SEND Menu** [2nd] [LINK] [F1]





### Initiating a Memory Backup

To initiate a memory backup, select **BCKUP** from the LINK SEND menu (2nd [LINK] [F1] [F1]). The screen to the right is displayed.



To complete memory backup, prepare the other unit to receive data transmission (page 239), and then select **XMIT** from the memory backup menu ([F1]).

**Warning:** When you transmit **BCKUP**, the transmitted memory overwrites all memory in the receiving unit; all information in the memory of the receiving unit is lost. To cancel initiation of a memory backup, press [EXIT].

*If a transmission error occurs during a backup, the receiving-calculator memory is reset.*

As a safety check to prevent accidental loss of memory, when the receiving calculator is notified of an incoming backup transmission, it displays the warning message and confirmation menu, as shown in the screen to the right.



- ◆ To continue the backup transmission, select **CONT**. The backup transmission continues, replacing all receiving-calculator memory with the backup data.
- ◆ To cancel backup and retain all receiving-calculator memory, select **EXIT**.

### Selecting Variables to Send

*If no data of the type you select is stored in memory, the message is displayed:*  
**NO VARS OF THIS TYPE.**

When you select any LINK SEND menu item, except **BCKUP** or **WIND**, each variable of the selected data type is listed in alphanumeric order on a selection screen. The screen to the right is the SEND ALL screen (2nd [LINK] [F1] [F5]).

- ◆ The data type of each variable is specified.
- ◆ Small squares indicate that **xStat**, **yStat**, and **Q2** are selected to be sent.
- ◆ The selection cursor is next to **Q4**.

fStat	LIST
▪ xStat	LIST
▪ yStat	LIST
Q1	REAL
▪ Q2	REAL
Q3	REAL
▶ Q4	REAL
XMIT SELECT ALL+ ALL-	

To select a specific variable to be sent, use  $\downarrow$  and  $\uparrow$  to move the selection cursor next to the variable, and then select **SELCT** (F2) from the selection screen menu.

- ◆ To select all variables of this type, select **ALL+** from the selection screen menu (F3).
- ◆ To deselect all variables of this type, select **ALL-** from the selection screen menu (F4).

To complete transmission of the selected variables, prepare the other unit to receive data transmission (page 239), and then select **XMIT** from the selection screen menu (F1).

### The SEND WIND (Window Variables) Screen

When you select **WIND** from the LINK SEND menu (2nd [LINK] [F1] [MORE] [MORE] [F3]), the SEND WIND screen is displayed. Each SEND WIND screen item represents the window variables, format settings, and any other graph-screen data for that TI-86 graphing mode and for **ZRCL** (user-created zoom). The screen to the right shows that the graph screen data for **Func** and **DifEq** graphing modes are selected.

▪ Func	WIND
Pol	WIND
Param	WIND
♦ DifEq	WIND
ZRCL	WIND
XMIT SELECT ALL+ ALL-	

**Func** Select to send **Func** graphing mode window variable values and format settings

- Pol** Select to send **Pol** graphing mode window variable values and format settings
- Param** Select to send **Param** graphing mode window variable values and format settings
- DifEq** Select to send **DifEq** graphing mode window variable values, **difTol**, axes settings, and format settings
- ZRCL** Select to send user-created zoom window variables, and format settings in any mode

To complete transmission of the selected variables, prepare the other unit to receive data transmission (below), and then select **XMIT** from the memory backup menu (**F1**).

### Sending Variables to a TI-85

The steps for selecting variables to send to a TI-85 are the same as those for selecting variables to send to a TI-86. However, the LINK SND85 menu has fewer items than the LINK SEND menu.

The TI-86 has more capacity for lists, vectors, and matrices than the TI-85. If you send to the TI-85 a list, vector, or matrix that has more elements than the TI-85 allows, the elements that exceed TI-85 capacity are truncated.

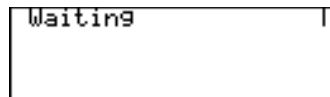
### The LINK SND85 (Send Data to TI-85) Menu **2nd** **[LINK]** **F3**

MATRX	LIST	VECTR	REAL	CPLX	▶	CONS	PIC	STRNG		
-------	------	-------	------	------	---	------	-----	-------	--	--

To prepare a PC to receive data, consult the TI-GGRAPH LINK guidebook.

## Preparing the Receiving Device

To prepare a TI-86 or TI-85 to receive data transmission, select **RECV** from the LINK menu (2nd [LINK] [F2]). The message **Waiting** and the busy indicator are displayed. The calculator is ready to receive transmitted items.



```
Waiting |
```

To cancel receive mode without receiving items, press [ON]. When the **LINK TRANSMISSION ERROR** message is displayed, select **EXIT** from the menu ([F1]). The LINK menu is displayed.

## Transmitting Data

After you select data types on the sending unit and prepare the receiving unit to receive data, you can begin transmitting.

To begin transmitting, select **XMIT** on the selection screen menu of the sending calculator ([F1]).

To interrupt transmission, press [ON] on either calculator. When the **LINK TRANSMISSION ERROR** message is displayed, select **EXIT** from the menu ([F1]). The LINK menu is displayed.

## Receiving Transmitted Data

As the TI-86 receives transmitted data, each variable name and data type is displayed line by line. If all selected items are transmitted successfully, the message **Done** is displayed. To scroll the transmitted variables, press [▼] and [▲].

During transmission, if a transmitted variable name is stored already in the memory of the receiving calculator, transmission is interrupted. The duplicated variable name, its data type, and the DUPLICATE NAME menu are displayed, as shown in the screen to the right.



To resume or cancel transmission, you must select an item from the DUPLICATE NAME menu.

- RENAM** Displays the **Name=** prompt; enter a unique variable name; press **ENTER** to continue transmission
- OVERW** (overwrite) Replaces data stored to the receiving unit's variable with sent variable data
- SKIP** Does not overwrite the receiving unit's data; attempts to send the next selected variable
- EXIT** Cancels the data transmission

### Repeating Transmission to Several Devices

After transmission is complete, the LINK menu is displayed and all selections remain. You can transmit the same selections to a different TI-86 without having to re-select data.

To repeat a transmission with another device, disconnect the unit-to-unit cable from the receiving unit; connect it to another device; prepare the device to receive data; and then select **SEND**, then **ALL**, and then **XMIT**.

### Error Conditions

A transmission error occurs after a few seconds if:

- ◆ The cable is not connected to the port of the sending calculator.
- ◆ The cable is not connected to the port of the receiving calculator.
- ◆ The receiving unit is not set to receive transmission.
- ◆ You attempt a backup between a TI-86 and a TI-85.

### Insufficient Memory in Receiving Unit

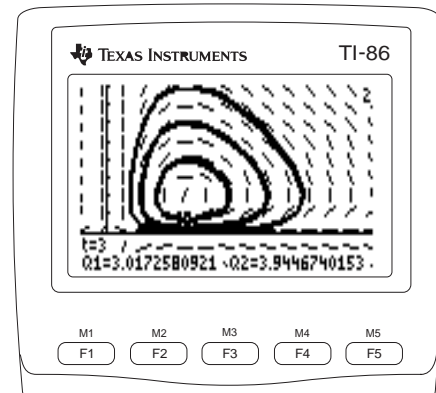
If the receiving unit does not have sufficient memory to receive an item, the receiving unit displays **LINK MEMORY FULL** and the variable name and data type.

- ◆ To skip the variable, select **SKIP**. Transmission resumes with the next item.
- ◆ To cancel transmission altogether, select **EXIT**.

*If the cable is connected but a transmission error occurs, push the cable in more firmly to both calculators and try again.*

# 19 Applications

Using Math Operations with Matrices .....	244
Finding the Area between Curves .....	245
The Fundamental Theorem of Calculus .....	246
Electrical Circuits .....	248
Program: Taylor Series .....	250
Characteristic Polynomial and Eigenvalues .....	252
Convergence of the Power Series .....	254
Reservoir Problem .....	256
Predator-Prey Model .....	258
Program: Sierpinski Triangle .....	260



## Using Math Operations with Matrices

- 1 In the matrix editor, enter matrix **A** as shown.
- 2 On the home screen, select **rref** from the MATRX OPS menu.
- 3 To append a  $3 \times 3$  identity matrix to matrix **A**, select **aug** from the MATRX OPS menu, enter **A**, select **ident** from the MATRX OPS menu, and then enter **3**. Execute the expression.
- 4 Enter **Ans** (to which the matrix from step 3 is stored). Define a submatrix that contains the solution portion of the result. The submatrix begins at element (1,4) and ends at element (3,6).
- 5 Select **►Frac** from the MATH MISC menu and display the fractional equivalent of the submatrix.
- 6 Check the result. Set the decimal mode to 11 (the last 1) Select **round** from the MATH NUM menu for the product of the fractional equivalent of the submatrix times **A**.

```
MATRX:A      3  x3
[0  1  2]
[1  2  1]
[1  1  1]
```

```
rref aug(A,ident 3)
[[1 0 0 .36842105263...
 [0 1 0 -.4736842105...
 [0 0 1 .57894736842...
 ]
```

```
NAMES EDIT MATH OPS CPLX
dim Fill ident rref rref
```

```
Ans(1,4,3,6)►Frac
[[7/19 6/19 -35/19...
 [-9/19 -5/19 45/19...
 [11/19 4/19 -36/19...
 ]
```

```
round(Ans#A,11)
[[1.00000000000 0.00...
 [0.00000000000 1.00...
 [0.00000000000 0.00...
 ]
```

*Displaying the result matrix elements to 11 decimal places illustrates accuracy.*



## Finding the Area between Curves

Find the area of the region bounded by:

$$f(x) = 300x / (x^2 + 625)$$

$$g(x) = 3 \cos(.1x)$$

$$x = 75$$

*If necessary, select ALL- from the equation editor menu to deselect all functions. Also, turn off all stat plots.*

- 1 In **Func** graphing mode, select **y(x)=** from the GRAPH menu to display the equation editor and enter the equations as shown.

$$y1 = 300x / (x^2 + 625) \quad y2 = 3 \cos(.1x)$$

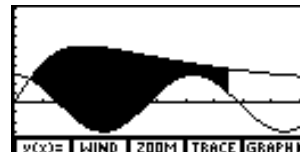
- 2 Select **WIND** from the GRAPH menu and set the window variables as shown.

$$xMin=0 \quad xMax=100 \quad xScl=10 \quad yMin=-5 \quad yMax=10 \quad yScl=1 \quad xRes=1$$

- 3 Select **GRAPH** from the GRAPH menu to display the graph screen.
- 4 Select **ISECT** from the GRAPH MATH menu. Move the trace cursor to the intersection of the functions. Press **ENTER** to select **y1**. The cursor moves to **y2**. Press **ENTER**. Then press **ENTER** again to set the current cursor location as the initial guess. The solution uses the solver. The value of **x** at the intersection, which is the lower limit of the integral, is stored to **Ans** and **x**.

- 5 The area to integrate is between **y1** and **y2**, from **x=5.5689088189** to **x=75**. To see the area on a graph, return to the home screen, select **Shade** from the GRAPH DRAW menu, and execute this expression:

$$\text{Shade}(y2, y1, \text{Ans}, 75)$$



- 6 Select **TOL** from the MEM menu and set **tol=1E-5**.
- 7 On the home screen, compute the integral with **fnInt** (CALC menu). The area is 325.839961998.

$$\text{fnInt}(y1 - y2, x, \text{Ans}, 75)$$

If necessary, select **ALL-** from the equation editor menu to deselect all functions. Also, turn off all stat plots.

In the example,  $\text{nDer}(y2,x)$  only approximates  $y3$ ; you cannot define  $y3$  as  $\text{der1}(y2,x)$ .

## The Fundamental Theorem of Calculus

Consider these three functions:

$$F(x)_1 = (\sin x)/x$$

$$F(x)_2 = \int_0^x (\sin t)/t$$

$$F(x)_3 = \frac{d}{dx} \int_0^x (\sin t)/t dt$$

- 1 In **Func** graphing mode, select **y(x)=** from the GRAPH menu, and then enter the functions and set graph styles in the equation editor as shown. (**fnInt** and **nDer** are CALC menu items.)

$$y1=(\sin x)/x$$

$$y2=fnInt(y1(t),t,0,x)$$

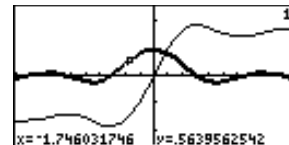
$$y3=nDer(y2,x)$$

- 2 Select **TOL** from the MEM menu to display the tolerance editor. To improve the rate of the calculations, set **tol=0.1** and  **$\delta=0.001$** .

- 3 Select **WIND** from the GRAPH menu and set the window variable values as shown.

$$xMin=-10 \quad xMax=10 \quad xScl=1 \quad yMin=-2.5 \quad yMax=2.5 \quad yScl=1 \quad xRes=4$$

- 4 Select **TRACE** from the GRAPH menu to display the graph and the trace cursor.
- 5 Trace **y1** and **y3** to verify that the graph of **y1** and the graph of **y3** are visually indistinguishable.



The inability to visually distinguish between the graphs of **y1** and **y3** graphically supports the fact that:

$$\frac{d}{dx} \int_0^x (\sin t)/t dt = (\sin x)/x$$

- 6 Deselect **y2** in the equation editor.
- 7 Select **TBLST** from the TABLE menu. Set **TblStart=1**, **ΔTbl=1**, and **Indpnt: Auto**.
- 8 Select **TABLE** from the TABLE menu to display the table.  
Compare the solution of **y1** with the solution of **y3** to numerically support the formula above.

X	Y1	Y3
1	.841471	.8414709
2	.4546487	.4546487
3	.04704	.04704
4	-.189201	-.189201
5	-.191785	-.191785
6	-.046569	-.046569

X=1

TBLST	SELECT	X	Y
-------	--------	---	---

## Electrical Circuits

A measurement device has measured the DC current (C) in milliamperes and voltage (V) in volts on an unknown circuit. From these measurements, you can calculate power (P) in milliwatts using the equation  $CV=P$ . What is the average of the measured power?

With the TI-86, you can estimate the power in milliwatts at a current of 125 milliamperes using the trace cursor, the interpolate/extrapolate editor, and a regression forecast.

- In two consecutive columns of the list editor, store the current measurements shown below to the list name **CURR** and the voltage measurements shown below to the list name **VOLT**.  
 $\{10, 20, 40, 60, 80, 100, 120, 140, 160\} \rightarrow \text{CURR}$   
 $\{2, 4.2, 10, 18, 32.8, 56, 73.2, 98, 136\} \rightarrow \text{VOLT}$
- In the next column of the list editor, enter the list name **POWER**.
- Enter the formula **CURR \* VOLT** in the list editor entry line for **POWER**. Press **ENTER** to calculate the values for power and store the answers to the list name **POWER**.
- Select **WIND** from the GRAPH menu and set the window variable values as shown.  
 $x\text{Min}=0$   $x\text{Max}=\text{max}(\text{POWER})$   $x\text{Scl}=1000$   $y\text{Min}=0$   $y\text{Max}=\text{max}(\text{CURR})$   $y\text{Scl}=10$   $x\text{Res}=4$
- From the home screen, select **FnOff** from the CATALOG and press **ENTER** to deselect all functions in the equation editor. Select **Plot1** from the CATALOG and set up a stat plot with **POWER** on the x-axis and **CURR** on the y-axis.

CURR	VOLT	POWER
10	2	-----
20	4.2	
40	10	
60	18	

POWER = CURR \* VOLT

←	→	NAMES	"	DPS
CURR	POWER	VOLT	rStat	xStat

CURR	VOLT	POWER
10	2	20
20	4.2	84
40	10	400
60	18	1080
80	32.8	2624
100	56	5600

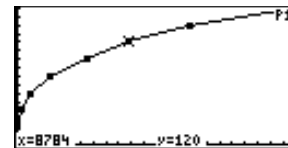
POWER(1) = 20

←	→	NAMES	"	DPS
---	---	-------	---	-----

FnOff	Done
Plot1(2, POWER, CURR, 1)	Done

The 7s and 8s in parentheses specify the 7th and 8th elements of **POWER** and **CURR**.

- 6 Select **TRACE** from the GRAPH menu to display the stat plot and trace cursor on the graph screen.
- 7 Trace the stat plot to approximate the value of **POWER** at **CURR=125**. With this statistical data, the closest to **CURR=125** that you can trace to is **CURR=120** (on the y-axis).



- 8 Select **INTER** from the MATH menu to display the interpolate/extrapolate editor. To interpolate **POWER** at **CURR=125**, enter the nearest pairs:

$x_1 = \text{POWER}(7)$        $y_1 = \text{CURR}(7)$   
 $x_2 = \text{POWER}(8)$        $y_2 = \text{CURR}(8)$

```
INTERPOLATE
x1=8784
y1=120
x2=13720
y2=140
x=10018
y=125
SOLVE
```

- 9 Enter  $y=125$  and solve for  $x$ .
- 10 On the home screen, select **LinR** from the STAT CALC menu to fit the linear regression model equation to the data stored to **POWER** and **CURR**. Write down the value of the result variable **corr**.

```
LinR POWER,CURR
```

To enter each regression after **LinR**, press  $\boxed{2nd}$  [ENTRY] and edit as needed.

- 11 Fit the logarithmic (**LnR**), exponential (**ExpR**), and power (**PwrR**) regressions to the data, writing down the value of **corr** for each regression. Compare the **corr** values of each regression to determine which model fits the data most accurately (the **corr** value closest to 1).
- 12 Execute the most accurate regression again, and then select **FCST** from the STAT menu. To forecast **POWER** at **CURR=125**, enter  $y=125$  and solve for  $x$ .

```
FORECAST:PwrReg
x=9393.6276510757
y=125
SOLVE
```

Compare this answer with the answer returned in step 9.

## Program: Taylor Series

When you run this program, you can enter a function and specify the order and center point. Then the program calculates the Taylor Series approximation for the function and plots the function you entered. This example shows how to call a program from another program as a subroutine.

- 1 Before you enter the program **TAYLOR**, select **EDIT** from the PRGM menu, enter **MOBIUS** at the **Name=** prompt, and then enter this brief program to store the Mobius Series. The program **TAYLOR** calls this program and runs it as a subroutine.

```
PROGRAM:MOBIUS
:{1,-1,-1,0,-1,1,-1,0,0,1,-1,0,-1,1,1,0,-1,0,-1,0}➤MSERIES
:Return
```

- 2 Select **EDIT** from the PRGM menu, enter **TAYLOR** at the **Name=** prompt, and then enter this program to calculate the Taylor Series.

	PROGRAM:TAYLOR
	:Func:FnoFF
	:y14=pEval(TPOLY,x-center)
	:GrSt1(14,2)
$\epsilon$ is on the CHAR GREEK menu	—— :1E-9➤ $\epsilon$ :.1➤rr
	:ClLCD
User enters equation function	—— :InpSt "FUNCTION: ",EQ
	:St▶Eq(EQ,y13)
User enters order	—— :Input "ORDER: ",order
	:order+1➤dimL TPOLY
	:Fill(0,TPOLY)
User enters center	—— :Input "CENTER: ",center
	:evalF(y13,x,center)➤f0
	:f0➤TPOLY(order+1)

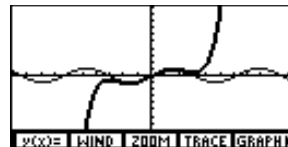
The higher-order derivative values necessary for this program are calculated numerically based on the methods in "Numerical Differentiation of Analytic Functions," J. N. Lyness and C. B. Moler, *SIAM Journal of Numerical Analysis* 4 (1967): 202-210.

```

:If order≥1
:der1(y13,x,center)→TPOLY(order)
:If order≥2
:der2(y13,x,center)/2→TPOLY(order-1)
:If order≥3
Begins Then group — :Then
Calls subroutine — :MOBIUS
Begins For group — :For(N,3,order,1)
:abs f0→gmax:gmax→bmi
:1→m:0→ssum
Begins While group — :While abs bmi≥ε*gmax
Creates nested While group — :While MSERIES(m)==0
: m+1→m
: End
:0→bsum
Creates nested For group — :For(J,1,m*N,1)
: rr*e^(2π(J/(m*N))*(0,1))+(center,0)→x
: real y13→gval
: bsum+gval→bsum
: max(abs gval,gmax)→gmax
: End
: bsum/(m*N)-f0→bmi
: ssum+MSERIES(m)*bmi→ssum
: m+1→m
Ends While group — :End
: ssum/(rr^N)→TPOLY(order+1-N)
Ends For group — :End
Ends Then group — :End
:ZStd

```

- ③ On the home screen, select **TAYLOR** from the PRGM NAMES menu, and then press **ENTER** to run the program.
- ④ When prompted, enter: **FUNCTION: sin x**  
**ORDER: 5**  
**CENTER: 0**



## Characteristic Polynomial and Eigenvalues

- ① In the matrix editor or on the home screen, enter matrix **A** as shown.  

$$\begin{bmatrix} -1 & 2 & 5 \\ 3 & -6 & 9 \\ 2 & -5 & 7 \end{bmatrix} \rightarrow A$$

- ② On the home screen, select **eigV1** from the MATRX MATH menu to find the complex eigenvalues for the matrix **A** and store them to the list name **EV**.

- ③ Graph the characteristic polynomial  $C_p(x)$  of matrix **A** without knowing the analytic form of  $C_p(x)$  based on the formula  $C_p(x) = \det(A - xI)$ . In **Func** graphing mode, select **y(x)=** from the GRAPH menu and enter the function in the equation editor as shown.

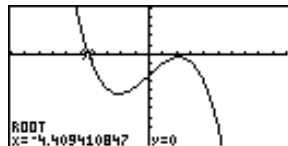
$\backslash y1 = \det(A - x \cdot \text{ident } 3)$

- ④ Select **WIND** from the GRAPH menu and set the window variable values as shown.

**xMin=-10**   **xMax=10**   **xScl=1**   **yMin=-100**   **yMax=50**   **yScl=10**   **xRes=4**

- ⑤ Select **ROOT** from the GRAPH MATH menu and use it to display the real eigenvalue interactively. Use **Left Bound=-5**, **Right Bound=4**, and **Guess=-4.5**.

Compare the root (**x** value) you displayed interactively with the first element of the result list in step 2.



*The first eigenvalue is real, since the imaginary part is 0.*

*If necessary, select **ALL-** from the equation editor menu to deselect all functions. Also, turn off all stat plots.*



Next, use the list editor and a degree-three polynomial regression to find an analytic formula in terms of  $x$  for the characteristic polynomial  $y1=\det(A-x*\text{ident } 3)$ . Create two lists that you can use to find the analytic formula.

- 6 In the list editor, create elements for **xStat** by entering the expression **seq(N,N,-10,21)** in the **xStat** entry line. **seq** is on the MATH MISC menu.
- 7 Create elements for **yStat** by attaching the formula "**y1(xStat)**" to **yStat** in the entry line. The expression is evaluated when you press **ENTER** or exit the list editor.
- 8 On the home screen, select **Plot1** (from the CATALOG and execute **Plot1(2,xStat,yStat,1)** to turn on **Plot1** as an xyLine plot using the lists **xStat** and **yStat**.
- 9 Select **GRAPH** from the GRAPH menu to display **Plot1** and **y1** on the graph screen.

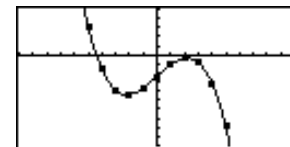
To clear the menus from the graph screen, press **CLEAR**.

- 10 On the home screen, select **P3Reg** from the STAT CALC menu. Execute **P3Reg xStat,yStat,y2** to find the explicit characteristic polynomial in terms of  $x$  and store it to **y2**.

The cubic regression coefficients stored in the result list **PRegC** suggest that  $a=-1$ ,  $b=0$ ,  $c=14$ , and  $d=-24$ . So the characteristic polynomial seems to be  $Cp(x)=-x^3+14x-24$ .

xStat	yStat	fStat
-----	-----	-----
xStat =seq(N,N,-10,21)		

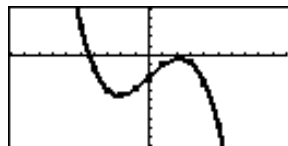
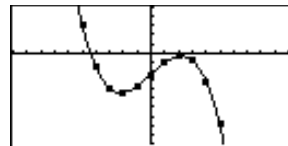
xStat	yStat	fStat	2
-10	-----	-----	
-9	-----	-----	
-8	-----	-----	
-7	-----	-----	
yStat = "y1(xStat)"			
← > NAMES " OPS			
EV fStat xStat yStat			



P3Reg xStat,yStat,y2

```
CubicReg
y=ax^3+bx^2+cx+d
n=32
PRegC=
(-1 -1E-12 14 -23.99...
```

- ⑪ Support this conjecture by graphing **y1**, **y2** (to which  $C_p(x)$  is stored), and **Plot1** together.
- ⑫ In the equation editor, enter the apparent characteristic polynomial of matrix **A** and select  $\overline{\text{■}}$  (thick) graph style as shown.
- $\overline{\text{■}} y_3 = x^3 + 14x - 24$
- ⑬ Graph **y1**, **y2**, **y3**, and **Plot1**.



- ⑭ Deselect **y2** in the equation editor.
- ⑮ Select **TABLE** from the TABLE menu to display **y1** and **y3** in the table.

Compare the values for the characteristic polynomial.

x	y1	y3
-3.2	-3.2	-3.2
-7.9	-7.9	-7.9
-15.6	-15.6	-15.6
-2.89	-2.89	-2.89
-4.24	-4.24	-4.24
-6.27	-6.27	-6.27

$y_3 = x^3 + 14x - 24$

TABLE SELECT x y

## Convergence of the Power Series

A closed-form analytic antiderivative of  $(\sin x)/x$  does not exist. However, substituting  $t$  for  $x$ , you can find an infinite series analytic solution by taking the series definition of  $\sin t$ , dividing each term of the series by  $t$ , and then integrating term by term to yield:

$$\sum_{n=1}^{\infty} -1^{n+1} t^{2n-1} / ((2n-1)(2n-1)!)$$

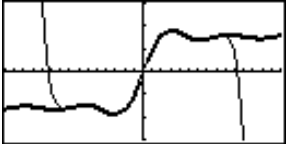
Plot finite approximations of this power series solution on the TI-86 with **sum** and **seq**.

If necessary, select **ALL-** from the equation editor menu to deselect all functions. Also, turn off all stat plots.

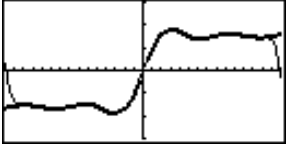
This example is set up in **Param** mode, which allows you to control the solution with **tStep** and increase plotting speed.

To clear the menus from the graph screen, press **[CLEAR]**.

- 1 Select **TOL** from the MEM menu and set **tol=1**.
- 2 On the mode screen, set **Radian** angle mode and **Param** graphing mode.
- 3 In the equation editor, enter the parametric equations for the power series approximation as shown. Select **sum** and **seq** from the LIST OPS menu. Select **!** from the MATH PROB menu.
 
$$\text{xt1=t} \qquad \text{yt1=sum seq((-1)^(j+1)t^(2j-1)/((2j-1)(2j-1)!),j,1,10,1)$$
- 4 In the equation editor, enter the parametric equations as shown to plot the antiderivative of  $(\sin x)/x$  and compare it with the plot of the power series approximation. (Select **fnInt** from the CALC menu.)
 
$$\text{xt2=t} \qquad \text{yt2=fnInt}((\sin w)/w,w,0,t)$$
- 5 Select **WIND** from the GRAPH menu and set the window variable values as shown.
 

<b>tMin=-15</b>	<b>xMin=-15</b>	<b>yMin=-3</b>
<b>tMax=15</b>	<b>xMax=15</b>	<b>yMax=3</b>
<b>tStep=0.5</b>	<b>xScl=1</b>	<b>yScl=1</b>
- 6 Select **FORMT** from the GRAPH menu and set **SimulG** format.
- 7 Select **GRAPH** from the GRAPH menu to plot the parametric equations on the graph screen.
 
- 8 In the equation editor, modify **yt1** to compute the first 16 terms of the power series by changing **10** to **16**. Plot the equations again.
 

In this example, the window variable **tStep** controls the plotting speed. Select **WIND** from the GRAPH menu and set **tStep=1** and observe the difference in plotting speed and curve smoothness.



## Reservoir Problem

On the TI-86, you can use parametric graphing animation to solve a problem.

Consider a water reservoir with a height of 2 meters. You must install a small valve on the side of the reservoir such that water spraying from the open valve hits the ground as far away from the reservoir as possible. At what height should you install the valve to maximize the length of the water stream when the valve is wide open?

Assume a full tank at time=0, no acceleration in the x direction, and no initial velocity in the y direction. Also, ignore valve-size and valve-type factors. Integrating the definition of acceleration in both the x and y directions twice yields the equations  $x=v_0t$  and  $y=h_0-(gt^2)/2$ . Solving Bernoulli's equation for  $v_0$  and substituting into  $v_0t$  results in this pair of parametric equations:

$$xt=t\sqrt{2g(2-h_0)} \quad yt=h_0-(gt^2)/2$$

t = time in seconds

$h_0$  = height of the valve in meters

g = the built-in acceleration of gravity constant

When you graph these equations on the TI-86, the y-axis ( $x=0$ ) is the side of the reservoir where the valve is to be installed. The x-axis ( $y=0$ ) is the ground. Each plotted parametric equation represents the water stream when the valve is at each of several heights.

If necessary, select **ALL-** from the equation editor menu to deselect all functions. Also, turn off all stat plots.

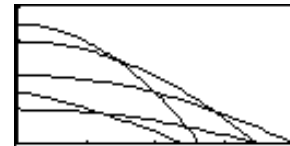
- 1 In **Param** graphing mode, select **E(t)=** from the GRAPH menu and enter the equations in the equation editor as shown. This pair of equations plots the path of the water stream when the valve is installed at a height of 0.5 meters.

$$x_1 = t\sqrt{2g(2-0.5)}$$

$$y_1 = 0.5 - (g \cdot t^2) / 2$$

- 2 Move the cursor to **x2=**. Press  $\boxed{2nd}$   $\boxed{RCL}$   $\boxed{F2}$   $\boxed{ALPHA}$  **1**, and press  $\boxed{ENTER}$  to recall the contents of **x1** into **x2**. For **x2**, change the valve height (which is **0.5**) to **0.75** meters. Do the same with **y1** and **y2**.
  - 3 Repeat step 3 to create three more pairs of equations. Change the valve height to **1.0** meters for **x3** and **y3**, **1.5** meters for **x4** and **y4**, and **1.75** meters for **x5** and **y5**.
  - 4 Select **WIND** from the GRAPH menu and set the window variable values as shown.
 

<b>tMin=0</b>	<b>xMin=0</b>	<b>yMin=0</b>
<b>tMax=<math>\sqrt{4/g}</math></b>	<b>xMax=2</b>	<b>yMax=2</b>
<b>tStep=0.01</b>	<b>xScl=0.5</b>	<b>yScl=0.5</b>
  - 5 Select **FORMT** from the GRAPH menu and set **SimulG** graph format.
  - 6 Select **GRAPH** from the GRAPH menu to plot the trajectory of the water jets from the five specified heights.
- Which height seems to create the longest water stream?



To clear the menus from the graph screen, press  $\boxed{CLEAR}$ .

## Predator-Prey Model

The growth rates of predator and prey populations, such as foxes and rabbits, depend upon the populations of both species. This initial-value problem is a form of the predator-prey model.

$$F' = -F + 0.1F * R \qquad R' = 3R - F * R$$

**Q1** = population of foxes (F)

**Q2** = population of rabbits (R)

**QI1** = initial population of foxes (2)

**QI2** = initial population of rabbits (5)

Find the population of foxes and rabbits after 3 months ( $t=3$ ).

- 1 In **DifEq** graphing mode, select **Q't=** from the GRAPH menu and enter the functions and set graph styles in the equation editor as shown.

$$\text{\textbackslash}Q'1 = -Q1 + 0.1Q1 * Q2 \qquad \text{\textbackslash}Q'2 = 3Q2 - Q1 * Q2$$

- 2 Select **FORMT** from the GRAPH menu and set **FldOff** field format.
- 3 Select **WIND** from the GRAPH menu and set the window variable values as shown.

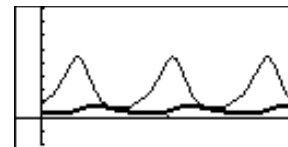
<b>tMin=0</b>	<b>xMin=-1</b>	<b>yMin=-10</b>
<b>tMax=10</b>	<b>xMax=10</b>	<b>yMax=40</b>
<b>tStep=<math>\pi/24</math></b>	<b>xScl=5</b>	<b>yScl=5</b>
<b>tPlot=0</b>		<b>difTol=.001</b>

- 4 Select **INITC** from the GRAPH menu and set the initial conditions as shown.

<b>tMin=0</b>	<b>QI1=2</b>	<b>QI2=5</b>
---------------	--------------	--------------

- 5 Select **GRAPH** from the GRAPH menu to plot the graph of the two populations over time.
- 6 To see the direction field of the phase-plane solution, select **FORMT** from the GRAPH screen, and then set **DirFld** field format.
- 7 Select **INITC** from the GRAPH menu and delete the values for **QI1** and **QI2**.
- 8 Select **GRAPH** from the GRAPH menu to display the direction field of the phase-plane solution.
- 9 To see a family of specific phase-plane solutions on top of the direction field, select **INITC** from the GRAPH menu, and then enter lists for **QI1** and **QI2** as shown.  
 $QI1=\{2,6,7\}$                        $QI2=\{6,12,18\}$
- 10 Select **TRACE** from the GRAPH menu to display the graph with the trace cursor.
- 11 Press **3** to see how many foxes and how many rabbits are alive at  $t=3$ . (Round the values of **Q1** (foxes) and **Q2** (rabbits) to whole numbers.) How many foxes and rabbits are alive at  $t=6$ ? at  $t=12$ ?

On what value of **Q1** and **Q2** do the phase-plane orbits seem to converge? What is the significance of this value?



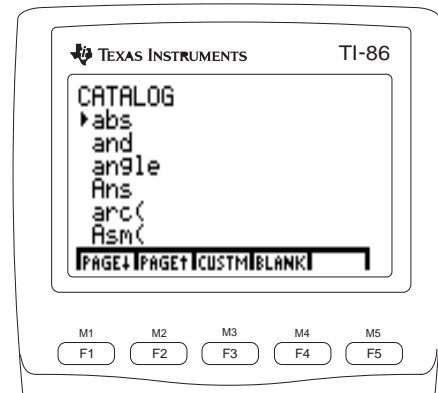




# 20

## A to Z Function and Instruction Reference

Quick-Find Locator..... 262  
Alphabetical Listing of Operations..... 266



## Quick-Find Locator

This section lists the TI-86 functions and instructions in functional groups along with the page numbers where they are described in this chapter.

### Graphing

<b>Axes</b> ( ..... 271	<b>DrInV</b> ..... 287	<b>Line</b> ( ..... 314	<b>RectGC</b> ..... 344	<b>ZFit</b> ..... 373
<b>AxesOff</b> ..... 271	<b>dxDer1</b> ..... 288	<b>Param</b> ..... 333	<b>SeqG</b> ..... 351	<b>ZIn</b> ..... 373
<b>AxesOn</b> ..... 271	<b>dxNDer</b> ..... 288	<b>Pol</b> ..... 336	<b>Shade</b> (..... 352	<b>ZInt</b> ..... 374
<b>Circl</b> ( ..... 273	<b>FldOff</b> ..... 295	<b>PolarGC</b> ..... 336	<b>SimulG</b> ..... 354	<b>ZOut</b> ..... 375
<b>CIDrw</b> ..... 273	<b>FnOff</b> ..... 296	<b>PtChg</b> (..... 338	<b>SlpFld</b> ..... 358	<b>ZPrev</b> ..... 375
<b>CoordOff</b> ..... 275	<b>FnOn</b> ..... 297	<b>PtOff</b> (..... 338	<b>StGDB</b> ..... 361	<b>ZRcl</b> ..... 376
<b>CoordOn</b> ..... 275	<b>Func</b> ..... 299	<b>PtOn</b> (..... 338	<b>StPic</b> ..... 362	<b>ZSqr</b> ..... 376
<b>DifEq</b> ..... 281	<b>GridOff</b> ..... 301	<b>PxChg</b> ( ..... 340	<b>TanLn</b> (..... 366	<b>ZStd</b> ..... 377
<b>DirFld</b> ..... 282	<b>GridOn</b> ..... 302	<b>PxOff</b> (..... 340	<b>Text</b> ( ..... 366	<b>ZTrig</b> ..... 378
<b>DrawDot</b> ..... 285	<b>GrStl</b> ( ..... 302	<b>PxOn</b> ( ..... 340	<b>Trace</b> ..... 367	
<b>DrawF</b> ..... 286	<b>Horiz</b> ..... 304	<b>PxTest</b> (..... 340	<b>Vert</b> ..... 369	
<b>DrawLine</b> ..... 286	<b>LabelOff</b> ..... 310	<b>RcGDB</b> ..... 343	<b>ZData</b> ..... 371	
<b>DrEqu</b> ( ..... 287	<b>LabelOn</b> ..... 310	<b>RcPic</b> ..... 343	<b>ZDecm</b> ..... 372	

### Lists

<b>aug</b> ( ..... 270	<b>→dimL</b> ..... 282	<b>li vc</b> ..... 316	<b>SetLEdit</b> ..... 351	<b>Sorty</b> ..... 359
<b>cSum</b> ( ..... 278	<b>Fill</b> ( ..... 295	<b>prod</b> ..... 338	<b>sortA</b> ..... 359	<b>sum</b> ..... 364
<b>DeltaIst</b> ( ..... 279	<b>Form</b> ( ..... 298	<b>Select</b> ( ..... 350	<b>sortD</b> ..... 359	<b>vc li</b> ..... 369
<b>dimL</b> ..... 282	List entry: { } ..... 316	<b>seq</b> ( ..... 351	<b>Sortx</b> ..... 359	

---

**Mathematics, Algebra, and Calculus**


---

<b>abs</b> .....267	Division: /.....284	Greater than or equal to: $\geq$ .....301	<b>not</b> .....325	<b>round</b> ( .....348
Addition: +.....267	DMS entry: ' .....285	<b>h</b> .....302	Not equal to: $\neq$ .....326	<b>Sci</b> .....349
<b>and</b> .....268	<b>►DMS</b> .....285	<b>Hex</b> .....302	<b>nPr</b> .....326	<b>shftL</b> .....353
<b>angle</b> .....269	<b>dxDer1</b> .....288	<b>►Hex</b> .....303	<b>o</b> .....326	<b>shftR</b> .....353
<b>Ans</b> .....269	<b>dxNDer</b> .....288	<b>imag</b> .....306	<b>Oct</b> .....327	<b>sign</b> .....354
<b>arc</b> (.....269	<b>e^</b> .....288	<b>int</b> .....308	<b>►Oct</b> .....327	<b>simult</b> (.....354
Assignment: = .....270	<b>Eng</b> .....290	<b>inter</b> (.....309	<b>or</b> .....328	<b>sin</b> .....355
<b>b</b> .....271	<b>Eq►St</b> (.....290	Inverse: $^{-1}$ .....309	Percent: %.....334	<b>sin<sup>-1</sup></b> .....355
<b>Bin</b> .....272	Equal: = .....290	<b>iPart</b> .....309	<b>pEval</b> (.....334	<b>sinh</b> .....356
<b>►Bin</b> .....272	Equal to: == .....291	<b>lcm</b> ( .....311	<b>►Pol</b> .....336	<b>sinh<sup>-1</sup></b> .....356
<b>ClrEnt</b> .....273	<b>Euler</b> .....291	Less than: <.....312	<b>PolarC</b> .....336	<b>Solver</b> (.....358
<b>CITbl</b> .....273	<b>eval</b> .....291	Less than or equal to: $\leq$ .....312	Polar complex: $\angle$ .....336	Square: $^2$ .....360
<b>conj</b> .....275	<b>evalF</b> (.....292	<b>In</b> .....316	<b>poly</b> .....337	Square root: $\sqrt{\quad}$ .....360
<b>cos</b> .....276	Exponent: <b>E</b> .....292	<b>log</b> .....318	Power: $^{\quad}$ .....337	<b>St►Eq</b> (.....361
<b>cos<sup>-1</sup></b> .....276	Factorial: ! .....294	<b>max</b> ( .....319	Power of 10: <b>10<sup>^</sup></b> .....337	Store to variable: <b>►</b> .....362
<b>cosh</b> .....277	<b>Fix</b> .....295	<b>min</b> (.....320	<b>Radian</b> .....341	Subtraction: -.....363
<b>cosh<sup>-1</sup></b> .....277	<b>Float</b> .....295	<b>mod</b> (.....320	Radian entry: ' .....341	<b>tan</b> .....364
<b>d</b> .....278	<b>fMax</b> (.....296	Multiplication: * .....321	<b>real</b> .....343	<b>tan<sup>-1</sup></b> .....365
<b>Dec</b> .....278	<b>fMin</b> (.....296	<b>nCr</b> .....322	<b>►Rec</b> .....343	<b>tanh</b> .....365
<b>►Dec</b> .....279	<b>fInt</b> ( .....296	<b>nDer</b> (.....323	<b>RectC</b> .....344	<b>tanh<sup>-1</sup></b> .....365
<b>Degree</b> .....279	<b>fPart</b> .....298	Negation: - .....323	<b>RK</b> .....345	<b>xor</b> .....370
Degree entry: $^{\circ}$ .....279	<b>►Frac</b> .....298	<b>Normal</b> .....324	Root: $\sqrt[x]{\quad}$ .....346	
<b>der1</b> ( .....280	<b>gcd</b> (.....299		<b>rotL</b> .....347	
<b>der2</b> ( .....280	Greater than: >.....300		<b>rotR</b> .....347	

---

**Matrices**


---

<b>aug</b> ( ..... 270	<b>&gt;dim</b> ..... 281	<b>LU</b> (..... 318	<b>rAdd</b> (..... 340	<b>rSwap</b> ( ..... 348
<b>cnorm</b> ..... 273	<b>eigVc</b> ..... 289	Matrix entry: [ ] ... 319	<b>randM</b> ( ..... 342	Transpose: $T$ ..... 367
<b>cond</b> ..... 274	<b>eigVl</b> ..... 289	<b>mRAdd</b> (..... 321	<b>ref</b> ..... 344	
<b>det</b> ..... 281	<b>Fill</b> ( ..... 295	<b>multR</b> (..... 322	<b>rnorm</b> ..... 346	
<b>dim</b> ..... 281	<b>ident</b> ..... 304	<b>norm</b> ..... 323	<b>rref</b> ..... 348	

---

**Programming**


---

<b>Asm</b> ( ..... 269	<b>DispT</b> ..... 284	<b>Get</b> ( ..... 299	<b>Input</b> ..... 307	<b>Prompt</b> ..... 338
<b>AsmComp</b> (..... 270	<b>DS&lt;</b> ( ..... 288	<b>getKey</b> ..... 300	<b>IS&gt;</b> (..... 310	<b>Repeat</b> ..... 345
<b>AsmPrgm</b> ..... 270	<b>Else</b> ..... 290	<b>Goto</b> ..... 300	<b>Lbl</b> ..... 311	<b>Return</b> ..... 345
<b>CILCD</b> ..... 273	<b>End</b> ..... 290	<b>lAsk</b> ..... 304	<b>LCust</b> (..... 311	<b>Send</b> ( ..... 350
<b>DelVar</b> ( ..... 280	Equal: = ..... 290	<b>lAuto</b> ..... 304	<b>Menu</b> (..... 320	<b>Stop</b> ..... 362
<b>Disp</b> ..... 283	Equal to: == ..... 291	<b>lf</b> ..... 305	<b>Outpt</b> (..... 329	<b>Then</b> ..... 366
<b>DispG</b> ..... 283	<b>For</b> (..... 297	<b>lnpSt</b> ..... 307	<b>Pause</b> ..... 333	<b>While</b> ..... 369

---

**Statistics**


---

<b>Box</b> ..... 272	<b>LnR</b> ..... 317	<b>PIOn</b> ..... 334	<b>randInt</b> ( ..... 342	<b>SinR</b> ..... 357
<b>ExpR</b> ..... 293	<b>MBox</b> ..... 319	<b>Plot1</b> (..... 335	<b>randM</b> ( ..... 342	<b>Sortx</b> ..... 359
<b>fcstx</b> ..... 294	<b>OneVar</b> ..... 327	<b>Plot2</b> (..... 335	<b>randNorm</b> ( ..... 342	<b>Sorty</b> ..... 359
<b>fcsty</b> ..... 294	<b>P2Reg</b> ..... 330	<b>Plot3</b> (..... 335	<b>Scatter</b> ..... 349	<b>StReg</b> (..... 362
<b>Hist</b> ..... 303	<b>P3Reg</b> ..... 331	<b>PwrR</b> ..... 339	<b>Select</b> (..... 350	<b>TwoVar</b> ..... 368
<b>LgstR</b> ..... 313	<b>P4Reg</b> ..... 332	<b>rand</b> ..... 341	<b>SetLEdit</b> ..... 351	<b>xyline</b> ..... 370
<b>LinR</b> ..... 315	<b>PIOff</b> ..... 334	<b>randBin</b> (..... 341	<b>ShwSt</b> ..... 354	

**Strings**

Concatenation: +.. 274	<b>lngh</b> ..... 316	<b>StEq</b> (..... 361	String entry: " .....363	<b>sub</b> (..... 363
<b>EqSt</b> ( ..... 290				

**Vectors**

<b>cnorm</b> ..... 273	<b>dim</b> ..... 281	<b>livc</b> ..... 316	<b>Sph</b> .....360	Vector entry: [ ] .... 369
<b>cross</b> ( ..... 277	<b>dim</b> ..... 281	<b>norm</b> ..... 323	<b>SphereV</b> .....360	
<b>Cyl</b> ..... 278	<b>dot</b> ( ..... 285	<b>RectV</b> ..... 344	<b>unitV</b> .....368	
<b>CylV</b> ..... 278	<b>Fill</b> ( ..... 295	<b>rnorm</b> ..... 346	<b>vc<li< b=""> .....369</li<></b>	

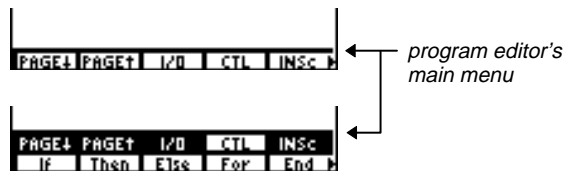
## Alphabetical Listing of Operations

All the operations in this section are included in the CATALOG. Non-alphabetic operations (such as  $+$ ,  $!$ , and  $>$ ) are listed at the end of the CATALOG. In this A to Z Reference, however, these operations are listed under their alphabetic equivalent (such as addition, factorial, and greater than).

You always can use the CATALOG to select an operation and paste it to the home screen or to a command line in the program editor. You also can use the specific keystrokes, menus, or screens listed in this section.

† Indicates menus or screens that paste the operation's name only if you are in the program editor. In most cases, you can use these menus or screens from the home screen to perform the operation interactively, without pasting the name.

‡ Indicates menus or screens that are valid only from the program editor's main menu. From the home screen, you cannot use these menus or screens to select an operation.



The syntax for some operations uses brackets [ ] to indicate optional arguments. If you use an optional argument, do not enter the brackets.

**abs**

MATH NUM menu  
 CPLX menu  
 MATRX CPLX menu  
 VECTR CPLX menu

**abs** *realNumber* or **abs** (*realExpression*)

Returns the absolute value of *realNumber* or *realExpression*.

abs -256.4  256.4abs -4\*3+13  25abs (-4\*3+13)  1**abs** (*complexNumber*)

Returns the magnitude (modulus) of *complexNumber*.

abs (3,4)  5abs (3∠4)  3**abs** (*real,imaginary*) returns  $\sqrt{(real^2+imaginary^2)}$ .**abs** (*magnitude∠angle*) returns *magnitude*.**abs** *list***abs** *matrix***abs** *vector*

Returns a list, matrix, or vector in which each element is the absolute value of the corresponding real or complex element in the argument.

abs {1.25,-5.67}  {1.25 5.67}abs [(3,4),(3∠4)]  [5 3]

## Addition: +

*numberA* + *numberB*

Returns the sum of two real or complex numbers.

In **RectC** complex number mode:(2,5)+(5,9)  (7,14)*number* + *list*

Returns a list in which a real or complex *number* is added to each element of a real or complex *list*.

4+{1,2,3}  {5 6 7}3+{1,7,(2,1)}  {(4,0) (10,0) (5,1)}

*listA* + *listB*

*matrixA* + *matrixB*

*vectorA* + *vectorB*

Returns a list, matrix, or vector that is the sum of the corresponding real or complex elements in the arguments. The two arguments must have the same dimension.

For information about adding two strings, refer to **Concatenation** on page 274.

```
{1,2,3}+{4,5,6} [ENTER]      {5 7 9}
[[1,2,3][4,5,6]]+[[4,5,6][7,8,9]]
[ENTER]                      [[5 7 9 ]
                               [11 13 15]]
[1,2,3]+[4,5,6] [ENTER]      [5 7 9]
```

## and

BASE BOOL menu

*integerA* and *integerB*

Compares two real integers bit by bit. Internally, both integers are converted to binary. When corresponding bits are compared, the result is 1 if both bits are 1; otherwise, the result is 0. The returned value is the sum of the bit results.

For example, 78 **and** 23 = 6.

$$\begin{array}{r} 78 = 1001110b \\ 23 = 0010111b \\ \hline 0000110b = 6 \end{array}$$

You can enter real numbers instead of integers, but they are truncated automatically before the comparison.

In **Dec** number base mode:

78 and 23 [ENTER] 6

In **Bin** number base mode:

1001110 and 10111 [ENTER] 110b  
Ans▶Dec [ENTER] 6d





## AsmComp(

CATALOG

**AsmComp**(*AsciiAssemblyPrgmName*,*HexAssemblyPrgmName*)

Compiles an assembly language program written in ASCII and stores the hex version. The compiled hex version, which uses about half the storage space of the ASCII version, cannot be edited.

When you execute the ASCII version, the TI-86 compiles it each time. To speed up execution, use **AsmComp**( to compile the ASCII version once and then execute the hex version each time you want to run the program.

## AsmPrgm

CATALOG

**AsmPrgm**

Must be used as the first line of an assembly language program.

## Assignment: =

[ALPHA] [=]

*equationVariable* = *expression*

Stores *expression* to *equationVariable*, without evaluating *expression*. (If you use [STO▶] to store an expression to a variable, the expression is evaluated and then the result is stored.)

$y1=2x^2+6x-5$  [ENTER] Done

The built-in equation variables used for graphing are case-sensitive. Use **y1**, not **Y1**.

## aug(

LIST OPS menu  
MATRX OPS menu

**aug**(*listA*,*listB*)

Returns a list consisting of *listB* appended (concatenated) to the end of *listA*. The lists can be real or complex.

$\text{aug}(\{1,-3,2\},\{5,4\})$  [ENTER]  
{1 -3 2 5 4}

**aug(matrixA,matrixB)**

Returns a matrix consisting of *matrixB* appended as new columns to the end of *matrixA*. The matrices can be real or complex. Both must have the same number of rows.

```
[[1,2,3][4,5,6]]>MATA ENTER
      [[1 2 3]
      [4 5 6]]
[[7,8][9,10]]>MATB ENTER
      [[7 8 ]
      [9 10]]
aug(MATA,MATB) ENTER
      [[1 2 3 7 8 ]
      [4 5 6 9 10]]
```

**aug(matrix,vector)**

Returns a matrix consisting of *vector* appended as a new column to the end of *matrix*. The arguments can be real or complex. The number of rows in *matrix* must equal the number of elements in *vector*.

**Axes(**

† GRAPH VARS menu

**Axes(xAxisVariable,yAxisVariable)**

Specifies the variables plotted for the axes in **DifEq** graphing mode. The *xAxisVariable* or *yAxisVariable* can be **t**, **Q1** through **Q9**, or **Q'1** through **Q'9**.

```
Axes(Q1,Q2) ENTER Done
```

**AxesOff**

† graph format screen

**AxesOff**

Turns off the graph axes.

**AxesOn**

† graph format screen

**AxesOn**

Turns on the graph axes.

**b**

BASE TYPE menu

*integerb*

Designates a real *integer* as binary, regardless of the number base mode setting.

In **Dec** number base mode:

```
10b ENTER 2
10b+10 ENTER 12
```

**Bin**

† mode screen

**Bin**

Sets binary number base mode. Results are displayed with the **b** suffix. In any number base mode, you can designate an appropriate value as binary, decimal, hexadecimal, or octal by using the **b**, **d**, **h**, or **o** designator, respectively, from the BASE TYPE menu.

In **Bin** number base mode:

10+**F**h+10o+10d **[ENTER]** 100011b

**►Bin**

BASE CONV menu

*number*►**Bin**

*list*►**Bin**

*matrix*►**Bin**

*vector*►**Bin**

Returns the binary equivalent of the real or complex argument.

In **Dec** number base mode:

2\*8 **[ENTER]** 16

Ans►**Bin** **[ENTER]** 10000b

{1,2,3,4}►**Bin** **[ENTER]** {1b 10b 11b 100b}

**Box**

† STAT DRAW menu

**Box** *xList*,*frequencyList*

Draws a box plot on the current graph, using the real data in *xList* and the frequencies in *frequencyList*.

Starting with a **ZStd** graph screen:

{1,2,3,4,5,9}►XL **[ENTER]** {1 2 3 4 5 9}

{1,1,1,4,1,1}►FL **[ENTER]** {1 1 1 4 1 1}

0►xMin:0►yMin **[ENTER]** 0

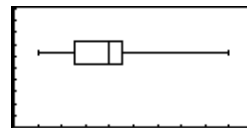
Box XL,FL **[ENTER]**

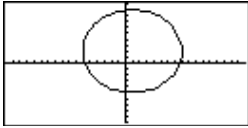
**Box** *xList*

Uses frequencies of 1.

**Box**

Uses the data in built-in variables **xStat** and **fStat**. These variables must contain valid data of the same dimension; otherwise, an error occurs.



<b>Circl</b> † GRAPH DRAW menu	<b>Circl</b> ( $x,y,radius$ ) Draws a circle with center ( $x,y$ ) and <i>radius</i> on the current graph.	Starting with a <b>ZStd</b> graph screen: ZSqr:Circl(1,2,7) <b>ENTER</b>
		
<b>CIDrw</b> † GRAPH DRAW menu † STAT DRAW menu	<b>CIDrw</b> Clears all drawn elements from the current graph.	
<b>CILCD</b> ‡ program editor I/O menu	<b>CILCD</b> Clears the home screen (LCD).	
<b>ClrEnt</b> MEM menu	<b>ClrEnt</b> Clears the contents of the Last Entry storage area.	
<b>CITbl</b> ‡ program editor I/O menu	<b>CITbl</b> Clears all values from the current table if <b>Indpnt: Ask</b> ( <b>IAsk</b> , page 304) is set.	
<b>cnorm</b> MATRX MATH menu	<b>cnorm</b> <i>matrix</i> Returns the column norm of a real or complex <i>matrix</i> . For each column, <b>cnorm</b> sums the absolute values (magnitudes of complex elements) of the elements in that column and returns the largest of those column sums.	$[[1,-2,3][4,5,-6]]\rightarrow\text{MAT}$ <b>ENTER</b> $\begin{bmatrix} 1 & -2 & 3 \\ 4 & 5 & -6 \end{bmatrix}$ cnorm MAT <b>ENTER</b> 9

**cnorm** *vector*

Returns the sum of the absolute values of the real or complex elements in *vector*.

```
[-1,2,-3]→VEC [ENTER]      [-1 2 -3]
cnorm VEC [ENTER]          6
```

**Concatenation: +**

⊕

**stringA + stringB**

Returns a string consisting of *stringB* appended (concatenated) to the end of *stringA*.

```
"your name:"→STR [ENTER]   your name:
"Enter "+STR [ENTER]       Enter your name:
```

**cond**

MATRX MATH menu

**cond** *squareMatrix*

Returns the condition number of a real or complex *squareMatrix*, which is calculated as:

$$\mathbf{cnorm} \mathit{squareMatrix} * \mathbf{cnorm} \mathit{squareMatrix}^{-1}$$

The condition number indicates how well-behaved *squareMatrix* is expected to be for certain matrix functions, particularly inverse. For a well-behaved matrix, the condition number is close to 1.

**log(cond *squareMatrix*)** indicates the number of digits that may be lost due to round-off errors in computing the inverse.

For a matrix with no inverse, **cond** returns an error.

```
[[[1,0,0][0,1,0][0,0,1]]→MAT1
[ENTER]                      [[1 0 0]
                              [0 1 0]
                              [0 0 1]]

cond MAT1 [ENTER]            1
log (Ans) [ENTER]           0

[[[1,2,3][4,5,6][7,8,9]]→MAT2
[ENTER]                      [[1 2 3]
                              [4 5 6]
                              [7 8 9]]

cond MAT2 [ENTER]            1.8E14
log (Ans) [ENTER]           14.2552725051
```

**conj**

CPLX menu  
 MATRX CPLX menu  
 VECTR CPLX menu

**conj** (*complexNumber*)

Returns the complex conjugate of *complexNumber*.

In **RectC** mode, **conj** (*real,imaginary*) returns (*real,-imaginary*).

In **PolarC** mode, **conj** (*magnitude∠angle*) returns (*magnitude∠-angle*),  $-\pi < \text{angle} \leq \pi$ .

**conj** *complexList***conj** *complexMatrix***conj** *complexVector*

Returns a complex list, matrix, or vector in which each element is the complex conjugate of the original.

In **RectC** complex number mode:

```
conj (3,4) [ENTER]      (3,-4)
conj (3∠2) [ENTER]      (-1.24844050964,-2.7...
```

In **PolarC** complex number mode:

```
conj (3∠2) [ENTER]      (3∠-2)
conj (3,4) [ENTER]      (5∠-.927295218002)
conj {√-2,(3,4)} [ENTER]
                        {(1.41421356237∠-1.5...
```

**CoordOff**

† graph format screen

**CoordOff**

Turns off cursor coordinates so they are not displayed at the bottom of a graph.

**CoordOn**

† graph format screen

**CoordOn**

Displays cursor coordinates at the bottom of a graph.

**COS**
 $\boxed{\text{COS}}$ 
**cos** *angle* or **cos** (*expression*)

Returns the cosine of *angle* or *expression*, which can be real or complex.

An angle is interpreted as degrees or radians according to the current angle mode. In any angle mode, you can designate an angle as degrees or radians by using the ° or  $\text{r}$  designator, respectively, from the MATH ANGLE menu.

**cos** *list*

Returns a list in which each element is the cosine of the corresponding element in *list*.

**cos** *squareMatrix*

Returns a square matrix that is the matrix cosine of *squareMatrix*. The matrix cosine corresponds to the result calculated using power series or Cayley-Hamilton Theorem techniques. This is *not* the same as simply calculating the cosine of each element.

 In **Radian** angle mode:

```
cos  $\pi/2$   $\boxed{\text{ENTER}}$           - .5
cos ( $\pi/2$ )  $\boxed{\text{ENTER}}$          0
cos 45°  $\boxed{\text{ENTER}}$            .707106781187
```

 In **Degree** angle mode:

```
cos 45  $\boxed{\text{ENTER}}$            .707106781187
cos ( $\pi/2$ ) $\text{r}$   $\boxed{\text{ENTER}}$        0
```

 In **Radian** angle mode:

```
cos {0, $\pi/2$ , $\pi$ }  $\boxed{\text{ENTER}}$     {1 0 -1}
```

 In **Degree** angle mode:

```
cos {0,60,90}  $\boxed{\text{ENTER}}$     {1 .5 0}
```

The *squareMatrix* cannot have repeated eigenvalues.

 **$\text{COS}^{-1}$** 
 $\boxed{2\text{nd}}$   $\boxed{\text{COS}^{-1}}$ 
 **$\text{cos}^{-1}$**  *number* or  **$\text{cos}^{-1}$**  (*expression*)

Returns the arccosine of *number* or *expression*, which can be real or complex.

 **$\text{cos}^{-1}$**  *list*

Returns a list in which each element is the arccosine of the corresponding element in *list*.

 In **Radian** angle mode:

```
 $\text{cos}^{-1}$  .5  $\boxed{\text{ENTER}}$           1.0471975512
```

 In **Degree** angle mode:

```
 $\text{cos}^{-1}$  1  $\boxed{\text{ENTER}}$           0
```

 In **Radian** angle mode:

```
 $\text{cos}^{-1}$  {0,.5}  $\boxed{\text{ENTER}}$     {1.57079632679,1.047...
```



<b>cosh</b> MATH HYP menu	<b>cosh</b> <i>number</i> or <b>cosh</b> ( <i>expression</i> )	cosh 1.2 <input type="button" value="ENTER"/>	1.81065556732
	Returns the hyperbolic cosine of <i>number</i> or <i>expression</i> , which can be real or complex.		
	<b>cosh</b> <i>list</i>	cosh {0,1.2} <input type="button" value="ENTER"/>	{1 1.81065556732}
	Returns a list in which each element is the hyperbolic cosine of the corresponding element in <i>list</i> .		
<b>cosh<sup>-1</sup></b> MATH HYP menu	<b>cosh<sup>-1</sup></b> <i>number</i> or <b>cos<sup>-1</sup></b> ( <i>expression</i> )	cosh <sup>-1</sup> 1 <input type="button" value="ENTER"/>	0
	Returns the inverse hyperbolic cosine of <i>number</i> or <i>expression</i> , which can be real or complex.		
	<b>cosh<sup>-1</sup></b> <i>list</i>	cosh <sup>-1</sup> {1,2.1,3} <input type="button" value="ENTER"/>	{0 1.37285914424 1.7...
	Returns a list in which each element is the inverse hyperbolic cosine of the corresponding element in <i>list</i> .		
<b>cross</b> VECTR MATH menu	<b>cross</b> ( <i>vectorA</i> , <i>vectorB</i> )	cross([1,2,3],[4,5,6]) <input type="button" value="ENTER"/>	[-3 6 -3]
	Returns the cross product of two real or complex vectors, where: <b>cross</b> ([ <i>a</i> , <i>b</i> , <i>c</i> ],[ <i>d</i> , <i>e</i> , <i>f</i> ]) = [ <i>bf-ce cd-af ae-bd</i> ]	cross([1,2],[3,4]) <input type="button" value="ENTER"/>	[0 0 -2]
	Both vectors must have the same dimension (either 2 or 3 elements). A 2-D vector is treated as a 3-D vector with 0 as the third element.		

<b>cSum(</b>	<b>cSum(list)</b>	<code>cSum({1,2,3,4})</code> <input type="button" value="ENTER"/>	{1 3 6 10}
LIST OPS menu	Returns a list of the cumulative sums of the real or complex elements in <i>list</i> , starting with the first element.	<code>{10,20,30}→L1</code> <input type="button" value="ENTER"/> <code>cSum(L1)</code> <input type="button" value="ENTER"/>	{10 20 30} {10 30 60}
<b>►Cyl</b>	<b>vector►Cyl</b>	<code>[-2,0]►Cyl</code> <input type="button" value="ENTER"/>	[2∠3.14159265359 0]
VECTR OPS menu	Displays a 2- or 3-element real <i>vector</i> result in cylindrical form, $[r∠\theta z]$ , even if the display mode is not set for cylindrical ( <b>CylV</b> ).	<code>[-2,0,1]►Cyl</code> <input type="button" value="ENTER"/>	[2∠3.14159265359 1]
<b>CylV</b>	<b>CylV</b>	In <b>CylV</b> vector coordinate mode and <b>Radian</b> angle mode: <code>[3,4,5]</code> <input type="button" value="ENTER"/>	[5∠.927295218002 5]
† mode screen	Sets cylindrical vector coordinate mode ( $[r∠\theta z]$ ).		
<b>d</b>	<b>numberd</b>	In <b>Bin</b> number base mode: <code>10d</code> <input type="button" value="ENTER"/> <code>10d+10</code> <input type="button" value="ENTER"/>	1010b 1100b
BASE TYPE menu	Designates a real <i>number</i> as decimal, regardless of the number base mode setting.		
<b>Dec</b>	<b>Dec</b>	In <b>Dec</b> number base mode: <code>10+10b+FH+10o</code> <input type="button" value="ENTER"/>	35
† mode screen	Sets decimal number base mode. In any number base mode, you can designate an appropriate value as binary, decimal, hexadecimal, or octal by using the <b>b</b> , <b>d</b> , <b>h</b> , or <b>o</b> designator, respectively, from the BASE TYPE menu.		

<p><b>►Dec</b> BASE CONV menu</p>	<p><i>number</i>►Dec <i>list</i>►Dec <i>matrix</i>►Dec <i>vector</i>►Dec</p>	<p>In <b>Hex</b> number base mode: 2*F <input type="button" value="ENTER"/> 1Eh Ans►Dec <input type="button" value="ENTER"/> 30d {A,B,C,D,E}►Dec <input type="button" value="ENTER"/> {10d 11d 12d 13d 14d}</p>
	<p>Returns the decimal equivalent of the real or complex argument.</p>	
<p><b>Degree</b> † mode screen</p>	<p><b>Degree</b> Sets degree angle mode.</p>	<p>In <b>Degree</b> angle mode: sin 90 <input type="button" value="ENTER"/> 1 sin (<math>\pi/2</math>) <input type="button" value="ENTER"/> .027412133592</p>
<p>Degree entry: ° MATH ANGLE menu</p>	<p><i>number</i>° or (<i>expression</i>)° Designates a real <i>number</i> or <i>expression</i> as degrees, regardless of the angle mode setting. <i>list</i>° Designates each element in <i>list</i> as degrees.</p>	<p>In <b>Radian</b> angle mode: cos 90 <input type="button" value="ENTER"/> -.448073616129 cos 90° <input type="button" value="ENTER"/> 0 cos {45,90,180}° <input type="button" value="ENTER"/> { .707106781187 0 -1 }</p>
<p><b>DeltaIst(</b> LIST OPS menu (DeltaI shows on menu)</p>	<p><b>DeltaIst(list)</b> Returns a list containing the differences between consecutive real or complex elements in <i>list</i>. This subtracts the first element in <i>list</i> from the second element, the second from the third, and so on. The resulting list is always one element shorter than <i>list</i>.</p>	<p>DeltaIst({20,30,45,70}) <input type="button" value="ENTER"/> {10 15 25}</p>

### DelVar(

‡ program editor  
CTL menu  
(DelVa shows  
on menu)

#### DelVar(*variable*)

Deletes the specified user-created *variable* from memory.

You cannot use **DelVar**( to delete a program variable or built-in variable.

2→A  2  
(A+2)<sup>2</sup>  16  
DelVar(A)  Done  
(A+2)<sup>2</sup>  ERROR 14 UNDEFINED

### der1(

CALC menu

#### der1(*expression,variable,value*)

Returns the first derivative of *expression* with respect to *variable* at the real or complex *value*.

der1(x^3,x,5)  75

#### der1(*expression,variable*)

Uses the current value of *variable*.

3→x  3  
der1(x^3,x)  27

#### der1(*expression,variable,list*)

Returns a list containing the first derivatives at the values specified by the elements in *list*.

der1(x^3,x,{5,3})  {75 27}

### der2(

CALC menu

#### der2(*expression,variable,value*)

Returns the second derivative of *expression* with respect to *variable* at the real or complex *value*.

der2(x^3,x,5)  30

#### der2(*expression,variable*)

Uses the current value of *variable*.

3→x  3  
der2(x^3,x)  18

#### der2(*expression,variable,list*)

Returns a list containing the second derivatives at the values specified by the elements in *list*.

der2(x^3,x,{5,3})  {30 18}

<b>det</b> MATRX MATH menu	<b>det</b> <i>squareMatrix</i> Returns the determinant of <i>squareMatrix</i> . The result is real for a real matrix, complex for a complex matrix.	[[1,2][3,4]]>MAT <b>ENTER</b> det MAT <b>ENTER</b>	[[1 2] [3 4]] -2
<b>DifEq</b> † mode screen	<b>DifEq</b> Sets differential equation graphing mode.		
<b>dim</b> MATRX OPS menu VECTR OPS menu	<b>dim</b> <i>matrix</i> Returns a list containing the dimensions (number of rows and columns) of a real or complex <i>matrix</i> .	[[2,7,1][-8,0,1]]>MAT <b>ENTER</b> dim MAT <b>ENTER</b>	[[2 7 1] [-8 0 1]] {2 3}
	<b>dim</b> <i>vector</i> Returns the length (number of elements) of a real or complex <i>vector</i> .	dim [-8,0,1] <b>ENTER</b>	3
<b>→dim</b> <b>STO▶</b> , then MATRX OPS menu <b>STO▶</b> , then VECTR OPS menu	<b>{rows,columns}&gt;dim</b> <i>matrixName</i> If <i>matrixName</i> does not exist, creates a new matrix with the specified dimensions and fills it with zeros. If <i>matrixName</i> exists, redimensions that matrix to the specified dimensions. Existing elements within the new dimensions are not changed; elements outside the new dimensions are deleted. If additional elements are created, they are filled with zeros.	[[2,7][-8,0]]>MAT <b>ENTER</b> {3,3}>dim MAT <b>ENTER</b> MAT <b>ENTER</b>	[[2 7] [-8 0]] {3 3} [[2 7 0] [-8 0 0] [0 0 0]]

*#ofElements*→**dim** *vectorName*

If *vectorName* does not exist, creates a new vector with the specified *#ofElements* and fills it with zeros.

If *vectorName* exists, redimensions that vector to the specified *#ofElements*. Existing elements within the new dimension are not changed; elements outside the new dimension are deleted. If additional elements are created, they are filled with zeros.

```
DelVar(VEC) [ENTER] Done
4→dim VEC [ENTER] 4
VEC [ENTER] [0 0 0 0]
[1,2,3,4]→VEC [ENTER] [1 2 3 4]
2→dim VEC [ENTER] 2
VEC [ENTER] [1 2]
3→dim VEC [ENTER] 3
VEC [ENTER] [1 2 0]
```

## dimL

LIST OPS menu

**dimL** *list*

Returns the length (number of elements) of a real or complex *list*.

```
dimL {2,7,-8,0} [ENTER] 4
1/dimL {2,7,-8,0} [ENTER] .25
```

## →dimL

[STO→], then LIST OPS menu

*#ofElements*→**dimL** *listName*

If *listName* does not exist, creates a new list with the specified *#ofElements* and fills it with zeros.

If *listName* exists, redimensions that list to the specified *#ofElements*. Existing elements within the new dimension are not changed; elements outside the new dimension are deleted. If additional elements are created, they are filled with zeros.

```
3→dimL NEWLIST [ENTER] 3
NEWLIST [ENTER] {0 0 0}
{2,7,-8,1}→L1 [ENTER] {2 7 -8 1}
5→dimL L1 [ENTER] 5
L1 [ENTER] {2 7 -8 1 0}
2→dimL L1 [ENTER] 2
L1 [ENTER] {2 7}
```

## DirFld

† graph format screen (scroll down to second screen)

**DirFld**

In **DifEq** graphing mode, turns on direction fields. To turn off direction and slope fields, use **FldOff**.

### Disp

‡ program editor  
I/O menu

**Disp** *valueA,valueB,valueC, ...*

Displays each value. The values can include strings and variable names.

```

10→x [ENTER] 10
Disp x^3+3 x-6 [ENTER] 1024
Done
"Hello"→STR [ENTER] Hello
Disp STR+", Jan" [ENTER] Hello, Jan
Done
    
```

### Disp

Displays the home screen.

### DispG

† GRAPH menu  
‡ program editor  
I/O menu

### DispG

Displays the current graph.

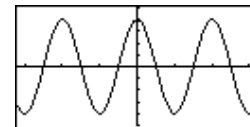
*Function names are case-sensitive. Use **y1**, not **Y1**.*

*To select from a list of window variable names, press [2nd] [CATLG-VARS] [MORE] [MORE] [F3].*

Program segment in **Func** graphing mode:

```

:
:y1=4cos x
:-10→xMin:10→xMax
:-5→yMin:5→yMax
:DispG
:
    
```



### DispT

‡ program editor  
I/O menu

### DispT

Displays the table.

Function names are case-sensitive. Use *y1*, not *Y1*.

Program segment in **Func** graphing mode:

```

:
:
:y1=4cos x
:DispT
:
:

```

X	Y1
0	4
1	2.46209
2	-1.66459
3	-3.95997
4	-2.61457
5	1.13469

### Division: /

⊞

*numberA*/*numberB* or *(expressionA)*/*(expressionB)*

Returns one argument divided by another. The arguments can be real or complex.

```

-98/4 [ENTER] -24.5
-98/(4*3) [ENTER] -8.16666666667

```

*number*/*list* or *(expression)*/*list*

Returns a list in which each element is *number* or *expression* divided by the corresponding element in *list*.

```

100/{10,25,2} [ENTER] {10 4 50}

```

*list*/*number* or *list*/*(expression)*

*vector*/*number* or *vector*/*(expression)*

Returns a list or vector in which each element of *list* or *vector* is divided by *number* or *expression*.

```

{120,92,8}/4 [ENTER] {30 23 2}

```

In **RectC** complex number mode:

```

[8,1,(5,2)]/2 [ENTER] [(4,0) (.5,0) (2.5,1...

```

*listA*/*listB*

Returns a list in which each element of *listA* is divided by the corresponding element of *listB*. The lists must have the same dimension.

```

{1,2,3}/{4,5,6} [ENTER] {.25 .4 .5}

```



**DMS entry: ' °**

MATH ANGLE menu

In a trig calculation, the result of a DMS entry is treated as degrees in the **Degree** angle mode only. It is treated as radians in **Radian** angle mode.

*degrees'minutes'seconds'*

Designates the entered angle is in DMS format. *degrees* ( $\leq 999,999$ ), *minutes* ( $< 60$ ), and *seconds* ( $< 60$ , may have decimal places) must be entered as real numbers, not as variable names or expressions.

Do not use ° and " symbols to specify *degrees* and *seconds*. For example,  $5^{\circ}59'$  is interpreted as implied multiplication of  $5^{\circ} * 59'$  according to the current angle mode setting.

54'32'30'  54.5416666667In **Degree** angle mode:cos 54'32'30'  .580110760699In **Radian** angle mode:cos 54'32'30'  -.422502666138

**Do not** use the following notation; in **Degree** angle mode:

5°59'  295**▶DMS**

MATH ANGLE menu

*angle▶DMS*

Displays *angle* in DMS format. The result is shown in *degrees°minutes'seconds"* format, even though you use *degrees'minutes'seconds'* to enter a DMS angle.

In **Degree** angle mode:45.371▶DMS  45°22'15.6"54'32'30'\*2  109.083333333Ans▶DMS  109°5'0"**dot(**

VECTR MATH menu

**dot**(*vectorA,vectorB*)

Returns the dot product of two real or complex vectors.

**dot**([*a,b,c*],[*d,e,f*]) returns ***a\*d+b\*e+c\*f***.dot([1,2,3],[4,5,6])  32**DrawDot**

† graph format screen

**DrawDot**

Sets dot graphing format.

### DrawF

GRAPH DRAW menu

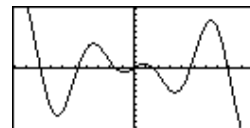
---

### DrawF *expression*

Draws *expression* (in terms of **x**) on the current graph.

In **Func** graphing mode:

ZStd:DrawF 1.25 x cos x



---

### DrawLine

† graph format screen

### DrawLine

Sets connected line graphing format.

---

**DrEqu(**

† GRAPH menu

To enter the ' character for the Q' variables, use the CHAR MISC menu.

**DrEqu**(*xAxisVariable*,*yAxisVariable*,*xList*,*yList*,*tList*)

In **DifEq** graphing mode, draws the solution to a set of differential equations stored in the **Q'** variables specified by *xAxisVariable* and *yAxisVariable*. If direction fields are off (**FldOff** is selected), the initial values must be stored also.

After the solution is drawn, **DrEqu(** waits for you to move the cursor to a new initial value and press **ENTER** to draw the new solution.

You then are prompted to press **Y** (to specify another initial value) or **N** (to stop).

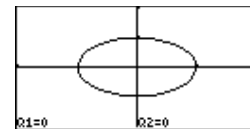
For the last-drawn solution, the **x**, **y**, and **t** values (beginning at their initial values) are stored to *xList*, *yList*, and *tList*, respectively.

**DrEqu**(*xAxisVariable*,*yAxisVariable*)

Does not store **x**, **y**, and **t** values for the solution.

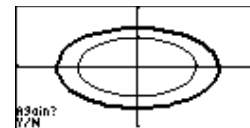
In **DifEq** graphing mode, starting with a **ZStd** graph screen:

```
Q'1=Q2:Q'2=-Q1 [ENTER] Done
0→tMin:1→QI1:0→QI2 [ENTER] 0
DrEqu(Q1,Q2,XL,YL,TL) [ENTER]
```



Move the cursor to a new initial value.

**ENTER**



Press **N** to stop graphing. You can then examine **XL**, **YL**, and **TL**.

**DrInv**

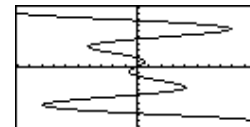
GRAPH DRAW menu

**DrInv** *expression*

Draws the inverse of *expression* by plotting **x** values on the y-axis and **y** values on the x-axis.

In **Func** graphing mode:

```
ZStd:DrInv 1.25 x cos x [ENTER]
```



### DS<(

‡ program editor  
CTL menu

**:DS<(variable,value)**  
**:command-if-variable≥value**  
**:commands**

Decrements *variable* by 1. If the result is  $< value$ , skips *command-if-variable≥value*.

If the result is  $\geq value$ , then *command-if-variable≥value* is executed.

*variable* cannot be a built-in variable.

Program segment:

```

:
:9>A
:Lb1 Start
:Disp A
:DS<(A,5)
:Goto Start
:Disp "A is now <5"
:

```

### dxDer1

† mode screen

#### dxDer1

Sets **der1** as the current differentiation type. **der1** differentiates exactly and calculates the value for each function in an expression. It is more accurate than **dxNDer**, but more restrictive in that only certain functions are valid in the expression.

The current differentiation type is used by the **arc**( and **TanLn**( functions, as well as interactive graphing operations  $dy/dx$ ,  $dr/d\theta$ ,  $dy/dt$ ,  $dx/dt$ , ARC, TanLn, and INFLC.

### dxNDer

† mode screen

#### dxNDer

Sets **nDer** as the current differentiation type. **nDer** differentiates numerically and calculates the value for an expression. It is less accurate than **dxDer1**, but less restrictive in the functions that are valid in the expression.

The current differentiation type is used by the **arc**( and **TanLn**( functions, as well as interactive graphing operations  $dy/dx$ ,  $dr/d\theta$ ,  $dy/dt$ ,  $dx/dt$ , ARC, TanLn, and INFLC.

### e^

2nd [e^x]

**e^power** or **e^(expression)**

Returns **e** raised to *power* or *expression*. The argument can be real or complex.

e^0 ENTER

1

**e<sup>list</sup>**

Returns a list in which each element is **e** raised to the power specified by the corresponding element in *list*.

```
e^{1,0,.5} ENTER
{2.71828182846 1 1.6...
```

**e<sup>squareMatrix</sup>**

Returns a square matrix that is the matrix exponential of *squareMatrix*. The matrix exponential corresponds to the result calculated using power series or Cayley-Hamilton Theorem techniques. This is *not* the same as simply calculating the exponential of each element.

*The squareMatrix cannot have repeated eigenvalues.*

**eigVc**

MATRIX MATH menu

*The squareMatrix cannot have repeated eigenvalues.*

**eigVc squareMatrix**

Returns a matrix containing the eigenvectors for a real or complex *squareMatrix*, where each column in the result corresponds to an eigenvalue. The eigenvectors of a real matrix may be complex. Note that an eigenvector is not unique; it may be scaled by any constant factor. TI-86 eigenvectors are normalized.

```
In RectC complex number mode:
[[-1,2,5][3,-6,9][2,-5,7]]>MAT
ENTER
[[-1 2 5]
 [3 -6 9]
 [2 -5 7]]
eigVc MAT ENTER
[[(.800906446592,0) ...
 [(-.484028886343,0)...
 [(-.352512270699,0)...
```

**eigVl**

MATRIX MATH menu

**eigVl squareMatrix**

Returns a list of the eigenvalues of a real or complex *squareMatrix*. The eigenvalues of a real matrix may be complex.

```
In RectC complex number mode:
[[-1,2,5][3,-6,9][2,-5,7]]>MAT
ENTER
[[-1 2 5]
 [3 -6 9]
 [2 -5 7]]
eigVl MAT ENTER
{(-4.40941084667,0) ...
```

<p><b>Else</b></p> <p>‡ program editor CTL menu</p>	<p>Refer to syntax information for <b>If</b>, beginning on page 305. See the <b>If:Then:Else:End</b> syntax.</p>		
<p><b>End</b></p> <p>‡ program editor CTL menu</p>	<p><b>End</b></p> <p>Identifies the end of a <b>While</b>, <b>For</b>, <b>Repeat</b>, or <b>If-Then-Else</b> loop.</p>		
<p><b>Eng</b></p> <p>† mode screen</p>	<table border="0"> <tr> <td data-bbox="344 347 1028 484"> <p><b>Eng</b></p> <p>Sets engineering notation mode, in which the power-of-10 exponent is a multiple of 3.</p> </td> <td data-bbox="1028 347 1466 484"> <p>In <b>Eng</b> notation mode:</p> <p>123456789 <input type="button" value="ENTER"/> 123.456789E6</p> <p>In <b>Normal</b> notation mode:</p> <p>123456789 <input type="button" value="ENTER"/> 123456789</p> </td> </tr> </table>	<p><b>Eng</b></p> <p>Sets engineering notation mode, in which the power-of-10 exponent is a multiple of 3.</p>	<p>In <b>Eng</b> notation mode:</p> <p>123456789 <input type="button" value="ENTER"/> 123.456789E6</p> <p>In <b>Normal</b> notation mode:</p> <p>123456789 <input type="button" value="ENTER"/> 123456789</p>
<p><b>Eng</b></p> <p>Sets engineering notation mode, in which the power-of-10 exponent is a multiple of 3.</p>	<p>In <b>Eng</b> notation mode:</p> <p>123456789 <input type="button" value="ENTER"/> 123.456789E6</p> <p>In <b>Normal</b> notation mode:</p> <p>123456789 <input type="button" value="ENTER"/> 123456789</p>		
<p><b>Eq&gt;St(</b></p> <p>STRNG menu</p>	<table border="0"> <tr> <td data-bbox="344 484 1028 723"> <p><b>Eq&gt;St</b>(<i>equationVariable</i>,<i>stringVariable</i>)</p> <p>Converts the contents of <i>equationVariable</i> to a string and stores it to <i>stringVariable</i>. Be sure to specify an equation variable, not an equation.</p> <p>To create an equation variable, use an equal sign (=) to define the variable. For example, enter <b>A=B*C</b>, not <b>B*C&gt;A</b>.</p> </td> <td data-bbox="1028 484 1466 723"> <p>A=B*C <input type="button" value="ENTER"/> Done</p> <p>5&gt;B <input type="button" value="ENTER"/> 5</p> <p>2&gt;C <input type="button" value="ENTER"/> 2</p> <p>A <input type="button" value="ENTER"/> 10</p> <p>Eq&gt;St(A,STR) <input type="button" value="ENTER"/> Done</p> <p>STR <input type="button" value="ENTER"/> B*C</p> </td> </tr> </table>	<p><b>Eq&gt;St</b>(<i>equationVariable</i>,<i>stringVariable</i>)</p> <p>Converts the contents of <i>equationVariable</i> to a string and stores it to <i>stringVariable</i>. Be sure to specify an equation variable, not an equation.</p> <p>To create an equation variable, use an equal sign (=) to define the variable. For example, enter <b>A=B*C</b>, not <b>B*C&gt;A</b>.</p>	<p>A=B*C <input type="button" value="ENTER"/> Done</p> <p>5&gt;B <input type="button" value="ENTER"/> 5</p> <p>2&gt;C <input type="button" value="ENTER"/> 2</p> <p>A <input type="button" value="ENTER"/> 10</p> <p>Eq&gt;St(A,STR) <input type="button" value="ENTER"/> Done</p> <p>STR <input type="button" value="ENTER"/> B*C</p>
<p><b>Eq&gt;St</b>(<i>equationVariable</i>,<i>stringVariable</i>)</p> <p>Converts the contents of <i>equationVariable</i> to a string and stores it to <i>stringVariable</i>. Be sure to specify an equation variable, not an equation.</p> <p>To create an equation variable, use an equal sign (=) to define the variable. For example, enter <b>A=B*C</b>, not <b>B*C&gt;A</b>.</p>	<p>A=B*C <input type="button" value="ENTER"/> Done</p> <p>5&gt;B <input type="button" value="ENTER"/> 5</p> <p>2&gt;C <input type="button" value="ENTER"/> 2</p> <p>A <input type="button" value="ENTER"/> 10</p> <p>Eq&gt;St(A,STR) <input type="button" value="ENTER"/> Done</p> <p>STR <input type="button" value="ENTER"/> B*C</p>		
<p><b>Equal: =</b></p> <p><input type="button" value="ALPHA"/> [=]</p>	<p>Refer to syntax information for <b>Assignment</b> on page 270.</p> <p>If you use = in an expression in which the first argument is not a variable name at the beginning of a line, the = is treated as -(.</p> <table border="0"> <tr> <td data-bbox="344 723 1028 881"> <p>Example of = treated as -(, where 4=6+1 is evaluated as 4-(6+1):</p> <p>4=6+1 <input type="button" value="ENTER"/> -3</p> <p>For true/false comparison, use == instead:</p> <p>4==6+1 <input type="button" value="ENTER"/> 0</p> </td> <td data-bbox="1028 723 1466 881"></td> </tr> </table>	<p>Example of = treated as -(, where 4=6+1 is evaluated as 4-(6+1):</p> <p>4=6+1 <input type="button" value="ENTER"/> -3</p> <p>For true/false comparison, use == instead:</p> <p>4==6+1 <input type="button" value="ENTER"/> 0</p>	
<p>Example of = treated as -(, where 4=6+1 is evaluated as 4-(6+1):</p> <p>4=6+1 <input type="button" value="ENTER"/> -3</p> <p>For true/false comparison, use == instead:</p> <p>4==6+1 <input type="button" value="ENTER"/> 0</p>			

**Equal to: ==**

TEST menu

The == operator is used to compare arguments, while = is used to assign a value or expression to a variable.

 $numberA == numberB$  $matrixA == matrixB$  $vectorA == vectorB$  $stringA == stringB$ 

Tests whether the condition  $argumentA == argumentB$  is true or false. Numbers, matrices, and vectors can be real or complex. If complex, the magnitude (modulus) of each element is compared. Strings are case-sensitive.

- If true ( $argumentA = argumentB$ ), returns **1**.
- If false ( $argumentA \neq argumentB$ ), returns **0**.

 $listA == listB$ 

Returns a list of **1s** and/or **0s** to indicate if each element in  $listA$  is = the corresponding element in  $listB$ .

 $2+2==2+2$   1 $2+(2==2)+2$   5 $[1,2]==[3-2,-1+3]$   1 $"A"=="a"$   0 $\{1,5,9\}==\{1,-6,9\}$   {1 0 1}**Euler**

† graph format screen  
(scroll down to  
second screen)

**Euler**

In **DifEq** graphing mode, uses an algorithm based on the Euler method to solve differential equations. Typically, **Euler** is less accurate than **RK** but finds the solutions much quicker.

**eval**

MATH MISC menu

**eval**  $xValue$ 

Returns a list containing the **y** values of all defined and selected functions evaluated at a real  $xValue$ .

Remember that built-in equation variables **y1** and **y2** are case-sensitive:

 $y1=x^3+x+5$   Done $y2=2 \times$   Done $eval\ 5$   {135 10}

<p><b>evalF(</b> CALC menu</p>	<p><b>evalF</b>(<i>expression,variable,value</i>) Returns the value of <i>expression</i> evaluated with respect to <i>variable</i> at a real or complex <i>value</i>.</p>	<p>evalF(x^3+x+5,x,5) <input type="button" value="ENTER"/> 135</p>
	<p><b>evalF</b>(<i>expression,variable,list</i>) Returns a list containing the values of <i>expression</i> evaluated with respect to <i>variable</i> at each element in <i>list</i>.</p>	<p>evalF(x^3+x+5,x,{3,5}) <input type="button" value="ENTER"/> {35 135}</p>
<p>Exponent: <b>E</b></p>	<p><i>number</i> <b>E</b><i>power</i> or (<i>expression</i>)<b>A</b> <b>E</b>(<i>expression</i>)<b>B</b></p>	<p>12.3456789E5 <input type="button" value="ENTER"/> 1234567.89 (1.78/2.34)E2 <input type="button" value="ENTER"/></p>
<p><input type="button" value="EE"/></p>	<p>Returns a real or complex <i>number</i> raised to the <i>power</i> of 10, where <i>power</i> is a real integer such that <math>-999 &lt; power &lt; 999</math>. Any <i>expressions</i> must evaluate to appropriate values.</p> <p><i>list</i> <b>E</b><i>power</i> or <i>list</i> <b>E</b>(<i>expression</i>) Returns a list in which each element is the corresponding element in <i>list</i> raised to the <i>power</i> of 10.</p>	<p>76.0683760684 {6.34,854.6}E3 <input type="button" value="ENTER"/> {6340 854600}</p>

---



**ExpR**

STAT CALC menu

Built-in equation variables such as **y1**, **r1**, and **xt1** are case-sensitive. Do not use **Y1**, **R1**, and **XT1**.

**ExpR** *xList,yList,frequencyList,equationVariable*

Fits an exponential regression model ( $y=ab^x$ ) to real data pairs in *xList* and *yList* (**y** values must be  $> 0$ ) and frequencies in *frequencyList*. The regression equation is stored to *equationVariable*, which must be a built-in equation variable such as **y1**, **r1**, and **xt1**.

Values used for *xList*, *yList*, and *frequencyList* are stored automatically to built-in variables **xStat**, **yStat**, and **fStat**, respectively. The regression equation is stored also to built-in equation variable **RegEq**.

**ExpR** *xList,yList,equationVariable*

Uses frequencies of 1.

**ExpR** *xList,yList,frequencyList*

Stores the regression equation to **RegEq** only.

**ExpR** *xList,yList*

Uses frequencies of 1, and stores the regression equation to **RegEq** only.

**ExpR** *equationVariable*

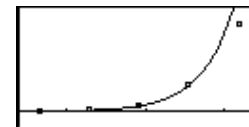
Uses **xStat**, **yStat**, and **fStat** for *xList*, *yList*, and *frequencyList*, respectively. These built-in variables must contain valid data of the same dimension; otherwise, an error occurs. The regression equation is stored to *equationVariable* and **RegEq**.

In **Func** graphing mode:

```
{1,2,3,4,5}→L1 [ENTER]
{1,2,3,4,5}
{1,20,55,230,742}→L2 [ENTER]
{1 20 55 230 742}
ExpR L1,L2,y1 [ENTER]
```

```
ExpReg
y=a*b^x
a=.411389488
b=4.78796057
corr=.97681282
n=5
```

```
Plot1(1,L1,L2) [ENTER]
ZData [ENTER] Done
```



**ExpR**

Uses **xStat**, **yStat**, and **fStat**, and stores the regression equation to **RegEq** only.

**Factorial: !**

MATH PROB menu

*number!* or (*expression*)!

6!

720

Returns the factorial of a real integer or non-integer, where  $0 \leq \text{integer} \leq 449$  and  $0 \leq \text{non-integer} \leq 449.9$ . For a non-integer, the Gamma function is used to find the factorial. An *expression* must evaluate to an appropriate value.

12.5!

1710542068.32

*list!*

{6,7,8}!

{720 5040 40320}

Returns a list in which each element is the factorial of the corresponding element in *list*.

**fcstx**

† STAT menu

**fcstx** *yValue*

Based on the current regression equation (**ReqEq**), returns the forecasted **x** at a real *yValue*.

**fcsty**

† STAT menu

**fcsty** *xValue*

Based on the current regression equation (**ReqEq**), returns the forecasted **y** at a real *xValue*.

---

**Fill(**

LIST OPS menu  
 MATRX OPS menu  
 VECTR OPS menu

**Fill**(*number,listName*)**Fill**(*number,matrixName*)**Fill**(*number,vectorName*)

Replaces each element in an existing *listName*,  
*matrixName*, or *vectorName* with a real or complex  
*number*.

{3,4,5}→L1  {3 4 5}  
 Fill(8,L1)  Done  
 L1  {8 8 8}  
 Fill((3,4),L1)  Done  
 L1  {(3,4) (3,4) (3,4)}

**Fix**

† mode screen

**Fix** *integer* or **Fix** (*expression*)

Sets fixed decimal mode for *integer* number of decimal  
 places, where  $0 \leq \textit{integer} \leq 11$ . An *expression* must  
 evaluate to an appropriate integer.

Fix 3  Done  
 $\pi/2$   1.571  
 Float  Done  
 $\pi/2$   1.57079632679

**FldOff**

† graph format screen  
 (scroll down to  
 second screen)

**FldOff**

In **DifEq** graphing mode, turns off the slope and  
 direction fields. To turn on slope fields, use **SlpFld**. To  
 turn on direction fields, use **DirFld**.

**Float**

† mode screen

**Float**

Sets floating decimal mode.

In **Radian** angle mode:

Fix 11  Done  
 $\sin(\pi/6)$   .5000000000  
 Float  Done  
 $\sin(\pi/6)$   .5

<b>fMax()</b> CALC menu	<b>fMax</b> ( <i>expression,variable,lower,upper</i> ) Returns the value at which a local maximum of <i>expression</i> with respect to <i>variable</i> occurs, between real <i>lower</i> and <i>upper</i> values for <i>variable</i> . The tolerance is controlled by the built-in variable <b>tol</b> , whose default is 1E-5. To view or set <b>tol</b> , press <b>[2nd]</b> <b>[MEM]</b> <b>[F4]</b> to display the tolerance editor.	$fMax(\sin x, x, -\pi, \pi)$ <b>[ENTER]</b> 1.57079632598
<b>fMin()</b> CALC menu	<b>fMin</b> ( <i>expression,variable,lower,upper</i> ) Returns the value at which a local minimum of <i>expression</i> with respect to <i>variable</i> occurs, between real <i>lower</i> and <i>upper</i> bounds for <i>variable</i> . The tolerance is controlled by the built-in variable <b>tol</b> , whose default is 1E-5. To view or set <b>tol</b> , press <b>[2nd]</b> <b>[MEM]</b> <b>[F4]</b> to display the tolerance editor.	$fMin(\sin x, x, -\pi, \pi)$ <b>[ENTER]</b> -1.57079632691
<b>fnInt()</b> CALC menu	<b>fnInt</b> ( <i>expression,variable,lower,upper</i> ) Returns the numerical function integral of <i>expression</i> with respect to <i>variable</i> , between real <i>lower</i> and <i>upper</i> bounds for <i>variable</i> . The tolerance is controlled by the built-in variable <b>tol</b> , whose default is 1E-5. To view or set <b>tol</b> , press <b>[2nd]</b> <b>[MEM]</b> <b>[F4]</b> to display the tolerance editor.	$fnInt(x^2, x, 0, 1)$ <b>[ENTER]</b> .333333333333
<b>FnOff</b> † GRAPH VARS menu	<b>FnOff</b> <i>function#,function#, ...</i> Deselects the specified equation function numbers.	$FnOff$ 1,3 <b>[ENTER]</b> Done

<b>FnOff</b>	Deselects all equation function numbers.	FnOff <input type="button" value="ENTER"/>	Done
--------------	--	--	------

**FnOn**

† GRAPH VARS menu

<b>FnOn</b>	<i>function#</i> , <i>function#</i> , ...	FnOn 1,3 <input type="button" value="ENTER"/>	Done
-------------	---	---	------

Selects the specified equation function numbers, in addition to any others already selected.

<b>FnOn</b>	Selects all equation function numbers.	FnOn <input type="button" value="ENTER"/>	Done
-------------	--	---	------

**For(**‡ program editor  
CTL menu

<b>:For</b> ( <i>variable</i> , <i>begin</i> , <i>end</i> , <i>step</i> ) or <b>:For</b> ( <i>variable</i> , <i>begin</i> , <i>end</i> )	<b>:loop</b>	Program segment:
<b>:End</b>	<b>:loop</b>	:
<b>:commands</b>	<b>:End</b>	For(A,0,8,2)
Executes the commands in <i>loop</i> iteratively, where the number of repetitions is controlled by <i>variable</i> . The first time through the loop, <i>variable</i> = <i>begin</i> . At the <b>End</b> of the loop, <i>variable</i> is incremented by <i>step</i> . The loop is repeated until <i>variable</i> > <i>end</i> . If you do not specify <i>step</i> , the default is 1.	<b>:commands</b>	Disp A <sup>2</sup>
You can specify values such that <i>begin</i> > <i>end</i> . If so, be sure to specify a negative <i>step</i> .		End
		:
		Displays 0, 4, 16, 36, and 64.
		:
		For(A,0,8)
		Disp A <sup>2</sup>
		End
		:
		Displays 0, 1, 4, 9, 16, 25, 36, 49, and 64.

<b>Form(</b> LIST OPS menu	<b>Form("formula",listName)</b> Generates the contents of <i>listName</i> automatically, based on the attached <i>formula</i> . If you express <i>formula</i> in terms of a list, you can generate one list based on the contents of another.  The contents of <i>listName</i> are updated automatically if you edit <i>formula</i> or edit a list referenced in <i>formula</i> .	$\{1,2,3,4\} \rightarrow L1$ [ENTER]      {1 2 3 4} Form("10*L1",L2) [ENTER]      Done L2 [ENTER]      {10 20 30 40}  $\{5,10,15,20\} \rightarrow L1$ [ENTER] L2 [ENTER]      {5 10 15 20} {50 100 150 200}  Form("L1/5",L2) [ENTER]      Done L2 [ENTER]      {1 2 3 4}
<b>fPart</b> MATH NUM menu	<b>fPart number</b> or <b>fPart (expression)</b> Returns the fractional part of a real or complex <i>number</i> or <i>expression</i> .  <b>fPart list</b> <b>fPart matrix</b> <b>fPart vector</b>  Returns a list, matrix, or vector in which each element is the fractional part of the corresponding element in the specified argument.	fPart 23.45 [ENTER]      .45 fPart (-17.26*8) [ENTER]      -.08  $[[1,-23.45][-99.5,47.15]] \rightarrow \text{MAT}$ [ENTER]      [[1      -23.45] [-99.5 47.15 ]]  fPart MAT [ENTER]      [[0      -.45] [-.5 .15 ]]
<b>Frac</b> MATH MISC menu	$number \rightarrow \text{Frac}$ Displays a real or complex <i>number</i> as its rational equivalent, a fraction reduced to its simplest terms.  If <i>number</i> cannot be simplified or if the denominator is more than four digits, the decimal equivalent is returned.	$1/3+2/7$ [ENTER]      .619047619048 Ans $\rightarrow \text{Frac}$ [ENTER]      13/21

<i>list</i> ►Frac	{1/2+1/3,1/6-3/8}►L1	ENTER	
<i>matrix</i> ►Frac			{.833333333333 -.208...
<i>vector</i> ►Frac	Ans►Frac	ENTER	{5/6 -5/24}

Returns a list, matrix, or vector in which each element is the rational equivalent of the corresponding element in the argument.

**Func**

† mode screen

**Func**

Sets function graphing mode.

**gcd(**

MATH MISC menu

<b>gcd</b> ( <i>integerA</i> , <i>integerB</i> )	gcd(18,33)	ENTER	3
	Returns the greatest common divisor of two nonnegative integers.		
<b>gcd</b> ( <i>listA</i> , <i>listB</i> )	gcd({12,14,16},{9,7,5})	ENTER	{3 7 1}
	Returns a list in which each element is the gcd of the two corresponding elements in <i>listA</i> and <i>listB</i> .		

**Get(**‡ program editor  
I/O menu**Get**(*variable*)Gets data from a CBL or CBR System or another TI-86 and stores it to *variable*.

### getKey

‡ program editor  
I/O menu

#### getKey

Returns the key code for the last key pressed. If no key has been pressed, **getKey** returns **0**. Refer to the TI-86 key code diagram in Chapter 16.

Program:

```
PROGRAM: CODES
: Lbl TOP
: getKey→KEY
: While KEY==0
:   getKey→KEY
: End
: Disp KEY
: Goto TOP
```

To break the program, press **[ON]** and then **[F5]**.

### Goto

‡ program editor  
CTL menu

#### Goto label

Transfers (branches) program control to the *label* specified by an existing **Lbl** instruction.

Program segment:

```
:
: 0→TEMP:1→J
: Lbl TOP
: TEMP+J→TEMP
: If J<10
: Then
:   J+1→J
: Goto TOP
: End
: Disp TEMP
:
```

### Greater than: >

TEST menu

*numberA* > *numberB* or (*expressionA*) > (*expressionB*)

Tests whether the condition is true or false. The arguments must be real numbers.

- If true (*numberA* > *numberB*), returns **1**.
- If false (*numberA* ≤ *numberB*), returns **0**.

2 > 0	<b>[ENTER]</b>	1
88 > 123	<b>[ENTER]</b>	0
-5 > -5	<b>[ENTER]</b>	0
(20*5/2) > (18*2)	<b>[ENTER]</b>	1



<i>number</i> > <i>list</i>	$1 > \{1, -6, 10\}$ <input type="text" value="ENTER"/>	{0 1 0}
Returns a list of <b>1s</b> and/or <b>0s</b> to indicate if <i>number</i> is > the corresponding element in <i>list</i> .		
<i>listA</i> > <i>listB</i>	$\{1, 5, 9\} > \{1, -6, 10\}$ <input type="text" value="ENTER"/>	{0 1 0}
Returns a list of <b>1s</b> and/or <b>0s</b> to indicate if each element in <i>listA</i> is > the corresponding element in <i>listB</i> .		

### Greater than or equal to: $\geq$

TEST menu

<i>numberA</i> $\geq$ <i>numberB</i> or ( <i>expressionA</i> ) $\geq$ ( <i>expressionB</i> )	$2 \geq 0$ <input type="text" value="ENTER"/>	1
Tests whether the condition is true or false. The arguments must be real numbers.		
	$88 \geq 123$ <input type="text" value="ENTER"/>	0
	$-5 \geq -5$ <input type="text" value="ENTER"/>	1
	$(20 * 5 / 2) \geq (18 * 2)$ <input type="text" value="ENTER"/>	1
<i>number</i> $\geq$ <i>list</i>	$1 \geq \{1, -6, 10\}$ <input type="text" value="ENTER"/>	{1 1 0}
Returns a list of <b>1s</b> and/or <b>0s</b> to indicate if <i>number</i> is $\geq$ the corresponding element in <i>list</i> .		
<i>listA</i> $\geq$ <i>listB</i>	$\{1, 5, 9\} \geq \{1, -6, 10\}$ <input type="text" value="ENTER"/>	{1 1 0}
Returns a list of <b>1s</b> and/or <b>0s</b> to indicate if each element in <i>listA</i> is $\geq$ the corresponding element in <i>listB</i> .		

### GridOff

† graph format screen

#### GridOff

Turns off grid format so that grid points are not displayed.

## GridOn

† graph format screen

## GridOn

Turns on grid format so that grid points are displayed in rows and columns corresponding to the tick marks on each axis.

## GrStl(

CATALOG

## GrStl(*function#*,*graphStyle#*)

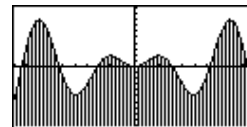
Sets the graph style for *function#*. For *graphStyle#*, specify an integer from 1 through 7:

- 1 = (line)      4 = (below)      7 = (dot)  
 2 = (thick)      5 = (path)  
 3 = (above)      6 = (animate)

Depending on the graphing mode, some graph styles may not be available.

In **Func** graphing mode:

y1=x sin x  Done  
 GrStl(1,4)  Done  
 ZStd



## h

BASE TYPE menu

## *integer*h

Designates a real *integer* as hexadecimal, regardless of the number base mode setting.

In **Dec** number base mode:

10h  16  
 10h+10  26

## Hex

† mode screen

## Hex

Sets hexadecimal number base mode. Results are displayed with the **h** suffix. In any number base mode, you can designate an appropriate value as binary, decimal, hexadecimal, or octal by using the **b**, **d**, **h**, or **o** designator, respectively, from the BASE TYPE menu.

To enter hexadecimal numbers **A** through **F**, use the BASE A-F menu. Do not use  to type a letter.

In **Hex** number base mode:

F+10b+10o+10d  23h

## ►Hex

BASE CONV menu

*number*►Hex  
*list*►Hex  
*matrix*►Hex  
*vector*►Hex

Returns the hexadecimal equivalent of the real or complex argument.

In **Bin** number base mode:

```
1010*1110 [ENTER]      10001100b
Ans►Hex [ENTER]        8Ch
{100,101,110}►Hex [ENTER]  {4h 5h 6h}
```

## Hist

† STAT DRAW menu

**Hist** *xList*,*frequencyList*

Draws a histogram on the current graph, using the real data in *xList* and the frequencies in *frequencyList*.

**Hist** *xList*

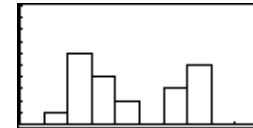
Uses frequencies of 1.

**Hist**

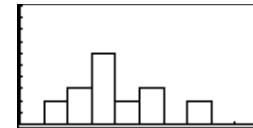
Uses the data in built-in variables **xStat** and **fStat**. These variables must contain valid data of the same dimension; otherwise, an error occurs.

Starting with a **ZStd** graph screen:

```
{1,2,3,4,6,7}►XL [ENTER]      {1 2 3 4 6 7}
{1,6,4,2,3,5}►FL [ENTER]      {1 6 4 2 3 5}
0►xMin:0►yMin [ENTER]        0
Hist XL,FL [ENTER]
```



```
{1,1,2,2,2,2,3,3,3,3,3,3,4,4,5,5,5,7,7}►XL [ENTER]
{1 1 2 2 2 2 3 3 3 3 3 3 4 4 5 5 5 7 7}►XL [ENTER]
C1Drw:Hist XL [ENTER]
```



### Horiz

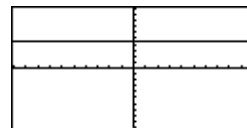
† GRAPH DRAW menu

### Horiz *yValue*

Draws a horizontal line on the current graph at *yValue*.

In a **ZStd** graph screen:

Horiz 4.5



### IAsk

CATALOG

### IAsk

Sets the table so that the user can enter individual values for the independent variable.

### IAuto

CATALOG

### IAuto

Sets the table so that the TI-86 generates the independent-variable values automatically, based on values entered for **TblStart** and **ΔTbl**.

### ident

MATRIX OPS menu

### ident *dimension*

Returns the identity matrix of *dimension* rows  $\times$  *dimension* columns.

ident 4

[[1 0 0 0]  
[0 1 0 0]  
[0 0 1 0]  
[0 0 0 1]]

**If**

‡ program editor  
CTL menu

**:if** *condition*  
**:command-if-true**  
**:commands**

If *condition* is true, executes *command-if-true*. Otherwise, skips *command-if-true*. The *condition* is true if it evaluates to any nonzero number, or false if it evaluates to zero.

To execute multiple commands if *condition* is true, use **If:Then:End** instead.

**:if** *condition*  
**:Then**  
**:commands-if-true**  
**:End**  
**:commands**

If *condition* is true (nonzero), executes *commands-if-true* from **Then** to **End**. Otherwise, skips *commands-if-true* and continues with the next command following **End**.

Program segment:

```

:
:If x<0
:Disp "x is negative"
:

```

Program segment:

```

:
:If x<0
:Then
:Disp "x is negative"
:abs(x)→x
:End
:

```

```

:if condition
:Then
:commands-if-true
:Else
:commands-if-false
:End
:commands

```

If *condition* is true (nonzero), executes *commands-if-true* from **Then** to **Else** and then continues with the next command following **End**.

If *condition* is false (zero), executes *commands-if-false* from **Else** to **End** and then continues with the next command following **End**.

```

Program segment:
:
:If x<0
:Then
: Disp "x is negative"
:Else
: Disp "x is positive or zero"
:End
:

```

## imag

CPLX menu

---

**imag** (*complexNumber*)

Returns the imaginary (nonreal) part of *complexNumber*. The imaginary part of a real number is always 0.

**imag** (*real,imaginary*) returns *imaginary*.

**imag** (*magnitude∠angle*) returns *magnitude sin angle*.

**imag** *complexList*

**imag** *complexMatrix*

**imag** *complexVector*

Returns a list, matrix, or vector in which each element is the imaginary part of the original argument.

---

```

imag (3,4) [ENTER] 4
imag (3∠4) [ENTER] -2.27040748592
imag {-2,(3,4),(3∠4)} [ENTER]
{0 4 -2.27040748592}

```

**InpSt**

‡ program editor  
I/O menu

**InpSt** *promptString,variable*

Pauses a program, displays *promptString*, and waits for the user to enter a response. The response is stored to *variable* always as a string. When entering the response, the user should not enter quotation marks.

To prompt for a number or expression instead of a string, use **Input**.

**InpSt** *variable*

Displays ? as the prompt.

Program segment:

```
⋮
:InpSt "Enter your name:",STR
⋮
```

**Input**

‡ program editor  
I/O menu

**Input** *promptString,variable*

Pauses a program, displays *promptString*, and waits for the user to enter a response. The response is stored to *variable* in the form in which the user enters it.

- A number or expression is stored as a number or expression.
- A list, vector, or matrix is stored as a list, vector, or matrix.
- An entry enclosed in " marks is stored as a string.

**Input** *variable*

Displays ? as the prompt.

Program segment:

```
⋮
:Input "Enter test score:",SCR
⋮
```

**Input**

Pauses a program, displays the graph screen, and lets the user update **x** and **y** (or **r** and  $\theta$  in **PolarGC** graph format) by moving the free-moving cursor. To resume the program, press **ENTER**.

Program segment in **RectGC** graph format:

```

:
:Input
:Disp x,y
:

```

**Input "CBLGET",variable**

Receives list data sent from a CBL or CBR System and stores it to *variable* on the TI-86. Use this "**CBLGET**" syntax for both CBL and CBR.

Input "CBLGET",L1 **ENTER** Done

You can receive data also by using **Get**( as described on page 299.

**int**

MATH NUM menu

**int number** or **int (expression)**

Returns the largest integer  $\leq$  *number* or *expression*. The argument can be real or complex.

int 23.45 **ENTER** 23

int -23.45 **ENTER** -24

For a negative non-integer, **int** returns the integer that is one less than the integer part of the number. To return the exact integer part, use **iPart** instead.

**int list**

**int matrix**

**int vector**

Returns a list, matrix, or vector in which each element is the largest integer less than or equal to the corresponding element in the specified argument.

```

[[[1.25,-23.45]][-99,47.15]]>MAT
ENTER [[1.25 -23.45]
[-99 47.15 ]]

```

int MAT **ENTER** [[1 -24]
[-99 47 ]]

---



<b>inter(</b> † MATH menu	<b>inter</b> ( $x1,y1,x2,y2,xValue$ )	Using points (3,5) and (4,4), find the y value at $x=1$ : inter(3,5,4,4,1) <b>[ENTER]</b>	7
	Calculates the line through points ( $x1,y1$ ) and ( $x2,y2$ ) and then interpolates or extrapolates a <b>y</b> value for the specified $xValue$ .		
	<b>inter</b> ( $y1,x1,y2,x2,yValue$ )	Using points (-4,-7) and (2,6), find the x value at $y=10$ : inter(-7,-4,6,2,10) <b>[ENTER]</b>	3.84615384615
	Interpolates or extrapolates an <b>x</b> value for the specified $yValue$ . Notice that points ( $x1,y1$ ) and ( $x2,y2$ ) must be entered as ( $y1,x1$ ) and ( $y2,x2$ ).		
<b>Inverse:</b> $^{-1}$ <b>[2nd]</b> <b>[<math>x^{-1}</math>]</b>	$number^{-1}$ or $(expression)^{-1}$	$5^{-1}$ <b>[ENTER]</b>	.2
	Returns 1 divided by a real or complex $number$ , where $number \neq 0$ .	$(10*6)^{-1}$ <b>[ENTER]</b>	.016666666667
	$list^{-1}$	$\{-.5,10,2/8\}^{-1}$ <b>[ENTER]</b>	$\{-2 .1 4\}$
	Returns a list in which each element is 1 divided by the corresponding element in $list$ .		
	$squareMatrix^{-1}$	$[[1,2][3,4]]^{-1}$ <b>[ENTER]</b>	$[[ -2 1 ] [ 1.5 -.5 ]]$
	Returns an inverted $squareMatrix$ , where $\det \neq 0$ .		
<b>iPart</b> MATH NUM menu	<b>iPart</b> $number$ or <b>iPart</b> ( $expression$ )	iPart 23.45 <b>[ENTER]</b>	23
	Returns the integer part of $number$ or $expression$ . The argument can be real or complex.	iPart -23.45 <b>[ENTER]</b>	-23

<b>iPart</b> <i>list</i>	<code>[[1.25,-23.45][-99.5,47.15]]&gt;MAT</code>
<b>iPart</b> <i>matrix</i>	<code>ENTER</code> <code>[[1.25 -23.45]</code>
<b>iPart</b> <i>vector</i>	<code>[-99.5 47.15 ]]</code>
Returns a list, matrix, or vector in which each element is the integer part of the corresponding element in the specified argument.	<code>iPart MAT</code> <code>ENTER</code> <code>[[1 -23]</code> <code>[-99 47 ]]</code>

**IS>(**

‡ program editor  
CTL menu

**:IS>(***variable,value*)  
:*command-if-variable*≤*value*  
:*commands*

Increments *variable* by 1. If the result is > *value*, skips *command-if-variable*≤*value*.

If the result is ≤ *value*, then *command-if-variable*≤*value* is executed.

*variable* cannot be a built-in variable.

Program segment:  
:  
:0>A  
:Lbl Start  
:Disp A  
:IS>(A,5)  
:Goto Start  
:Disp "A is now >5"  
:  
:

**LabelOff**

† graph format screen

**LabelOff**  
Turns off axes labels.

**LabelOn**

† graph format screen

**LabelOn**  
Turns on axes labels.

### Lbl

‡ program editor  
CTL menu

#### Lbl *label*

Creates a *label* of up to eight characters. A program can use a **Goto** instruction to transfer control (branch) to a specified label.

*InpSt stores input as a string, so be sure to store a string to the password variable.*

Program segment, assuming a correct password has already been stored to the **password** variable:

```

:
:
: Lbl Start
: InpSt "Enter password:",PSW
: If PSW≠password
: Goto Start
: Disp "Welcome"
:
:

```

### lcm(

MATH MISC menu

#### lcm(*integerA*,*integerB*)

Returns the least common multiple of two nonnegative integers.

```

lcm(5,2) [ENTER] 10
lcm(6,9) [ENTER] 18
lcm(18,33) [ENTER] 198

```

### LCust(

‡ program editor  
CTL menu

#### LCust(*item#*,"*title*"[,*item#*,"*title*", ...])

Loads (defines) the TI-86's custom menu, which is displayed when the user presses [CUSTOM]. The menu can have up to 15 items, shown in three groups of five items. For each *item#*/*title* pair:

- *item#* — integer from 1 through 15 that identifies the item's position in the menu. The item numbers must be specified in order, but you can skip numbers.
- "*title*" — string with up to 8 characters (not counting the quotes) that will be pasted to the current cursor location when the item is selected. This can be a variable name, expression, function name, program name, or any text string.

Program segment:

```

:
:
: LCust(1,"t",2,"Q'1",3,"Q'2",4,"R
: K",5,"Euler",6,"QI1",7,"QI2",8,"t
: Min")
:
:

```

After executed and when the user presses [CUSTOM]:



Less than: < TEST menu	$numberA < numberB$ or $(expressionA) < (expressionB)$	2 < 0 <input type="button" value="ENTER"/>	0
	Tests whether the condition is true or false. The arguments must be real numbers.	88 < 123 <input type="button" value="ENTER"/>	1
	<ul style="list-style-type: none"> <li>• If true (<math>numberA &lt; numberB</math>), returns <b>1</b>.</li> <li>• If false (<math>numberA \geq numberB</math>), returns <b>0</b>.</li> </ul>	-5 < -5 <input type="button" value="ENTER"/>	0
		(20*5/2) < (18*3) <input type="button" value="ENTER"/>	1
	$number < list$	1 < {1,-6,10} <input type="button" value="ENTER"/>	{0 0 1}
	Returns a list of <b>1s</b> and/or <b>0s</b> to indicate if <i>number</i> is < the corresponding element in <i>list</i> .		
	$listA < listB$	{1,5,9} < {1,-6,10} <input type="button" value="ENTER"/>	{0 0 1}
	Returns a list of <b>1s</b> and/or <b>0s</b> to indicate if each element in <i>listA</i> is < the corresponding element in <i>listB</i> .		
Less than or equal to: ≤ TEST menu	$numberA \leq numberB$ or $(expressionA) \leq (expressionB)$	2 ≤ 0 <input type="button" value="ENTER"/>	0
	Tests whether the condition is true or false. The arguments must be real numbers.	88 ≤ 123 <input type="button" value="ENTER"/>	1
	<ul style="list-style-type: none"> <li>• If true (<math>numberA \leq numberB</math>), returns <b>1</b>.</li> <li>• If false (<math>numberA &gt; numberB</math>), returns <b>0</b>.</li> </ul>	-5 ≤ -5 <input type="button" value="ENTER"/>	1
		(20*5/2) ≤ (18*3) <input type="button" value="ENTER"/>	1
	$number \leq list$	1 ≤ {1,-6,10} <input type="button" value="ENTER"/>	{1 0 1}
	Returns a list of <b>1s</b> and/or <b>0s</b> to indicate if <i>number</i> is ≤ the corresponding element in <i>list</i> .		
	$listA \leq listB$	{1,5,9} ≤ {1,-6,10} <input type="button" value="ENTER"/>	{1 0 1}
	Returns a list of <b>1s</b> and/or <b>0s</b> to indicate if each element in <i>listA</i> is ≤ the corresponding element in <i>listB</i> .		

## LgstR

STAT CALC menu

Built-in equation variables such as **y1**, **r1**, and **xt1** are case-sensitive. Do not use **Y1**, **R1**, and **XT1**.

**LgstR** returns a **tolMet** value that indicates if the result meets the TI-86's internal tolerance.

- If **tolMet=1**, the result is within the internal tolerance.
- If **tolmet=0**, the result is outside the internal tolerance, although it may be useful for general purposes.

**LgstR** [*iterations*,]*xList*,*yList*,*frequencyList*,*equationVariable*

Fits a logistic regression model ( $y=a/(1+be^{cx})+d$ ) to real data pairs in *xList* and *yList* and frequencies in *frequencyList*. The regression equation is stored to *equationVariable*, which must be a built-in equation variable such as **y1**, **r1**, and **xt1**. The equation's coefficients always are stored as a list to built-in variable **PRegC**.

The number of *iterations* is optional. If omitted, 64 is the default. A large number of *iterations* may produce more accurate results but may require longer calculation times. A smaller number may produce less accurate results but with shorter calculation times.

Values used for *xList*, *yList*, and *frequencyList* are stored automatically to built-in variables **xStat**, **yStat**, and **fStat**, respectively. The regression equation is stored also to built-in equation variable **RegEq**.

**LgstR** [*iterations*,]*xList*,*yList*,*equationVariable*

Uses frequencies of 1.

**LgstR** [*iterations*,]*xList*,*yList*,*frequencyList*

Stores the regression equation to **RegEq** only.

**LgstR** [*iterations*,]*xList*,*yList*

Uses frequencies of 1, and stores the regression equation to **RegEq** only.

In **Func** graphing mode:

```

{1,2,3,4,5,6}→L1 [ENTER]
                                {1 2 3 4 5 6}
{1,1.3,2.5,3.5,4.5,4.8}→L2 [ENTER]
                                {1 1.3 2.5 3.5 4.5 4...}
LgstR L1,L2,y1 [ENTER]

```

```

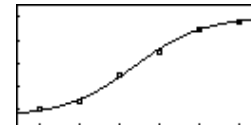
LogisticReg
y=a/(1+be^(cx))+d
n=6
tolMet=1
PRegC=
{4.31285605279 51.75...

```

```

Plot1(1,L1,L2) [ENTER]
ZData [ENTER] Done

```



**LgstR** [*iterations*,]*equationVariable*

Uses **xStat**, **yStat**, and **fStat** for *xList*, *yList*, and *frequencyList*, respectively. These built-in variables must contain valid data of the same dimension; otherwise, an error occurs. The regression equation is stored to *equationVariable* and **RegEq**.

**LgstR** [*iterations*]

Uses **xStat**, **yStat**, and **fStat**, and stores the regression equation to **RegEq** only.

**Line**(

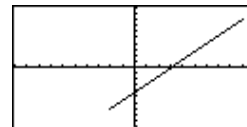
† GRAPH DRAW menu

**Line**(*x1,y1,x2,y2*)

Draws a line from point (*x1,y1*) to (*x2,y2*).

In **Func** graphing mode and a **ZStd** graph screen:

Line(-2,-7,9,8)



**LinR**

STAT CALC menu

Built-in equation variables such as **y1**, **r1**, and **xt1** are case-sensitive. Do not use **Y1**, **R1**, and **XT1**.

**LinR** *xList,yList,frequencyList,equationVariable*

Fits a linear regression model ( $y=a+bx$ ) to real data pairs in *xList* and *yList* and frequencies in *frequencyList*. The regression equation is stored to *equationVariable*, which must be a built-in equation variable such as **y1**, **r1**, and **xt1**.

Values used for *xList*, *yList*, and *frequencyList* are stored automatically to built-in variables **xStat**, **yStat**, and **fStat**, respectively. The regression equation is stored also to built-in equation variable **RegEq**.

**LinR** *xList,yList,equationVariable*

Uses frequencies of 1.

**LinR** *xList,yList,frequencyList*

Stores the regression equation to **RegEq** only.

**LinR** *xList,yList*

Uses frequencies of 1, and stores the regression equation to **RegEq** only.

**LinR** *equationVariable*

Uses **xStat**, **yStat**, and **fStat** for *xList*, *yList*, and *frequencyList*, respectively. These built-in variables must contain valid data of the same dimension; otherwise, an error occurs. The regression equation is stored to *equationVariable* and **RegEq**.

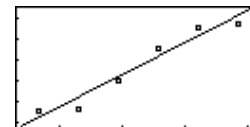
In **Func** graphing mode:

```
{1,2,3,4,5,6}→L1 [ENTER]
                                     {1 2 3 4 5 6}
{4.5,4.6,6,7.5,8.5,8.7}→L2 [ENTER]
                                     {4.5 4.6 6 7.5 8.5 8.7}
LinR L1,L2,y1 [ENTER]
```

```
LinReg
y=a+bx
a=3.21333333
b=.977142857
corr=.97454752
n=6
```

```
Plot1(1,L1,L2) [ENTER]
ZData [ENTER]
```

Done



**LinR**

Uses **xStat**, **yStat**, and **fStat**, and stores the regression equation to **RegEq** only.

List entry: { }

LIST menu

$\{element1, element2, \dots\}$	{1,2,3}→L1 [ENTER] (1 2 3)
Defines a list in which each element is a real or complex number or variable.	In <b>RectC</b> complex number mode: {3,(2,4),8*2}→L2 [ENTER] {(3,0) (2,4) (16,0)}

**li►vc**

LIST OPS menu

VECTR OPS menu

<b>li►vc list</b>	li►vc {2,7,-8,0} [ENTER] [2 7 -8 0]
Returns a vector converted from a real or complex <i>list</i> .	

**In**

[LN]

<b>In number</b> or <b>In (expression)</b>	ln 2 [ENTER] .69314718056
Returns the natural logarithm of a real or complex number or <i>expression</i> .	ln (36.4/3) [ENTER] 2.49595648597
	In <b>RectC</b> complex number mode:
<b>In list</b>	ln -3 [ENTER] (1.09861228867,3.141...
Returns a list in which each element is the natural logarithm of the corresponding element in <i>list</i> .	ln {2,3} [ENTER] { .69314718056 1.0986...

**Ingth**

STRNG menu

<b>Ingth string</b>	Ingth "The answer is:" [ENTER] 14
Returns the length (number of characters) of <i>string</i> . The character count includes spaces but not quotation marks.	"The answer is:"→STR [ENTER] The answer is: Ingth STR [ENTER] 14



**LnR**

STAT CALC menu

*Built-in equation variables such as **y1**, **r1**, and **xt1** are case-sensitive. Do not use **Y1**, **R1**, and **XT1**.*

**LnR** *xList,yList,frequencyList,equationVariable*

Fits a logarithmic regression model ( $y=a+b \ln x$ ) to the real data pairs in *xList* and *yList* (**x** values must be  $> 0$ ) and frequencies in *frequencyList*. The regression equation is stored to *equationVariable*, which must be a built-in equation variable such as **y1**, **r1**, and **xt1**.

Values used for *xList*, *yList*, and *frequencyList* are stored automatically to built-in variables **xStat**, **yStat**, and **fStat**, respectively. The regression equation is stored also to built-in equation variable **RegEq**.

**LnR** *xList,yList,equationVariable*

Uses frequencies of 1.

**LnR** *xList,yList,frequencyList*

Stores the regression equation to **RegEq** only.

**LnR** *xList,yList*

Uses frequencies of 1, and stores the regression equation to **RegEq** only.

**LnR** *equationVariable*

Uses **xStat**, **yStat**, and **fStat** for *xList*, *yList*, and *frequencyList*, respectively. These built-in variables must contain valid data of the same dimension; otherwise, an error occurs. The regression equation is stored to *equationVariable* and **RegEq**.

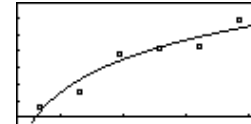
In **Func** graphing mode:

```
{1,2,3,4,5,6}→L1 [ENTER]
                                {1 2 3 4 5 6}
{.6,1.5,3.8,4.2,4.3,5.9}→L2 [ENTER]
                                {.6 1.5 3.8 4.2 4.3 5.9}
LnR L1,L2,y1 [ENTER]
```

```
LnReg
y=a+blnx
a=.252233501
b=2.85543117
corr=.962862433
n=6
```

```
Plot1(1,L1,L2) [ENTER]
ZData [ENTER]
```

Done



**LnR**

Uses **xStat**, **yStat**, and **fStat**, and stores the regression equation to **RegEq** only.

**log**

**LOG**

**log** *number* or **log** (*expression*)

Returns the logarithm of a real or complex *number* or *expression*, where:

$$10^{\text{logarithm}} = \text{number}$$

log 2 **ENTER** .301029995664

log (36.4/3) **ENTER** 1.08398012893

In **RectC** complex number mode:

log (3,4) **ENTER**  
(.698970004336, .4027...

**log** *list*

Returns a list in which each element is the logarithm of the corresponding element in *list*.

In **RectC** complex number mode:

log {-3,2} **ENTER**  
{(.47712125472, 1.364...

**LU(**

MATRIX MATH menu

**LU**(*matrix*,*LMatrixName*, *uMatrixName*, *pMatrixName*)

Calculates the Crout LU (lower-upper) decomposition of a real or complex *matrix*. The lower triangular matrix is stored in *LMatrixName*, the upper triangular matrix in *uMatrixName*, and the permutation matrix (which describes the row swaps done during the calculation) in *pMatrixName*.

$$LMatrixName * uMatrixName = pMatrixName * matrix$$

[[6,12,18][5,14,31][3,8,18]]

→MAT **ENTER**  
[[6 12 18]  
[5 14 31]  
[3 8 18]]

LU(MAT,L,U,P) **ENTER** Done

L **ENTER**  
[[6 0 0]  
[5 4 0]  
[3 2 1]]

U **ENTER**  
[[1 2 3]  
[0 1 4]  
[0 0 1]]

P **ENTER**  
[[1 0 0]  
[0 1 0]  
[0 0 1]]

**Matrix entry: [ ]**

$\boxed{2\text{nd}}$  [ ] and  $\boxed{2\text{nd}}$  [ ]

**[ [row1][row2] ... ]**

Defines a matrix entered row-by-row in which each element is a real or complex number or variable.

Enter each [row] as [element,element, ... ].

$[[1,2,3][4,5,6]]\rightarrow\text{MAT}$   $\boxed{\text{ENTER}}$

$\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix}$

**max(**

MATH NUM menu

**max(numberA,numberB)**

Returns the larger of two real or complex numbers.

$\text{max}(2.3,1.4)$   $\boxed{\text{ENTER}}$

2.3

**max(list)**

Returns the largest element in *list*.

$\text{max}(\{1,9,\pi/2,e^2\})$   $\boxed{\text{ENTER}}$

9

**max(listA,listB)**

Returns a list in which each element is the larger of the corresponding elements in *listA* and *listB*.

$\text{max}(\{1,10\},\{2,9\})$   $\boxed{\text{ENTER}}$

{2 10}

**MBox**

† STAT DRAW menu

**MBox *xList,frequencyList***

Draws a modified box plot on the current graph, using the real data in *xList* and the frequencies in *frequencyList*.

Starting with a **ZStd** graph screen:

$\{1,2,3,4,5,9\}\rightarrow\text{XL}$   $\boxed{\text{ENTER}}$

{1 2 3 4 5 9}

$\{1,1,1,4,1,1\}\rightarrow\text{FL}$   $\boxed{\text{ENTER}}$

{1 1 1 4 1 1}

$0\rightarrow\text{xMin}:0\rightarrow\text{yMin}$   $\boxed{\text{ENTER}}$

0

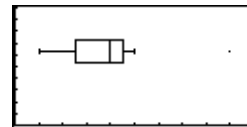
MBox XL,FL  $\boxed{\text{ENTER}}$

**MBox *xList***

Uses frequencies of 1.

**MBox**

Uses the data in built-in variables **xStat** and **fStat**. These variables must contain valid data of the same dimension; otherwise, an error occurs.



**Menu(**

‡ program editor  
CTL menu

**Menu**(*item#*,"*title1*",*label1*[,...,*item#*,"*title15*",*label15*])

Generates a menu of up to 15 items during program execution. Menus are displayed as three groups of five items. For each item:

- *item#* — integer from 1 through 15 that identifies this item's position in the menu.
- "*title*" — text string that will be displayed for this item on the menu. Typically, use from 1 through 5 characters; additional characters may not be seen on the menu.
- *label* — valid label to which program execution will branch when the user selects this item.

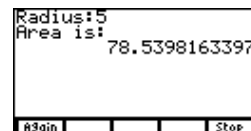
Program segment:

```

:
: Lbl A
: Input "Radius:",RADIUS
: Disp "Area is:",π*RADIUS^2
: Menu(1,"Again",A,5,"Stop",B)
: Lbl B
: Disp "The End"

```

Example when executed:

**min(**

MATH NUM menu

**min**(*numberA*,*numberB*)

Returns the smaller of two real or complex numbers.

```

min(3,-5) [ENTER] -5
min(-5.2,-5.3) [ENTER] -5.3
min(5,2+2) [ENTER] 4

```

**min**(*list*)

Returns the smallest element in *list*.

```

min({1,3,-5}) [ENTER] -5

```

**min**(*listA*,*listB*)

Returns a list in which each element is the smaller of the corresponding elements in *listA* and *listB*.

```

min({1,2,3},{3,2,1}) [ENTER] {1 2 1}

```

**mod(**

MATH NUM menu

**mod**(*numberA*,*numberB*)

Returns *numberA* modulo *numberB*. The arguments must be real.

```

mod(7,0) [ENTER] 7
mod(7,3) [ENTER] 1
mod(-7,3) [ENTER] 2
mod(7,-3) [ENTER] -2
mod(-7,-3) [ENTER] -1

```

### mRAdd(

MATRIX OPS menu

<b>mRAdd</b> ( <i>number,matrix,rowA,rowB</i> )	Returns the result of a “multiply and add row” matrix operation, where:	[[5,3,1][2,0,4][3,-1,2]]>MAT [ENTER] [[5 3 1] [2 0 4] [3 -1 2]]
a. <i>rowA</i> of a real or complex <i>matrix</i> is multiplied by a real or complex <i>number</i> .	mRAdd(5,MAT,2,3) [ENTER]	[[5 3 1] [2 0 4] [13 -1 22]]
b. The results are added to (and then stored in) <i>rowB</i> .		

### Multiplication: \*

⊗

<i>numberA</i> * <i>numberB</i>	Returns the product of two real or complex numbers.	2*5 [ENTER] 10
<i>number</i> * <i>list</i> or <i>list</i> * <i>number</i> <i>number</i> * <i>matrix</i> or <i>matrix</i> * <i>number</i> <i>number</i> * <i>vector</i> or <i>vector</i> * <i>number</i>	Returns a list, matrix, or vector in which each element is <i>number</i> multiplied by the corresponding element in <i>list</i> , <i>matrix</i> , or <i>vector</i> .	4*[10,9,8] [ENTER] {40 36 32} In RectC complex number mode: [8,1,(5,2)]*3 [ENTER] [(24,0) (3,0) (15,6)]
<i>listA</i> * <i>listB</i>	Returns a list in which each element of <i>listA</i> is multiplied by the corresponding element of <i>listB</i> . The lists must have the same dimension.	{1,2,3}*{4,5,6} [ENTER] {4 10 18}
<i>matrix</i> * <i>vector</i>	Returns a vector in which <i>matrix</i> is multiplied by <i>vector</i> . The number of columns in <i>matrix</i> must equal the number of elements in <i>vector</i> .	[[1,2,3][4,5,6]]>MAT [ENTER] [[1 2 3] [4 5 6]] MAT*[7,8,9] [ENTER] [50 122]

*matrixA*\**matrixB*

Returns a matrix in which *matrixA* is multiplied by *matrixB*. The number of columns in *matrixA* must equal the number of rows in *matrixB*.

```
[[2,2][3,4]]>MATA     [[2 2]
                                                    [3 4]]
[[1,2,3][4,5,6]]>MATB 
                                                    [[1 2 3]
                                                    [4 5 6]]
MATA*MATB     [[10 14 18]
                                                    [19 26 33]]
```

### multR(

MATRX OPS menu

**multR**(*number*,*matrix*,*row*)

Returns the result of a “row multiplication” matrix operation, where:

- The specified *row* of a real or complex *matrix* is multiplied by a real or complex *number*.
- The results are stored in the same *row*.

```
[[5,3,1]][2,0,4][3,-1,2]]>MAT
    [[5 3 1]
                               [2 0 4]
                               [3 -1 2]]
multR(5,MAT,2) 
                                                    [[5 3 1]
                                                    [10 0 20]
                                                    [3 -1 2]]
```

### nCr

MATH PROB menu

*items* **nCr** *number*

Returns the number of combinations of *items* (**n**) taken *number* (**r**) at a time. Both arguments must be real nonnegative integers.

```
5 nCr 2     10
```

---

<b>nDer(</b>	<b>nDer(expression,variable,value)</b>	For $\delta=.001$ : nDer(x^3,x,5) <b>ENTER</b>	75.000001
CALC menu	Returns an approximate numerical derivative of <i>expression</i> with respect to <i>variable</i> evaluated at a real or complex <i>value</i> . The approximate numerical derivative is the slope of the secant line through the points: ( <i>value</i> - $\delta$ ,f( <i>value</i> - $\delta$ )) and ( <i>value</i> + $\delta$ ,f( <i>value</i> + $\delta$ )) As the step value $\delta$ gets smaller, the approximation usually gets more accurate.	For $\delta=1E-4$ : nDer(x^3,x,5) <b>ENTER</b>	75
To view or set the value for $\delta$ , press <b>2nd</b> <b>[MEM]</b> <b>F4</b> to display the tolerance screen.			
	<b>nDer(expression,variable)</b>	5 $\rightarrow$ x <b>ENTER</b> nDer(x^3,x) <b>ENTER</b>	5 75
	Uses the current value of <i>variable</i> .		
<b>Negation: -</b>	- <i>number</i> or -( <i>expression</i> )	-2+5 <b>ENTER</b>	3
<b>[-]</b>	- <i>list</i>	-(2+5) <b>ENTER</b>	-7
	- <i>matrix</i>	{0,-5,5} <b>ENTER</b>	{0 5 -5}
	- <i>vector</i>		
	Returns the negative of the real or complex argument.		
<b>norm</b>	<b>norm matrix</b>	[[1,-2][3,4]] $\rightarrow$ MAT <b>ENTER</b>	[[1 -2] [-3 4 ]] 5.47722557505
MATRX MATH menu	Returns the Frobenius norm of a real or complex <i>matrix</i> , calculated as: $\sqrt{\sum(\text{real}^2+\text{imaginary}^2)}$	norm MAT <b>ENTER</b>	
VECTR MATH menu	where the sum is over all elements.		

<b>norm</b> <i>vector</i>	norm [3,4,5] <input type="text" value="ENTER"/>	7.07106781187
Returns the length of a real or complex <i>vector</i> , where: <b>norm</b> [a,b,c] returns $\sqrt{a^2+b^2+c^2}$ .		
<b>norm</b> <i>number</i> or <b>norm</b> ( <i>expression</i> )	norm -25 <input type="text" value="ENTER"/>	25
<b>norm</b> <i>list</i>	In <b>Radian</b> angle mode: norm {-25,cos $-(\pi/3)$ } <input type="text" value="ENTER"/>	{25 .5}

## Normal

† mode screen

<b>Normal</b>	In <b>Eng</b> notation mode: 123456789 <input type="text" value="ENTER"/>	123.456789E6
Sets normal notation mode.	In <b>Sci</b> notation mode: 123456789 <input type="text" value="ENTER"/>	1.23456789E8
	In <b>Normal</b> notation mode: 123456789 <input type="text" value="ENTER"/>	123456789

---



**not**

BASE BOOL menu

**not** *integer*

Returns the one's complement of a real *integer*. Internally, *integer* is represented as a 16-bit binary number. The value of each bit is flipped (0 becomes 1, and vice versa) for the one's complement.

For example, **not** 78:

```
78 = 0000000001001110b
    111111110110001b (one's complement)
```

└─ Sign bit; 1 indicates a negative number

To find the magnitude of a negative binary number, determine its two's complement (take the one's complement and then add 1). For example:

```
111111110110001b = one's complement of 78
0000000001001110b (one's complement)
+ 0000000000000001b
0000000001001111b = 79 (two's complement)
```

Therefore, **not** 78 = -79.

You can enter real numbers instead of integers, but they are truncated automatically before the comparison.

In **Dec** number base mode:

```
not 78 [ENTER] -79
```

In **Bin** number base mode:

```
not 1001110 [ENTER] 111111110110001b
Ans▶Dec [ENTER] -79d
```

Not equal to: $\neq$ TEST menu	$numberA \neq numberB$	$2+2\neq3+2$ <input type="button" value="ENTER"/>	1
	$matrixA \neq matrixB$	$2+(2\neq3)+2$ <input type="button" value="ENTER"/>	5
	$vectorA \neq vectorB$	$[1,2]\neq[3-2,-1+3]$ <input type="button" value="ENTER"/>	0
	$stringA \neq stringB$	"A" $\neq$ "a" <input type="button" value="ENTER"/>	1
	Tests whether the condition $argumentA \neq argumentB$ is true or false. Numbers, matrices, and vectors can be real or complex. If complex, the magnitude (modulus) of each element is compared. Strings are case-sensitive.		
	<ul style="list-style-type: none"> <li>• If true (<math>argumentA \neq argumentB</math>), returns <b>1</b>.</li> <li>• If false (<math>argumentA = argumentB</math>), returns <b>0</b>.</li> </ul>		
	$listA \neq listB$	$\{1,5,9\}\neq\{1,-6,9\}$ <input type="button" value="ENTER"/>	{0 1 0}
	Returns a list of <b>1</b> s and/or <b>0</b> s to indicate if each element in $listA$ is $\neq$ the corresponding element in $listB$ .		
<b>nPr</b> MATH PROB menu	$items$ <b>nPr</b> $number$	5 nPr 2 <input type="button" value="ENTER"/>	20
	Returns the number of permutations of $items$ ( <b>n</b> ) taken $number$ ( <b>r</b> ) at a time. Both arguments must be real nonnegative integers.		
<b>0</b> BASE TYPE menu	$integer$ <b>o</b>	In <b>Dec</b> number base mode: 10 <b>o</b> <input type="button" value="ENTER"/> 10 <b>o</b> +10 <input type="button" value="ENTER"/>	8 18
	Designates a real $integer$ as octal, regardless of the number base mode setting.		

**Oct**

† mode screen

**Oct**

Sets octal number base mode. Results are displayed with the **o** suffix. In any number base mode, you can designate an appropriate value as binary, decimal, hexadecimal, or octal by using the **b**, **d**, **h**, or **o** designator, respectively, from the BASE TYPE menu.

In **Oct** number base mode:

10+10b+Fh+10d  43o

**►Oct**

BASE CONV menu

*number*►**Oct**  
*list*►**Oct**  
*matrix*►**Oct**  
*vector*►**Oct**

Returns the octal equivalent of the real or complex argument.

In **Dec** number base mode:

2\*8  16

Ans►Oct  20o

{7,8,9,10}►Oct  {7o 10o 11o 12o}

**OneVar**

STAT CALC menu  
 (OneVa shows on menu)

**OneVar** *xList*, *frequencyList*

Performs one-variable statistical analysis using real data points in *xList* and frequencies in *frequencyList*.

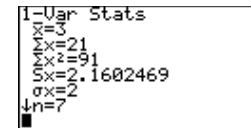
The values used for *xList* and *frequencyList* are stored automatically to built-in variables **xStat** and **fStat**, respectively.

**OneVar** *xList*

Uses frequencies of 1.

{0,1,2,3,4,5,6}►XL  {0 1 2 3 4 5 6}

OneVar XL



Scroll down to see more results.

**OneVar**

Uses **xStat** and **fStat** for *xList* and *frequencyList*. These built-in variables must contain valid data of the same dimension; otherwise, an error occurs.

**or**

BASE BOOL menu

*integerA or integerB*

Compares two real integers bit by bit. Internally, both integers are converted to binary. When corresponding bits are compared, the result is 1 if either bit is 1; the result is 0 only if both bits are 0. The returned value is the sum of the bit results.

For example,  $78 \text{ or } 23 = 95$ .

$$78 = 1001110\text{b}$$

$$23 = 0010111\text{b}$$

$$1011111\text{b} = 95$$

You can enter real numbers instead of integers, but they are truncated automatically before the comparison.

In **Dec** number base mode:78 or 23 

95

In **Bin** number base mode:1001110 or 10111 

1011111b

Ans▶Dec 

95d

**Outpt(**

‡ program editor  
I/O menu

**Outpt(*row,column,string*)**

Displays *string* beginning at *row* and *column*, where  $1 \leq \text{row} \leq 8$  and  $1 \leq \text{column} \leq 21$ .

**Outpt(*row,column,value*)**

Displays *value* beginning at the specified *row* and *column*.

**Outpt("CBLSEND",*listName*)**

Sends the contents of *listName* to the CBL or CBR System.

You can send data also by using **Send(** as described on page 350.

Program segment:

```

:
: C1LCD
: For(i,1,8)
:   Outpt(i,randInt(1,21),"A")
: End
:

```

Example result after execution:

```

A
      A      A
    A      A
  A      A
A

```

**P2Reg**

STAT CALC menu

Built-in equation variables such as **y1**, **r1**, and **xt1** are case-sensitive. Do not use **Y1**, **R1**, and **XT1**.

**P2Reg** *xList,yList,frequencyList,equationVariable*

Performs a second order polynomial regression using real data pairs in *xList* and *yList* and frequencies in *frequencyList*. The regression equation is stored to *equationVariable*, which must be a built-in equation variable such as **y1**, **r1**, and **xt1**. The equation's coefficients always are stored as a list to built-in variable **PRegC**.

Values used for *xList*, *yList*, and *frequencyList* are stored automatically to built-in variables **xStat**, **yStat**, and **fStat**, respectively. The regression equation is stored also to built-in equation variable **RegEq**.

**P2Reg** *xList,yList,equationVariable*

Uses frequencies of 1.

**P2Reg** *xList,yList,frequencyList*

Stores the regression equation to **RegEq** only.

**P2Reg** *xList,yList*

Uses frequencies of 1, and stores the regression equation to **RegEq** only.

**P2Reg** *equationVariable*

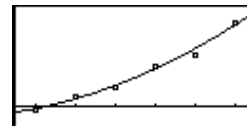
Uses **xStat**, **yStat**, and **fStat** for *xList*, *yList*, and *frequencyList*, respectively. These built-in variables must contain valid data of the same dimension; otherwise, an error occurs. The regression equation is stored to *equationVariable* and **RegEq**.

In **Func** graphing mode:

```
{1,2,3,4,5,6}→L1 [ENTER]
{1 2 3 4 5 6}
{-2,6,11,23,29,47}→L2 [ENTER]
{-2 6 11 23 29 47}
P2Reg L1,L2,y1 [ENTER]
```

```
QuadraticReg
y=ax2+bx+c
n=6
PRegC=
C.964285714286 2.564...
```

```
Plot1(1,L1,L2) [ENTER] Done
ZData [ENTER]
```



**P2Reg**

Uses **xStat**, **yStat**, and **fStat**, and stores the regression equation to **RegEq** only.

**P3Reg**

STAT CALC menu

*Built-in equation variables such as **y1**, **r1**, and **xt1** are case-sensitive. Do not use **Y1**, **R1**, and **XT1**.*

**P3Reg** *xList,yList,frequencyList,equationVariable*

Performs a third order polynomial regression using real data pairs in *xList* and *yList* and frequencies in *frequencyList*. The regression equation is stored to *equationVariable*, which must be a built-in equation variable such as **y1**, **r1**, and **xt1**. The equation's coefficients always are stored as a list to built-in variable **PRegC**.

Values used for *xList*, *yList*, and *frequencyList* are stored automatically to built-in variables **xStat**, **yStat**, and **fStat**, respectively. The regression equation is stored also to built-in equation variable **RegEq**.

**P3Reg** *xList,yList,equationVariable*

Uses frequencies of 1.

**P3Reg** *xList,yList,frequencyList*

Stores the regression equation to **RegEq** only.

**P3Reg** *xList,yList*

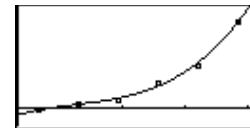
Uses frequencies of 1, and stores the regression equation to **RegEq** only.

In **Func** graphing mode:

```
{1,2,3,4,5,6}→L1 [ENTER] {1 2 3 4 5 6}
{-6,15,27,88,145,294}→L2 [ENTER] {-6 15 27 88 145 294}
P3Reg L1,L2,y1 [ENTER]
```

```
CubicReg
y=ax3+bx2+cx+d
n=6
PRegC=
{3.2037037037 -18.99...
```

```
Plot1(1,L1,L2) [ENTER] Done
ZData [ENTER]
```



**P3Reg** *equationVariable*

Uses **xStat**, **yStat**, and **fStat** for *xList*, *yList*, and *frequencyList*, respectively. These built-in variables must contain valid data of the same dimension; otherwise, an error occurs. The regression equation is stored to *equationVariable* and **RegEq**.

**P3Reg**

Uses **xStat**, **yStat**, and **fStat**, and stores the regression equation to **RegEq** only.

**P4Reg**

STAT CALC menu

*Built-in equation variables such as **y1**, **r1**, and **xt1** are case-sensitive. Do not use **Y1**, **R1**, and **XT1**.*

**P4Reg** *xList,yList,frequencyList,equationVariable*

Performs a fourth order polynomial regression using real data pairs in *xList* and *yList* and frequencies in *frequencyList*. The regression equation is stored to *equationVariable*, which must be a built-in equation variable such as **y1**, **r1**, and **xt1**. The equation's coefficients always are stored as a list to built-in variable **PRegC**.

Values used for *xList*, *yList*, and *frequencyList* are stored automatically to built-in variables **xStat**, **yStat**, and **fStat**, respectively. The regression equation is stored also to built-in equation variable **RegEq**.

**P4Reg** *xList,yList,equationVariable*

Uses frequencies of 1.

**P4Reg** *xList,yList,frequencyList*

Stores the regression equation to **RegEq** only.

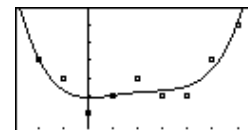
In **Func** graphing mode:

```
{-2,-1,0,1,2,3,4,5,6}→L1 [ENTER]
      {-2 -1 0 1 2 3 4 5 6}
{4,3,1,2,3,2,2,4,6}→L2 [ENTER]
      {4 3 1 2 3 2 2 4 6}
P4Reg L1,L2,y1 [ENTER]
```

```
QuarticReg
y=ax4+bx3+cx2+dx+e
n=9
PRegC=
{.014568764569 -.109...
```

```
Plot1(1,L1,L2) [ENTER]
ZData [ENTER]
```

Done





**P4Reg** *xList,yList*

Uses frequencies of 1, and stores the regression equation to **RegEq** only.

**P4Reg** *equationVariable*

Uses **xStat**, **yStat**, and **fStat** for *xList*, *yList*, and *frequencyList*, respectively. These built-in variables must contain valid data of the same dimension; otherwise, an error occurs. The regression equation is stored to *equationVariable* and **RegEq**.

**P4Reg**

Uses **xStat**, **yStat**, and **fStat**, and stores the regression equation to **RegEq** only.

**Param**

† mode screen

**Param**

Sets parametric graphing mode.

**Pause**

‡ program editor  
CTL menu

**Pause** *string***Pause** *value***Pause** *list***Pause** *matrix***Pause** *vector*

Displays the specified argument and then suspends program execution until the user presses **ENTER**.

Program segment:

```

:
:Input "Enter x:",x
:y1=x2-6
:Disp "y1 is:",y1
:Pause "Press ENTER to graph"
:ZStd
:

```

**Pause**

Suspends program execution until the user presses

 $\boxed{\text{ENTER}}$ .

**Percent: %**

MATH MISC menu

 $number\%$  or  $(expression)\%$ 

 Returns a real *number* or *expression* divided by 100.

 $5\%$   $\boxed{\text{ENTER}}$ 

.05

 $5\%*200$   $\boxed{\text{ENTER}}$ 

10

 $(10+5)\%*200$   $\boxed{\text{ENTER}}$ 

30

**pEval(**

MATH MISC menu

**pEval**(*coefficientList*,*xValue*)

 Returns the value of a polynomial (whose coefficients are given in *coefficientList*) at *xValue*.

 Evaluate  $y=2x^2+2x+3$  at  $x=5$ :

 $pEval(\{2,2,3\},5)$   $\boxed{\text{ENTER}}$ 

63

**PIOff**

STAT PLOT menu

**PIOff** [1,2,3]

Deselects the specified stat plot numbers.

 $PIOff$  1,3  $\boxed{\text{ENTER}}$ 

Done

**PIOff**

Deselects all stat plot numbers.

 $PIOff$   $\boxed{\text{ENTER}}$ 

Done

**PIOn**

STAT PLOT menu

**PIOn** [1,2,3]

Selects the specified stat plot numbers, in addition to any plot numbers that are already selected.

 $PIOn$  2,3  $\boxed{\text{ENTER}}$ 

Done

**PIOn**

Selects all stat plot numbers.

 $PIOn$   $\boxed{\text{ENTER}}$ 

Done

## Plot1( Plot2( Plot3(

† STAT PLOT menu

The syntax and descriptions to the right refer to **Plot1**(, but they apply as well to **Plot2**( and **Plot3**(.

Scatter plot  $\square$

**Plot1**(1,*xListName*,*yListName*,*mark*)

**Plot1**(1,*xListName*,*yListName*)

Defines and selects the plot using real data pairs in *xListName* and *yListName*.

The optional *mark* specifies the character used to plot the points. If you omit *mark*, a box is used.

*mark*: 1 = box ( $\square$ ) 2 = cross (+) 3 = dot ( $\bullet$ )

xyLine plot  $\square$

**Plot1**(2,*xListName*,*yListName*,*mark*)

**Plot1**(2,*xListName*,*yListName*)

Modified box plot  $\square$

**Plot1**(3,*xListName*,1 or *frequencyListName*,*mark*)

**Plot1**(3,*xListName*,1 or *frequencyListName*)

**Plot1**(3,*xListName*)

Defines and selects the plot using real data points in *xListName* with the specified frequencies. If you omit 1 or *frequencyListName*, frequencies of 1 are used.

Histogram  $\square$

**Plot1**(4,*xListName*,1 or *frequencyListName*)

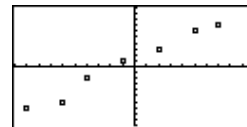
**Plot1**(4,*xListName*)

Box plot  $\square$

**Plot1**(5,*xListName*,1 or *frequencyListName*)

**Plot1**(5,*xListName*)

```
{-9,-6,-4,-1,2,5,7,10}→L1 [ENTER]
{-9 -6 -4 -1 2 5 7 1...
{-7,-6,-2,1,3,6,7,9}→L2 [ENTER]
{-7 -6 -2 1 3 6 7 9}
Plot1(1,L1,L2) [ENTER] Done
ZStd [ENTER]
```



**Pol**

† mode screen

**Pol**

CPLX menu

**Pol**

Sets polar graphing mode.

*complexNumber* ▶ **Pol**

Displays *complexNumber* in polar form (*magnitude*∠*angle*), regardless of the complex number mode.

*list* ▶ **Pol**

*matrix* ▶ **Pol**

*vector* ▶ **Pol**

Returns a list, matrix, or vector in which each element of the argument is displayed in polar form.

In **RectC** complex number mode:

$\sqrt{-2}$  [ENTER] (0,1.41421356237)

Ans▶Pol [ENTER] (1.41421356237∠1.570...

{1,√-2} [ENTER] {(1,0) (0,1.141421356...

Ans▶Pol [ENTER] {(1∠0) (1.4142135623...

**PolarC**

† mode screen

**PolarC**

Sets polar complex number mode (*magnitude*∠*angle*).

In **PolarC** complex number mode:

$\sqrt{-2}$  [ENTER] (1.41421356237∠1.570...

Polar complex: ∠

[2nd] [∠]

*magnitude*∠*angle*

Used to enter complex numbers in polar form. The *angle* is interpreted according to the current angle mode.

In **Radian** angle mode and **PolarC** complex number mode:

(1,2)+(3∠π/4) [ENTER] (5.16990542093∠.9226...

**PolarGC**

† graph format screen

**PolarGC**

Displays graph coordinates in polar form.

<b>poly</b> † [2nd] [POLY]	<b>poly</b> <i>coefficientList</i> Returns a list containing the real and complex roots of a polynomial whose coefficients are given in <i>coefficientList</i> . $a_n x^n + \dots + a_2 x^2 + a_1 x^1 + a_0 x^0 = 0$	Find the roots of $2x^3 - 8x^2 - 14x + 20 = 0$ : poly {2,-8,-14,20} [ENTER] {5 -2 1}
<b>Power: ^</b> □	$number^{power}$ or $(expression)^{(expression)}$ Returns <i>number</i> raised to <i>power</i> . The arguments can be real or complex.	$4^2$ [ENTER] 16 $2^{-5}$ [ENTER] .03125
	$listA^{listB}$ Returns a list in which each element of <i>listA</i> is raised to the power specified by the corresponding element in <i>listB</i> .	$\{2,3,4\}^{\{3,4,5\}}$ [ENTER] {8 81 1024}
	$squareMatrix^{power}$ Returns a matrix equivalent to <i>squareMatrix</i> multiplied by itself <i>power</i> number of times, where $0 \leq power \leq 255$ . This is not the same as simply raising each element to <i>power</i> .	$[[2,3][4,5]]^3$ [ENTER] [[116 153] [204 269]]
<b>Power of 10: 10^</b> [2nd] [10^x]	$10^{power}$ or $10^{(expression)}$ Returns 10 raised to <i>power</i> or <i>expression</i> , which can be real or complex.	$10^{1.5}$ [ENTER] 31.6227766017 $10^{-2}$ [ENTER] .01

**10<sup>^</sup>list** Returns a list in which each element is 10 raised to the power specified by the corresponding element in *list*. **10<sup>^</sup>{1.5,-2}**  {31.6227766017 .01}

**prod** LIST OPS menu MATH MISC menu  
**prod list** Returns the product of all real or complex elements in *list*. **prod {1,2,4,8}**  64  
**prod {2,7,-8}**  -112

**Prompt** ‡ program editor I/O menu (Promp shows on menu)  
**Prompt variableA[,variableB,...]** Prompts the user to enter a value for *variableA*, then *variableB*, and so on. Program segment:  
 ∴  
 ∴Prompt A,B,C  
 ∴

**PtChg(** † GRAPH DRAW menu  
**PtChg(x,y)** Reverses the point at graph coordinates  $(x,y)$ . PtChg(-6,2)

**PtOff(** † GRAPH DRAW menu  
**PtOff(x,y)** Erases the point at graph coordinates  $(x,y)$ . PtOff(3,5)

**PtOn(** † GRAPH DRAW menu  
**PtOn(x,y)** Draws the point at graph coordinates  $(x,y)$ . PtOn(3,5)

**PwrR**

STAT CALC menu

Built-in equation variables such as **y1**, **r1**, and **xt1** are case-sensitive. Do not use **Y1**, **R1**, and **XT1**.

**PwrR** *xList,yList,frequencyList,equationVariable*

Fits a power regression model ( $y=ax^b$ ) to positive real data pairs in *xList* and *yList*, using frequencies in *frequencyList*. The regression equation is stored to *equationVariable*, which must be a built-in equation variable such as **y1**, **r1**, and **xt1**.

Values used for *xList*, *yList*, and *frequencyList* are stored automatically to built-in variables **xStat**, **yStat**, and **fStat**, respectively. The regression equation is stored also to built-in equation variable **RegEq**.

**PwrR** *xList,yList,equationVariable*

Uses frequencies of 1.

**PwrR** *xList,yList,frequencyList*

Stores the regression equation to **RegEq** only.

**PwrR** *xList,yList*

Uses frequencies of 1, and stores the regression equation to **RegEq** only.

**PwrR** *equationVariable*

Uses **xStat**, **yStat**, and **fStat** for *xList*, *yList*, and *frequencyList*, respectively. These built-in variables must contain valid data of the same dimension; otherwise, an error occurs. The regression equation is stored to *equationVariable* and **RegEq**.

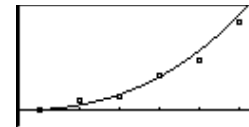
In **Func** graphing mode:

```
{1,2,3,4,5,6}→L1 [ENTER]
{1,17,21,52,75,133}→L2 [ENTER]
PwrR L1,L2,y1 [ENTER]
```

```
PwrReg
y=a*x^b
a=1.43992723
b=2.56096944
corr=.977662979
n=6
```

```
Plot1(1,L1,L2) [ENTER]
ZData [ENTER]
```

Done



**PwrR**

Uses **xStat**, **yStat**, and **fStat**, and stores the regression equation to **RegEq** only.

**PxChg(**

GRAPH DRAW menu

**PxChg**(*row,column*)

PxChg(10,95)

Reverses the pixel at (*row*, *column*), where  $0 \leq row \leq 62$  and  $0 \leq column \leq 126$ .

**PxOff(**

GRAPH DRAW menu

**PxOff**(*row,column*)

PxOff(10,95)

Erases the pixel at (*row*, *column*), where  $0 \leq row \leq 62$  and  $0 \leq column \leq 126$ .

**PxOn(**

GRAPH DRAW menu

**PxOn**(*row,column*)

PxOn(10,95)

Draws the pixel at (*row*, *column*), where  $0 \leq row \leq 62$  and  $0 \leq column \leq 126$ .

**PxTest(**

GRAPH DRAW menu

**PxTest**(*row,column*)

Assuming the pixel at (10,95) is already on:

Returns **1** if the pixel at (*row*, *column*) is on, **0** if it is off;  $0 \leq row \leq 62$  and  $0 \leq column \leq 126$ .

PxTest(10,95)  1

**rAdd(**

MATRIX OPS menu

**rAdd**(*matrix,rowA,rowB*)

Returns a matrix in which *rowA* of a real or complex *matrix* is added to (and stored in) *rowB*.

[[5,3,1][2,0,4][3,-1,2]]>MAT  
  $\begin{bmatrix} 5 & 3 & 1 \\ 2 & 0 & 4 \\ 3 & -1 & 2 \end{bmatrix}$   
 rAdd(MAT,2,3)   $\begin{bmatrix} 5 & 3 & 1 \\ 2 & 0 & 4 \\ 5 & -1 & 6 \end{bmatrix}$



**Radian**

† [2nd] [MODE]

**Radian entry:**  $\uparrow$ 

MATH ANGLE menu

**Radian**

Sets radian angle mode.

 $number^{\uparrow}$  or  $(expression)^{\uparrow}$ Designates a real *number* or *expression* as radians, regardless of the angle mode setting. $list^{\uparrow}$ Designates each element in a real *list* as radians.In **Radian** angle mode:
 $\sin(\pi/2)$  [ENTER] 1  
 $\sin 90$  [ENTER] .893996663601
In **Degree** angle mode:
 $\cos(\pi/2)$  [ENTER] .999624216859  
 $\cos(\pi/2)^{\uparrow}$  [ENTER] 0

 $\cos\{\pi/2, \pi\}^{\uparrow}$  [ENTER] {0 -1}
**rand**

MATH PROB menu

**rand**

Returns a random number between 0 and 1.

To control a random number sequence, first store an integer seed value to **rand** (such as  $0 \rightarrow \mathbf{rand}$ ).

You may have different results for the first two examples:

 $\mathbf{rand}$  [ENTER] .943597402492

 $\mathbf{rand}$  [ENTER] .146687829222

 $0 \rightarrow \mathbf{rand} : \mathbf{rand}$  [ENTER] .943597402492

 $0 \rightarrow \mathbf{rand} : \mathbf{rand}$  [ENTER] .943597402492
**randBin(**MATH PROB menu  
(randBi shows on menu)**randBin(#ofTrials,probabilityOfSuccess,#ofSimulations)**Returns a list of random integers from a binomial distribution, where  $\#ofTrials \geq 1$  and  $0 \leq \#ofSimulations \leq 1$ . The  $\#ofSimulations$  is an integer  $\geq 1$  that specifies the number of integers returned in the list.A seed value stored to **rand** also affects **randBin(**.
 $1 \rightarrow \mathbf{rand} : \mathbf{randBin}(5, .2, 3)$  [ENTER] {0 3 2}
**randBin(#ofTrials,probabilityOfSuccess)**

Returns a single random integer.

 $0 \rightarrow \mathbf{rand} : \mathbf{randBin}(5, .2)$  [ENTER] 1

## randInt(

MATH PROB menu  
(randIn shows on menu)

---

**randInt**(*lower,upper,#ofTrials*)

Returns a list of random integers bound by the specified integers,  $lower \leq integer \leq upper$ . The *#ofTrials* is an integer  $\geq 1$  that specifies the number of integers returned in the list.

1→rand:randInt(1,10,3)  {8 9 3}

A seed value stored to **rand** also affects **randInt**(.

---

**randInt**(*lower,upper*)

Returns a single random integer.

0→rand:randInt(1,10)  10

---

## randM(

MATRIX OPS menu

---

**randM**(*rows,columns*)

Returns a *rows* × *columns* matrix filled with random one-digit integers (-9 to 9).

0→rand:randM(2,3)   $\begin{bmatrix} 4 & -2 & 0 \\ -7 & 8 & 8 \end{bmatrix}$

---

## randNorm(

MATH PROB menu  
(randN shows on menu)

---

**randNorm**(*mean,stdDeviation,#ofTrials*)

Returns a list of random numbers from a normal distribution specified by *mean* and *stdDeviation*. The *#ofTrials* is an integer  $\geq 1$  that specifies how many numbers are returned. Each returned number could be any real number, but most will be within the interval:  $[mean-3(stdDeviation), mean+3(stdDeviation)]$ .

1→rand:randNorm(0,1,3)  {- .660585055265 -1.0...

A seed value stored to **rand** also affects **randNorm**(.

---

**randNorm**(*mean,stdDeviation*)

Returns a single random number.

0→rand:randNorm(0,1)  -1.58570962271

---

**RcGDB**

† GRAPH menu

**RcGDB** *graphDataBaseName*

Restores all settings stored in *graphDataBaseName*.  
For a list of settings, refer to **StGDB** on page 361.

**RcPic**

† GRAPH menu

**RcPic** *pictureName*

Displays the current graph and adds the picture stored  
in *pictureName*.

**real**

CPLX menu

**real** (*complexNumber*)Returns the real part of *complexNumber*.**real** (*real,imaginary*) returns *real*.**real** (*magnitude∠angle*) returns *magnitude\*cos (angle)*.**real** *complexList***real** *complexMatrix***real** *complexVector*

Returns a list, matrix, or vector in which each element  
is the real part of the corresponding element in the  
argument.

In **Radian** angle mode:real (3,4)  3real (3∠4)  -1.96093086259In **Radian** angle mode:real {-2,(3,4),(3∠4)}   
{-2 3 -1.96093086259}**►Rec**

CPLX menu

*complexNumber*►**Rec**

Displays *complexNumber* in rectangular form  
(*real,imaginary*) regardless of the complex number  
mode.

In **PolarC** complex number mode:√-2  (1.41421356237∠1.570...  
Ans►Rec  (0,1.41421356237)

*complexList*►**Rec**  
*complexMatrix*►**Rec**  
*complexVector*►**Rec**

Returns a list, matrix, or vector in which each element of the argument is displayed in rectangular form.

In **PolarC** complex number mode:  
 $[(3\angle\pi/6),\sqrt{-2}]$  **[ENTER]**  
 $[(3\angle.523598775598) (...]$   
 Ans►**Rec** **[ENTER]**  $[(2.59807621135,1.5)...$

### RectC

† mode screen

### RectC

Sets rectangular complex number mode (*real,imaginary*).

In **RectC** complex number mode:  
 $\sqrt{-2}$  **[ENTER]**  $(0,1.41421356237)$

### RectGC

† graph format screen

### RectGC

Displays graph coordinates in rectangular form.

### RectV

† mode screen

### RectV

Sets rectangular vector coordinate mode **[x y z]**.

In **RectV** vector coordinate mode:  
 $3*[4\angle 5]$  **[ENTER]**  
 $[3.40394622556 -11.5...$

### ref

MATRIX OPS menu

### ref *matrix*

Returns the row-echelon form of a real or complex *matrix*. The number of columns must be greater than or equal to the number of rows.

$[[4,5,6][7,8,9]]\rightarrow$ **MAT** **[ENTER]**  
 $[[4 5 6]$   
 $[7 8 9]]$   
 ref **MAT** **[ENTER]**  
 $[[[1 1.14285714286 1...$   
 $[0 1 2 ...$

**Repeat**

‡ program editor  
CTL menu  
(Repea shows  
on menu)

**:Repeat** *condition*  
**:commands-to-repeat**  
**:End**  
**:commands**

Executes *commands-to-repeat* until *condition* is true.

Program segment:  
:  
:6>N  
:1>Fact  
:Repeat N<1  
: Fact\*N>Fact  
: N-1>N  
:End  
:Disp "6!=",Fact  
:

**Return**

‡ program editor  
CTL menu  
(Retur shows  
on menu)

**Return**

In a subroutine, exits the subroutine and returns to the calling program. In the main program, stops execution and returns to the home screen.

Program segment in the calling program:  
:  
:Input "Diameter:",DIAM  
:Input "Height:",HT  
:AREACIRC  
:VOL=AREA\*HT  
:Disp "Volume =",VOL  
:

AREACIRC subroutine program:  
PROGRAM:AREACIRC  
:RADIUS=DIAM/2  
:AREA= $\pi$ \*RADIUS<sup>2</sup>  
:Return

**RK**

‡ graph format screen  
(scroll down to  
second screen)

**RK**

In **DifEq** graphing mode, uses an algorithm based on the Runge-Kutta method to solve differential equations. Typically, **RK** is more accurate than **Euler** but takes longer to find the solutions.

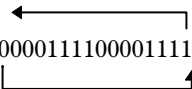
<b>rnorm</b> MATRX MATH menu	<b>rnorm</b> <i>matrix</i>	$\begin{bmatrix} [-5,6,-7] & [3,3,9] & [9,-9,-7] \\ \rightarrow \text{MAT} & \text{[ENTER]} & \end{bmatrix}$ $\begin{bmatrix} [-5 & 6 & -7] \\ [3 & 3 & 9] \\ [9 & -9 & -7] \end{bmatrix}$	25
	<b>rnorm</b> <i>vector</i>	$\text{rnorm [15,-18,7] [ENTER]}$	18
	<p>Returns the row norm of a real or complex <i>matrix</i>. For each row, <b>rnorm</b> sums the absolute values (magnitudes of complex elements) of all elements on that row. The returned value is the largest of the sums.</p> <p>Returns the largest absolute value (or magnitude) in a real or complex <i>vector</i>.</p>		
Root: $\sqrt[x]{\quad}$ MATH MISC menu	$x^{\text{th}}\text{root}$ $\sqrt[x]{\text{number}}$ or $x^{\text{th}}\text{root}$ $\sqrt[x]{\text{expression}}$	$5^{\sqrt{32}} \text{ [ENTER]}$	2
	$x^{\text{th}}\text{root}$ $\sqrt[x]{\text{list}}$	$5^{\sqrt{\{32,243\}}} \text{ [ENTER]}$	{2 3}
	$x^{\text{th}}\text{rootList}$ $\sqrt[x]{\text{list}}$	$\{5,2\}^{\sqrt{\{32,25\}}} \text{ [ENTER]}$	{2 5}
	<p>Returns the <math>x^{\text{th}}\text{root}</math> of <i>number</i> or <i>expression</i>. The arguments can be real or complex.</p> <p>Returns a list in which each element is the <math>x^{\text{th}}\text{root}</math> of the corresponding element in <i>list</i>.</p> <p>Returns a list in which each element is the root specified by the corresponding elements in <math>x^{\text{th}}\text{rootList}</math> and <i>list</i>.</p>		

**rotL**

BASE BIT menu

**rotL** *integer*

Returns a real *integer* with bits rotated one to the left. Internally, *integer* is represented as a 16-bit binary number. When the bits are rotated left, the leftmost bit rotates to the rightmost bit.



**rotL** 0000111100001111b = 0001111000011110b

**rotL** is not valid in **Dec** number base mode. To enter hexadecimal numbers **A** through **F**, use the BASE A-F menu. Do not use **[ALPHA]** to type a letter.

In **Bin** number base mode:

```
rotL 0000111100001111 [ENTER]
                        1111000011110b
```

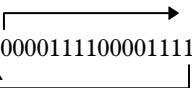
Leading zeros are not displayed.

**rotR**

BASE BIT menu

**rotR** *integer*

Returns a real *integer* with bits rotated one to the right. Internally, *integer* is represented as a 16-bit binary number. When the bits are rotated right, the rightmost bit rotates to the leftmost bit.



**rotR** 0000111100001111b = 1000011110000111b

**rotR** is not valid in **Dec** number base mode. To enter hexadecimal numbers **A** through **F**, use the BASE A-F menu. Do not use **[ALPHA]** to type a letter.

In **Bin** number base mode:

```
rotR 0000111100001111 [ENTER]
                        1000011110000111b
```

**round(**

MATH NUM menu

**round**(*number*,#ofDecimals)

**round**(*number*)

Returns a real or complex *number* rounded to the specified #ofDecimals (0 to 11). If #ofDecimals is omitted, *number* is rounded to 12 decimal places.

**round**(*list*,#ofDecimals)

**round**(*matrix*,#ofDecimals)

**round**(*vector*,#ofDecimals)

Returns a list, matrix, or vector in which each element is the rounded value of the corresponding element in the argument. #ofDecimals is optional.

round( $\pi$ ,4) **ENTER** 3.1416

round( $\pi/4$ ,4) **ENTER** .7854

round( $\pi/4$ ) **ENTER** .785398163397

round({ $\pi$ , $\sqrt{2}$ ,ln 2},3) **ENTER**  
{3.142 1.414 .693}

round([[ln 5,ln 3][ $\pi$ ,e<sup>1</sup>]],2) **ENTER**  
[[1.61 1.1 ]  
[3.14 2.72]]

**rref**

MATRX OPS menu

**rref** *matrix*

Returns the reduced row-echelon form of a real or complex *matrix*. The number of columns must be greater than or equal to the number of rows.

[[4,5,6][7,8,9]]**MAT** **ENTER**  
[[4 5 6]  
[7 8 9]]

rref **MAT** **ENTER**  
[[1 0 -.999999999999...  
[0 1 2 ...

**rSwap(**

MATRX OPS menu

**rSwap**(*matrix*,*rowA*,*rowB*)

Returns a matrix with *rowA* of a real or complex *matrix* swapped with *rowB*.

[[5,3,1][2,0,4][3,-1,2]]**MAT** **ENTER**  
[[5 3 1]  
[2 0 4]  
[3 -1 2]]

rSwap(**MAT**,2,3) **ENTER**  
[[5 3 1]  
[3 -1 2]  
[2 0 4]]



**Scatter**

† STAT DRAW menu  
(Scatte shows  
on menu)

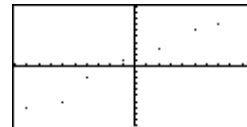
**Scatter**  $xList,yList$ 

Draws a scatter plot on the current graph, using the real data pairs in  $xList$  and  $yList$ .

**Scatter**

Uses the data in built-in variables **xStat** and **yStat**. These variables must contain valid data of the same dimension; otherwise, an error occurs.

```
{-9,-6,-4,-1,2,5,7,10}→XL 
{-9 -6 -4 -1 2 5 7 1...
{-7,-6,-2,1,3,6,7,9}→YL 
{-7 -6 -2 1 3 6 7 9}
ZStd:Scatter XL,YL 
```

**Sci**

† mode screen

**Sci**

Sets scientific notation display mode.

In **Sci** notation mode:

```
123456789  1.23456789E8
```

In **Normal** notation mode:

```
123456789  123456789
```

### Select(

LIST OPS menu

#### Select(*xListName*,*yListName*)

If a scatter plot or xyline plot is currently selected and plotted on the graph screen, you can select a subset (range) of those data points. The selected data points are stored to *xListName* and *yListName*.

**Select**(*xListName*,*yListName*) displays the current graph screen and starts an interactive session during which you select a range of data points.

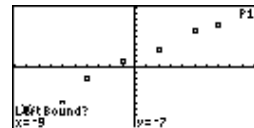
- Move the cursor to the leftmost (left bound) point of the range you want to select and press **ENTER**.
- Then move the cursor to the rightmost (right bound) point of the range you want to select and press **ENTER**.

A new stat plot of *xListName* and *yListName* replaces the plot from which you selected the points.

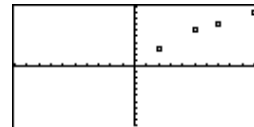
```
{-9,-6,-4,-1,2,5,7,10}→L1 ENTER
{-9 -6 -4 -1 2 5 7 1...
{-7,-6,-2,1,3,6,7,9}→L2 ENTER
{-7 -6 -2 1 3 6 7 9)
Plot1(1,L1,L2):ZStd ENTER
```

After the graph is displayed:

```
Select(L10,L20) ENTER
```



Move the cursor to point (2,3) and press **ENTER**. Then move to (10,9) and press **ENTER**.



```
L10 ENTER {2 5 7 10}
L20 ENTER {3 6 7 9}
```

```
{1,2,3,4,5}→L1:Send(L1) ENTER
```

Done

### Send(

‡ program editor  
I/O menu

#### Send(*listName*)

Sends the contents of *listName* to the CBL or CBR System.

**seq(**

MATH MISC menu

**seq(expression,variable,begin,end,step)**Returns a list containing a sequence of numbers created by evaluating *expression* from *variable = begin* to *variable = end* in increments of *step*.seq(x<sup>2</sup>,x,1,8,2) **ENTER**

{1 9 25 49}

**seq(expression,variable,begin,end)**Uses a *step* of 1.seq(x<sup>2</sup>,x,1,8) **ENTER**

{1 4 9 16 25 36 49 6..}

**SeqG**

† graph format screen

**SeqG**

Sets sequential graphing format, in which selected functions are plotted one at a time.

**SetLEdit**LIST OPS menu  
(SetLE shows on menu)**SetLEdit column1ListName[, ... ,column20ListName]**Removes all lists from the list editor and then stores one or more *ListNames* in the specified order, starting with column 1.{1,2,3,4} → L1 **ENTER**

{1 2 3 4}

{5,6,7,8} → L2 **ENTER**

{5 6 7 8}

SetLEdit L1,L2 **ENTER**

Done

The list editor now contains:

L1	L2	----- 1
1	2	
2	3	
3	4	
4	5	
5	6	
6	7	
7	8	
8		
L1(1) = 1		
← → NAMES " OPS		

**Shade(**

GRAPH DRAW menu

**Shade**(*lowerFunc*,*upperFunc*,*xLeft*,*xRight*,*pattern*,*patternRes*)

Draws *lowerFunc* and *upperFunc* in terms of **x** on the current graph and shades the area bounded by *lowerFunc*, *upperFunc*, *xLeft*, and *xRight*. The shading style is determined by *pattern* (1 through 4) and *patternRes* (1 through 8).

*pattern*:

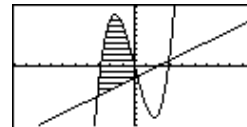
- |                        |                        |
|------------------------|------------------------|
| 1 = vertical (default) | 3 = negative-slope 45° |
| 2 = horizontal         | 4 = positive-slope 45° |

*patternRes* (resolution):

- |                           |                     |
|---------------------------|---------------------|
| 1 = every pixel (default) | 5 = every 5th pixel |
| 2 = every 2nd pixel       | 6 = every 6th pixel |
| 3 = every 3rd pixel       | 7 = every 7th pixel |
| 4 = every 4th pixel       | 8 = every 8th pixel |

**Shade**(*lowerFunc*,*upperFunc*)

Sets *xLeft* and *xRight* to **xMin** and **xMax**, respectively, and uses the defaults for *pattern* and *patternRes*.

In **Func** graphing mode:Shade( $x-2$ , $x^3-8$   $x$ , $-5$ , $1$ , $2$ , $3$ ) **ENTER**C1Drw:Shade( $x^3-8$   $x$ , $x-2$ ) **ENTER**

**shftL**

BASE BIT menu

**shftL** *integer*

Returns a real *integer* with bits shifted one to the left. Internally, *integer* is represented as a 16-bit binary number. When the bits are shifted left, the leftmost bit is dropped and 0 is used as the rightmost bit.

**shftL** 0000111100001111b = 0001111000011110b

**shftL** is not valid in **Dec** number base mode. To enter hexadecimal numbers **A** through **F**, use the BASE A-F menu. Do not use **[ALPHA]** to type a letter.

In **Bin** number base mode:

```
shftL 0000111100001111 [ENTER]
                        1111000011110b
```

Leading zeros are not displayed.

**shftR**

BASE BIT menu

**shftR** *integer*

Returns a real *integer* with bits shifted one to the right. Internally, *integer* is represented as a 16-bit binary number. When the bits are shifted right, the rightmost bit is dropped and 0 is used as the leftmost bit.

**shftR** 0000111100001111b = 0000011110000111b

**shftR** is not valid in **Dec** number base mode. To enter hexadecimal numbers **A** through **F**, use the BASE A-F menu. Do not use **[ALPHA]** to type a letter.

In **Bin** number base mode:

```
shftR 0000111100001111 [ENTER]
                        11110000111b
```

Leading zeros are not displayed.

### ShwSt

CATALOG

### ShwSt

Displays the results of the most recent stat calculation.

### sign

MATH NUM menu

### sign *number* or **sign** (*expression*)

Returns **-1** if the argument is  $< 0$ , **1** if it is  $> 0$ , or **0** if it is  $= 0$ . The argument must be real.

sign -3.2 **ENTER** -1

sign (6+2-8) **ENTER** 0

### sign *list*

Returns a list in which each element is -1, 1, or 0 to indicate the sign of the corresponding element in *list*.

sign {-3.2,16.8,6+2-8} **ENTER** {-1 1 0}

### SimulG

† graph format screen

### SimulG

Sets simultaneous graphing format, in which all selected functions are plotted at the same time.

### simult(

† **2nd** [SIMULT]

### simult(*squareMatrix*,*vector*)

Returns a vector containing the solutions to a system of simultaneous linear equations that have the form:

$$a_{1,1}x_1 + a_{1,2}x_2 + a_{1,3}x_3 + \dots = b_1$$

$$a_{2,1}x_1 + a_{2,2}x_2 + a_{2,3}x_3 + \dots = b_2$$

$$a_{3,1}x_1 + a_{3,2}x_2 + a_{3,3}x_3 + \dots = b_3$$

Each row in *squareMatrix* contains the **a** coefficients of an equation, and *vector* contains the **b** constants.

Solve the following for x and y:

$$\begin{aligned} 3x - 4y &= 7 \\ x + 6y &= 6 \end{aligned}$$

[[3,-4][1,6]]**→MAT** **ENTER** [[3 -4]  
[1 6 ]]

[7,6]**→VEC** **ENTER** [7 6]

simult(MAT,VEC) **ENTER** [3 .5]

The solution is  $x=3$  and  $y=.5$ .

**sin**

[SIN]

**sin** *angle* or **sin** (*expression*)

Returns the sine of *angle* or *expression*, which can be real or complex.

An angle is interpreted as degrees or radians according to the current angle mode. In any angle mode, you can designate an angle as degrees or radians by using the ° or  $\text{r}$  designator, respectively, from the MATH ANGLE menu.

In **Radian** angle mode:

```
sin  $\pi/2$  [ENTER] 0
sin ( $\pi/2$ ) [ENTER] 1
sin 45° [ENTER] .707106781187
```

In **Degree** angle mode:

```
sin 45 [ENTER] .707106781187
sin ( $\pi/2$ )r [ENTER] 1
```

**sin** *list*

Returns a list in which each element is the sine of the corresponding element in *list*.

In **Radian** angle mode:

```
sin {0, $\pi/2$ , $\pi$ } [ENTER] {0 1 0}
```

In **Degree** angle mode:

```
sin {0,30,90} [ENTER] {0 .5 1}
```

**sin** *squareMatrix*

Returns a square matrix that is the matrix sine of *squareMatrix*. The matrix sine corresponds to the result calculated using power series or Cayley-Hamilton Theorem techniques. This is *not* the same as simply calculating the sine of each element.

The squareMatrix cannot have repeated eigenvalues.

**sin<sup>-1</sup>**[2nd] [SIN<sup>-1</sup>]**sin<sup>-1</sup>** *number* or **sin<sup>-1</sup>** (*expression*)

Returns the arcsine of *number* or *expression*, which can be real or complex.

In **Radian** angle mode:

```
sin-1 .5 [ENTER] .523598775598
sin-1 {0,.5} [ENTER] {0 .523598775598}
```

**sin<sup>-1</sup>** *list*

Returns a list in which each element is the arcsine of the corresponding element in *list*.

In **Degree** angle mode:

```
sin-1 1 [ENTER] 90
```

<b>sinh</b> MATH HYP menu	<b>sinh</b> <i>number</i> or <b>sinh</b> ( <i>expression</i> )	sinh 1.2 <input type="button" value="ENTER"/>	1.50946135541
	Returns the hyperbolic sine of <i>number</i> or <i>expression</i> , which can be real or complex.		
	<b>sinh</b> <i>list</i>	sinh {0,1.2} <input type="button" value="ENTER"/>	{0 1.50946135541}
	Returns a list in which each element is the hyperbolic sine of the corresponding element in <i>list</i> .		
<b>sinh<sup>-1</sup></b> MATH HYP menu	<b>sinh<sup>-1</sup></b> <i>number</i> or <b>sinh<sup>-1</sup></b> ( <i>expression</i> )	sinh <sup>-1</sup> 1 <input type="button" value="ENTER"/>	.88137358702
	Returns the inverse hyperbolic sine of <i>number</i> or <i>expression</i> , which can be real or complex.		
	<b>sinh<sup>-1</sup></b> <i>list</i>	sinh <sup>-1</sup> {1,2.1,3} <input type="button" value="ENTER"/>	{.88137358702 1.4874...}
	Returns a list in which each element is the inverse hyperbolic sine of the corresponding element in <i>list</i> .		

---



## SinR

STAT CALC menu

Built-in equation variables such as **y1**, **r1**, and **xt1** are case-sensitive. Do not use **Y1**, **R1**, and **XT1**.

If you specify a period, the TI-86 may find a solution more quickly or it may find a solution when one would not have been found otherwise.

**SinR** [*iterations*,]*xList*,*yList* [,*period*],*equationVariable*

Attempts to fit a sinusoidal regression model ( $y = a \sin(bx + c) + d$ ) to real data pairs in *xList* and *yList*, using an optional estimated *period*. The regression equation is stored to *equationVariable*, which must be a built-in equation variable such as **y1**, **r1**, and **xt1**. The equation's coefficients always are stored as a list to built-in variable **PRegC**.

*iterations* is optional; it specifies the maximum number of times (1 through 16) the TI-86 will attempt to find a solution. If omitted, 8 is used. Typically, larger values result in better accuracy but longer execution times, and vice versa.

If you omit the optional *period*, the difference between values in *xList* should be equal and in sequential order. If you specify *period*, the differences between x values can be unequal.

Values used for *xList* and *yList* are stored automatically to built-in variables **xStat** and **yStat**, respectively. The regression equation is stored also to built-in equation variable **RegEq**.

The output of **SinR** is always in radians, regardless of the angle mode setting.

**SinR** [*iterations*,]*xList*,*yList* [,*period*]

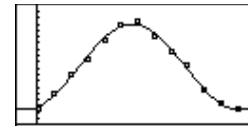
Stores the regression equation to **RegEq** only.

```
seq(x,x,1,361,30)→L1 [ENTER]
{1 31 61 91 121 151 ...
{5.5,8,11,13.5,16.5,19,19.5,17,
14.5,12.5,8.5,6.5,5.5}→L2 [ENTER]
{5.5 8 11 13.5 16.5...
SinR L1,L2,y1 [ENTER]
```

```
SinReg
y=a*sin(bx+c)+d
PRegC=
{6.77022677941 .0162...
```

```
Plot1(1,L1,L2) [ENTER]
ZData [ENTER]
```

Done



**SinR** [*iterations*,] *equationVariable*

Uses **xStat** and **yStat** for *xList* and *yList*, respectively. These built-in variables must contain valid data of the same dimension; otherwise, an error occurs. The regression equation is stored to *equationVariable* and **RegEq**.

**SinR** [*iterations*]

Uses **xStat** and **yStat**, and stores the regression equation to **RegEq** only.

**SlpFld**

† graph format screen  
(scroll down to  
second screen)

**SlpFld**

In **DifEq** graphing mode, turns on slope fields. To turn off direction and slope fields, use **FldOff**.

**Solver(**

† [2nd] [SOLVER]

**Solver**(*equation,variable,guess,{lower,upper}*)

Solves *equation* for *variable*, given an initial *guess* and *lower* and *upper* bounds within which the solution is sought. *equation* can be an expression, which is assumed to equal 0.

If  $y=5$ , solve  $x^3+y^2=125$  for  $x$ . You guess the solution is approximately 4:

```
5>y [ENTER] 5
Solver(x^3+y^2=125,x,4) [ENTER] Done
x [ENTER] 4.64158883361
```

**Solver**(*equation,variable,guess*)

Uses  $-1E99$  and  $1E99$  for *upper* and *lower*, respectively.

**Solver**(*equation,variable,{guessLower,guessUpper}*)

Uses the secant line between *guessLower* and *guessUpper* to start the search. **Solver(** will still search for a solution outside of this range.

<b>sortA</b> LIST OPS menu	<b>SortA</b> <i>list</i> Returns a list in which the real or complex elements of <i>list</i> are sorted in ascending order.	$\{5,8,-4,0,-6\} \rightarrow L1$ <input type="button" value="ENTER"/> SortA L1 <input type="button" value="ENTER"/>	$\begin{pmatrix} 5 & 8 & -4 & 0 & -6 \\ -6 & -4 & 0 & 5 & 8 \end{pmatrix}$
<b>sortD</b> LIST OPS menu	<b>SortD</b> <i>list</i> Returns a list in which the real or complex elements of <i>list</i> are sorted in descending order.	$\{5,8,-4,0,-6\} \rightarrow L1$ <input type="button" value="ENTER"/> SortD L1 <input type="button" value="ENTER"/>	$\begin{pmatrix} 5 & 8 & -4 & 0 & -6 \\ 8 & 5 & 0 & -4 & -6 \end{pmatrix}$
<b>Sortx</b> LIST OPS menu	<b>Sortx</b> <i>xListName,yListName,frequencyListName</i> <b>Sortx</b> <i>xListName,yListName</i> In ascending order of <b>x</b> elements, sorts real or complex <b>x</b> and <b>y</b> data pairs and, optionally, their frequencies in <i>xListName</i> , <i>yListName</i> , and <i>frequencyListName</i> . The lists' contents are updated to reflect the changes.	$\{3,1,2\} \rightarrow XL$ <input type="button" value="ENTER"/> $\{0,8,-4\} \rightarrow YL$ <input type="button" value="ENTER"/> Sortx XL,YL <input type="button" value="ENTER"/> XL <input type="button" value="ENTER"/> YL <input type="button" value="ENTER"/>	$\begin{pmatrix} 3 & 1 & 2 \\ 0 & 8 & -4 \end{pmatrix}$ Done $\begin{pmatrix} 1 & 2 & 3 \\ 8 & -4 & 0 \end{pmatrix}$
	<b>Sortx</b> Uses built-in variables <b>xStat</b> and <b>yStat</b> for <i>xListName</i> and <i>yListName</i> , respectively. These built-in variables must contain valid data of the same dimension; otherwise, an error occurs.		
<b>Sorty</b> LIST OPS menu	<b>Sorty</b> <i>xListName,yListName,frequencyListName</i> <b>Sorty</b> <i>xListName,yListName</i> In ascending order of <b>y</b> elements, sorts real or complex <b>x</b> and <b>y</b> data pairs and, optionally, their frequencies in <i>xListName</i> , <i>yListName</i> , and <i>frequencyListName</i> . The lists' contents are updated to reflect the changes.	$\{3,1,2\} \rightarrow XL$ <input type="button" value="ENTER"/> $\{0,8,-4\} \rightarrow YL$ <input type="button" value="ENTER"/> Sorty XL,YL <input type="button" value="ENTER"/> YL <input type="button" value="ENTER"/> XL <input type="button" value="ENTER"/>	$\begin{pmatrix} 3 & 1 & 2 \\ 0 & 8 & -4 \end{pmatrix}$ Done $\begin{pmatrix} -4 & 0 & 8 \\ 2 & 3 & 1 \end{pmatrix}$

### Sorty

Uses built-in variables **xStat** and **yStat** for *xListName* and *yListName*, respectively. These built-in variables must contain valid data of the same dimension; otherwise, an error occurs.

### ►Sph

VECTR OPS menu

#### *vector*►Sph

Displays a 2- or 3-element *vector* as spherical coordinates in  $[r \angle \theta \angle 0]$  or  $[r \angle \theta \angle \phi]$  form, respectively, even if the display mode is not set for spherical (**SphereV**).

In **RectV** vector coordinate mode:

$[0, -1]$ ►Sph  $\boxed{\text{ENTER}}$   
 $[1 \angle -1.57079632679 \angle 1 \dots]$

$[0, 0, -1]$ ►Sph  $\boxed{\text{ENTER}}$   
 $[1 \angle 0 \angle 3.14159265359]$

### SphereV

†  $\boxed{2\text{nd}}$   $\boxed{\text{MODE}}$

#### **SphereV**

Sets spherical vector coordinate mode  $[r \angle \theta \angle \phi]$ .

In **SphereV** vector coordinate mode:

$[1, 2]$   $\boxed{\text{ENTER}}$   
 $[2.2360679775 \angle 1.1071 \dots]$

### Square: $^2$

$\boxed{x^2}$

*number* $^2$  or (*expression*) $^2$

*list* $^2$

*squareMatrix* $^2$

Returns a real or complex argument multiplied by itself. To square a negative number, enclose it in parentheses.

A *squareMatrix* multiplied by itself is not the same as simply squaring each element.

$25^2$   $\boxed{\text{ENTER}}$  625

$(16+9)^2$   $\boxed{\text{ENTER}}$  625

$-2^2$   $\boxed{\text{ENTER}}$  -4

$(-2)^2$   $\boxed{\text{ENTER}}$  4

$\{-2, 4, 25\}^2$   $\boxed{\text{ENTER}}$   $\{4 \ 16 \ 625\}$

$[[2, 3][4, 5]]^2$   $\boxed{\text{ENTER}}$   $[[16 \ 21]$

$[28 \ 37]]$

### Square root: $\sqrt{\quad}$

$\boxed{2\text{nd}}$   $\boxed{\sqrt{\quad}}$

$\sqrt{\textit{number}}$  or  $\sqrt{\textit{expression}}$

Returns the square root of *number* or *expression*, which can be real or complex.

$\sqrt{25}$   $\boxed{\text{ENTER}}$  5

$\sqrt{(25+11)}$   $\boxed{\text{ENTER}}$  6

$\sqrt{\textit{list}}$

Returns a list in which element is the square root of the corresponding element in *list*.

In **RectC** complex number mode:

$\sqrt{\{-2,25\}}$  **ENTER**  $\{(0,1.41421356237) (\dots)$

## StEq(

STRNG menu

**StEq**(*stringVariable*,*equationVariable*)

Converts *stringVariable* to a number, expression, or equation, and stores it in *equationVariable*.

To convert the string and retain the same variable name, you can set *equationVariable* equal to *stringVariable*.

*If you use **Input** instead of **InpSt** here, the entered expression is evaluated at the current value of x and the result (not the expression) is stored.*

"5"→x:6 x **ENTER**

ERROR 10 DATA TYPE

"5"→x:StEq(x,x):6 x **ENTER** 30

Program segment:

```

:
:
:InpSt "Enter y1(x):",STR
:StEq(STR,y1)
:Input "Enter x:",x
:Disp "Result is:",y1(x)
:

```

You cannot store a string directly to a built-in equation variable.

## StGDB

† GRAPH menu

**StGDB** *graphDataBaseName*

Creates a graph database (GDB) variable that contains the current:

- Graphing mode, graph format settings, and range variables.
- Functions in the equation editor, whether they are selected, and their graph styles.

To restore the database and recreate the graph, use **RcGDB** (page 343).

## Stop

‡ program editor  
CTL menu

### Stop

Ends program execution and returns to the home screen.

Use  $N=999$ ,  
not  $N=999$ .

Program segment:

```

:
:Input N
:If N==999
:Stop
:

```

Store to variable: →

**STO**▶

*number* → *variable* or (*expression*) → *variable*  
*string* → *variable*  
*list* → *variable*  
*vector* → *variable*  
*matrix* → *variable*

Stores the specified argument to *variable*.

```

10→A:4*A [ENTER] 40
"Hello"→STR [ENTER] Hello
{1,2,3}→L1 [ENTER] {1 2 3}
[1,2,3]→VEC [ENTER] [1 2 3]
[[1,2,3][4,5,6]]→MAT [ENTER]
[[1 2 3]
[4 5 6]]

```

## StPic

‡ GRAPH menu

### StPic *pictureName*

Stores a picture of the current graph screen to *pictureName*.

## StReg(

STAT CALC menu

### StReg(*variable*)

Stores the most recently calculated regression equation to *variable*. This lets you save a regression equation by storing it to a variable as opposed to a built-in equation variable.

**2nd** [RCL] EQ [ENTER] recalls the equation. Then [ENTER] evaluates it at the current value of  $x$ .

```

{1,2,3,4,5}→L1 [ENTER] {1 2 3 4 5}
{1,20,55,230,742}→L2 [ENTER] {1 20 55 230 742}
ExpR L1,L2:StReg(EQ) [ENTER] Done
8→x [ENTER] 8
Rcl EQ [ENTER]
.41138948780597*4.7879605684671^x
[ENTER] 113620.765451

```

## String entry: "

STRNG menu  
 ‡ program editor  
 I/O menu

**"string"**

Defines a string. When you display a string, it is left-justified on the screen.

Strings are interpreted as text characters, not numbers. For example, you cannot perform a calculation with strings such as "4" or "A\*8". To convert between string variables and equation variables, use **EqSt(** and **StEq(** as described on pages 290 and 361, respectively.

**"Hello"→STR** 

Disp STR+", Jan"  Hello  
 Hello, Jan Done

**sub(**

STRNG menu

**sub(string,begin,length)**

Returns a new string that is a subset of *string*, starting at character number *begin* and continuing for the specified *length*.

**"The answer is:"→STR** 

sub(STR,5,6)  The answer is:  
 answer

## Subtraction: −

*numberA* − *numberB*

Returns the value of *numberB* subtracted from *numberA*. The arguments can be real or complex.

**6−2** 4  
**10−−4.5**  14.5*list* − *number*

Returns a list in which *number* is subtracted from each element of *list*. The arguments can be real or complex.

**{10,9,8}−4** 

{6 5 4}  
 In **RectC** complex number mode:  
**{8,1,(5,2)}−3**   
 {(5,0) (−2,0) (2,2)}

<i>listA</i> - <i>listB</i>	$\{5,7,9\} - \{4,5,6\}$ <b>ENTER</b>	$\{1\ 2\ 3\}$
<i>matrixA</i> - <i>matrixB</i>	$[[5,7,9][11,13,15]] - [[4,5,6][7,8,9]]$ <b>ENTER</b>	$[[1\ 2\ 3][4\ 5\ 6]]$
<i>vectorA</i> - <i>vectorB</i>	$[5,7,9] - [1,2,3]$ <b>ENTER</b>	$[4\ 5\ 6]$

Returns a list, matrix, or vector that is the result of each element in the second argument subtracted from the corresponding element in the first argument. The two real or complex arguments must have the same dimension.

### sum

MATH MISC menu  
LIST OPS menu

<b>sum</b> <i>list</i>	$\text{sum } \{1,2,4,8\}$ <b>ENTER</b>	15
Returns the sum of all real or complex elements in <i>list</i> .	$\text{sum } \{2,7,-8,0\}$ <b>ENTER</b>	1

### tan

**TAN**

<b>tan</b> <i>angle</i> or <b>tan</b> ( <i>expression</i> )	In <b>Radian</b> angle mode:	
Returns the tangent of <i>angle</i> or <i>expression</i> , which can be real or complex.	$\tan \pi/4$ <b>ENTER</b>	0
	$\tan (\pi/4)$ <b>ENTER</b>	1
	$\tan 45^\circ$ <b>ENTER</b>	1
An angle is interpreted as degrees or radians according to the current angle mode. In any angle mode, you can designate an angle as degrees or radians by using the $^\circ$ or $^r$ designator, respectively, from the MATH ANGLE menu.	In <b>Degree</b> angle mode:	
	$\tan 45$ <b>ENTER</b>	1
	$\tan (\pi/4)^r$ <b>ENTER</b>	1
<b>tan</b> <i>list</i>	In <b>Degree</b> angle mode:	
Returns a list in which each element is the tangent of the corresponding element in <i>list</i> .	$\tan \{0,45,60\}$ <b>ENTER</b>	$\{0\ 1\ 1.73205080757\}$



<b>tan<sup>-1</sup></b> [2nd] [TAN <sup>-1</sup> ]	<b>tan<sup>-1</sup> number</b> or <b>tan<sup>-1</sup> (expression)</b>	In <b>Radian</b> angle mode: tan <sup>-1</sup> .5 [ENTER] .463647609001
	Returns the arctangent of <i>number</i> or <i>expression</i> , which can be real or complex.	In <b>Degree</b> angle mode: tan <sup>-1</sup> 1 [ENTER] 45
	<b>tan<sup>-1</sup> list</b>	In <b>Radian</b> angle mode: tan <sup>-1</sup> {0, .2, .5} [ENTER] {0 .19739555985 .463...
<b>tanh</b> MATH HYP menu	<b>tanh number</b> or <b>tanh (expression)</b>	tanh 1.2 [ENTER] .833654607012
	Returns the hyperbolic tangent of <i>number</i> or <i>expression</i> , which can be real or complex.	tanh {0,1.2} [ENTER] {0 .833654607012}
	<b>tanh list</b>	
<b>tanh<sup>-1</sup></b> MATH HYP menu	<b>tanh<sup>-1</sup> number</b> or <b>tanh<sup>-1</sup>(expression)</b>	tanh <sup>-1</sup> 0 [ENTER] 0
	Returns the inverse hyperbolic tangent of <i>number</i> or <i>expression</i> , which can be real or complex.	In <b>RectC</b> complex number mode: tanh <sup>-1</sup> {0,2.1} [ENTER] {(0,0) (.51804596584...
	<b>tanh<sup>-1</sup> list</b>	
	Returns a list in which each element is the inverse hyperbolic tangent of the corresponding element in <i>list</i> .	

### TanLn(

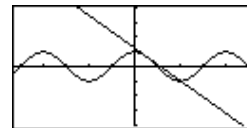
GRAPH DRAW menu

#### TanLn(*expression*,*xValue*)

Draws *expression* on the current graph and then draws a tangent line at *xValue*.

In **Func** graphing mode and **Radian** angle mode:

ZTrig: TanLn(cos x,  $\pi/4$ ) **[ENTER]**



### Text(

† GRAPH DRAW menu

#### Text(*row*,*column*,*string*)

Writes a text *string* on the current graph beginning at pixel (*row*,*column*), where  $0 \leq \text{row} \leq 57$  and  $0 \leq \text{column} \leq 123$ .

Text at the bottom of the graph may be covered by a displayed menu. To remove the menu, press **[CLEAR]**.

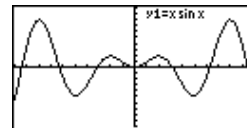
Program segment in **Func** graphing mode and a **ZStd** graph screen:

```

:
:y1=x sin x
:Text(0,70,"y1=x sin x")
:

```

When executed:



### Then

‡ program editor  
CTL menu

Refer to syntax information for **If**, beginning on page 305. See the **If:Then:End** and **If:Then:Else:End** syntax.



**TwoVar**

STAT CALC menu  
(TwoVa shows on menu)

**TwoVar**  $xList, yList, frequencyList$ 

Performs two-variable statistical analysis on the real data pairs in  $xList$  and  $yList$ , using the frequencies in  $frequencyList$ .

Values used for  $xList$ ,  $yList$ , and  $frequencyList$  are stored automatically to the built-in variables **xStat**, **yStat**, and **fStat**, respectively.

**TwoVar**  $xList, yList$ 

Uses frequencies of 1.

**TwoVar**

Uses **xStat**, **yStat**, and **fStat** for  $xList$ ,  $yList$ , and  $frequencyList$ . These built-in variables must contain valid data of the same dimension; otherwise, an error occurs.

```
{0,1,2,3,4,5,6}→L1 [ENTER]
{0 1 2 3 4 5 6}
{0,1,2,3,4,5,6}→L2 [ENTER]
{0 1 2 3 4 5 6}
TwoVar L1,L2 [ENTER]
```

```
2-Var Stats
Mx=3
My=2.1
Sx=2.91
Sy=2.1602469
σx=2
σy=2
n=7
```

Scroll down to see more results.

**unitV**

VECTR MATH menu

**unitV**  $vector$ 

Returns a unit vector of a real or complex  $vector$ , where:

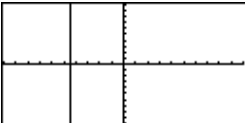
**unitV**  $[a,b,c]$  returns  $\left[ \frac{a}{\text{norm}} \frac{b}{\text{norm}} \frac{c}{\text{norm}} \right]$

and

**norm** is  $\sqrt{a^2+b^2+c^2}$ .

In **RectV** vector coordinate mode:

```
unitV [1,2,1] [ENTER]
[.408248290464 .8164...
```

<b>vc&gt;li</b> LIST OPS menu VECTR OPS menu	<b>vc&gt;li</b> <i>vector</i> Returns a real or complex <i>vector</i> converted to a list.	$vc>li [2,7,-8,0]$ $\boxed{\text{ENTER}}$ {2 7 -8 0} $(vc>li [2,7,-8,0])^2$ $\boxed{\text{ENTER}}$ {4 49 64 0}
<b>Vector entry: [ ]</b> $\boxed{2nd}$ [ ] and $\boxed{2nd}$ [1]	$[element1,element2,...]$ Defines a vector in which each element is a real or complex number or variable.	$[4,5,6]$ $\rightarrow$ VEC $\boxed{\text{ENTER}}$ [4 5 6] In <b>PolarC</b> complex number mode: $[5,(2\angle\pi/4)]$ $\rightarrow$ VEC $\boxed{\text{ENTER}}$ $[(5\angle 0) (2\angle .785398163...)]$
<b>Vert</b> † GRAPH DRAW menu	<b>Vert</b> <i>xValue</i> Draws a vertical line on the current graph at <i>xValue</i> .	In a <b>ZStd</b> graph screen: Vert -4.5 $\boxed{\text{ENTER}}$ 
<b>While</b> † program editor CTL menu	<b>:While</b> <i>condition</i> <i>:commands-while-true</i> <b>:End</b> <i>:command</i> Executes <i>commands-while-true</i> as long as <i>condition</i> is true.	Program segment: : :1 $\rightarrow$ J :0 $\rightarrow$ TEMP :While J $\leq$ 20 : TEMP+1/J $\rightarrow$ TEMP : J+1 $\rightarrow$ J :End :Disp "Reciprocal sums to 20",TEMP : :

**xor**

BASE BOOL menu

*integerA xor integerB*

Compares two real integers bit by bit. Internally, both integers are converted to binary. When corresponding bits are compared, the result is 1 if either bit (but not both) is 1; the result is 0 if both bits are 0 or both bits are 1. The returned value is the sum of the bit results.

For example,  $78 \text{ xor } 23 = 89$ .

$$\begin{array}{r} 78 = 1001110\text{b} \\ 23 = 0010111\text{b} \\ \hline 1011001\text{b} = 89 \end{array}$$

You can enter real numbers instead of integers, but they are truncated automatically before the comparison.

In **Dec** number base mode:

78 xor 23  89

In **Bin** number base mode:

1001110 xor 10111  1011001b

Ans▶Dec  89d

**xyline**

† STAT DRAW menu

**xyline** *xList,yList*

Draws a line plot on the current graph, using the real data pairs in *xList* and *yList*.

**xyline**

Uses the data in built-in variables **xStat** and **yStat**. These variables must contain valid data of the same dimension; otherwise, an error occurs.

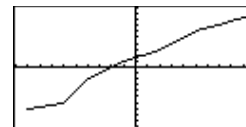
{-9,-6,-4,-1,2,5,7,10}▶XL

{-9 -6 -4 -1 2 5 7 1...

{-7,-6,-2,1,3,6,7,9}▶YL

{-7 -6 -2 1 3 6 7 9}

ZStd:xyline XL,YL



**ZData**

† GRAPH ZOOM menu

**ZData**

Adjusts the window variable values based on the currently defined statistical plots so that all stat data points will be plotted, and then updates the graph screen.

In **Func** graphing mode:{1,2,3,4}→XL **[ENTER]**

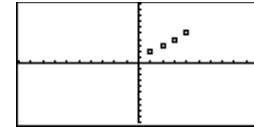
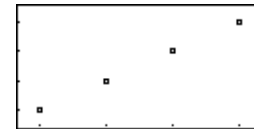
{1 2 3 4}

{2,3,4,5}→YL **[ENTER]**

{2 3 4 5}

Plot1(1,XL,YL) **[ENTER]**

Done

ZStd **[ENTER]**ZData **[ENTER]**

**ZDecm**

† GRAPH ZOOM menu

**ZDecm**

Sets the window variable values such that  $\Delta x = \Delta y = .1$ , and then updates the graph screen with the origin centered on the screen.

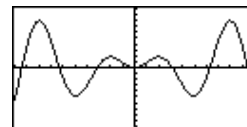
**xMin=-6.3**    **yMin=-3.1**  
**xMax=6.3**    **yMax=3.1**  
**xScl=1**        **yScl=1**

One of the benefits of **ZDecm** is that you can trace in .1 increments.

In **Func** graphing mode:

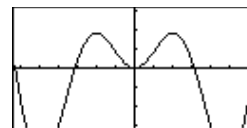
$y1=x \sin x$  [ENTER]  
 ZStd [ENTER]

Done



If you trace the graph above, **x** values start at 0 and increment by .1587301587.

ZDecm [ENTER]



If you trace this graph, the **x** values increment by .1.



**ZFit**

† GRAPH ZOOM menu

**ZFit**

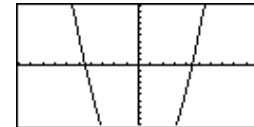
Recalculates **yMin** and **yMax** to include the minimum and maximum **y** values of the selected functions between the current **xMin** and **xMax**, and then updates the graph screen.

This does not affect **xMin** and **xMax**.

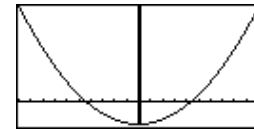
In **Func** graphing mode: $y1=x^2-20$  [ENTER]

ZStd [ENTER]

Done



ZFit [ENTER]

**ZIn**

† GRAPH ZOOM menu

**ZIn**

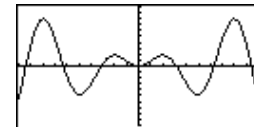
Zooms in on the part of the graph centered around the current cursor location.

Zoom factors are set by the values of built-in variables **xFact** and **yFact**; the default is 4 for both factors.

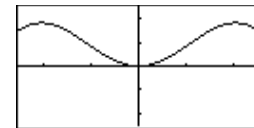
In **Func** graphing mode: $y1=x \sin x$  [ENTER]

ZStd [ENTER]

Done



ZIn [ENTER]



**ZInt**

† GRAPH ZOOM menu

**ZInt**

Sets the window variable values so that each pixel is an integer in all directions ( $\Delta x = \Delta y = 1$ ), sets **xScl=yScl=10**, and then updates the graph screen.

The current cursor location becomes the center of the new graph.

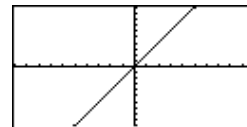
One of the benefits of **ZInt** is that you can trace in whole number increments.

In **Func** graphing mode:

$y1 = \text{der1}(x^2 - 20, x)$

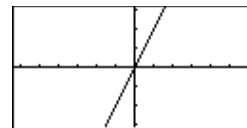
ZStd

Done



If you trace the graph above, **x** values start at 0 and increment by .1587301587.

ZInt



If you trace this graph, **x** values increment by 1.

**ZOut**

† GRAPH ZOOM menu

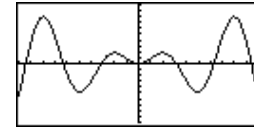
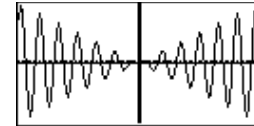
**ZOut**

Zooms out to display more of the graph, centered around the current cursor location.

Zoom factors are set by the values of built-in variables **xFact** and **yFact**; the default is 4 for both factors.

In **Func** graphing mode:y1=x sin x **ENTER**ZStd **ENTER**

Done

ZOut **ENTER****ZPrev**

† GRAPH ZOOM menu

**ZPrev**

Replots the graph using the window variable values of the graph that was displayed before you executed the previous **ZOOM** instruction.

**ZRcl**

† GRAPH ZOOM menu

**ZRcl**

Sets the window variables to values stored previously in the user-defined zoom-window variables, and then updates the graph screen.

To set user-defined zoom-window variables, either:

- Press **[GRAPH] [F3] [MORE] [MORE] [MORE] [F1] (ZSTO)** to store the current graph's window variables.  
– or –
- Store the applicable values to the zoom-window variables, whose names begin with **z** followed by the regular window variable name. For example, store a value for **xMin** to **zxMin**, **yMin** to **zyMin**, etc.

**ZSqr**

† GRAPH ZOOM menu

**ZSqr**

Sets the window variable values to produce “square” pixels where  $\Delta x = \Delta y$ , and then updates the graph screen.

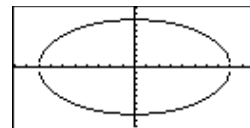
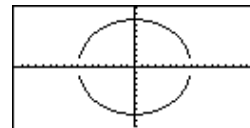
The center of the current graph (not necessarily the axes intersection) becomes the center of the new graph.

In other types of zooms, squares may look like rectangles and circles may look like ovals. Use **ZSqr** for a more accurate shape.

In Func graphing mode:

 $y1 = \sqrt{8^2 - x^2}; y2 = -y1$  **[ENTER]**  
**ZStd** **[ENTER]**

Done

**ZSqr** **[ENTER]**

**ZStd**

† GRAPH ZOOM menu

**ZStd**

Sets the window variables to the standard default values, and then updates the graph screen.

**Func** graphing mode:

**xMin=-10**      **yMin=-10**  
**xMax=10**      **yMax=10**  
**xScl=1**        **yScl=1**

**Pol** graphing mode:

**θMin=0**                      **xMin=-10**   **yMin=-10**  
**θMax=6.28318530718 (2π)**   **xMax=10**   **yMax=10**  
**θStep=.130899693899... (π/24)**   **xScl=1**   **yScl=1**

**Param** graphing mode:

**tMin=0**                      **xMin=-10**   **yMin=-10**  
**tMax=6.28318530718 (2π)**   **xMax=10**   **yMax=10**  
**tStep=.130899693899... (π/24)**   **xScl=1**   **yScl=1**

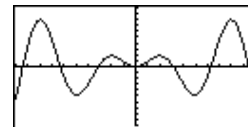
**DifEq** graphing mode:

**tMin=0**                      **xMin=-10**   **yMin=-10**  
**tMax=6.28318530718 (2π)**   **xMax=10**   **yMax=10**  
**tStep=.130899693899... (π/24)**   **xScl=1**   **yScl=1**  
**tPlot=0**                      **difTol=.001**

In **Func** graphing mode:

**y1=x sin x** [ENTER]  
**ZStd** [ENTER]

Done



## ZTrig

† GRAPH ZOOM menu

### ZTrig

Sets the window variables to preset values appropriate for plotting trig functions in **Radian** angle mode ( $\Delta x = \pi/24$ ), and then updates the graph screen.

**xMin**=-8.24668071567

**xMax**=8.24668071567

**xScl**=1.5707963267949 ( $\pi/2$ )

**yMin**=-4

**yMax**=4

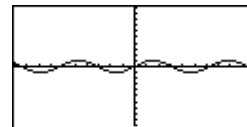
**yScl**=1

In **Func** graphing mode:

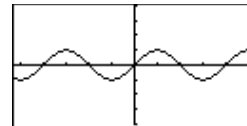
y1=sin x **ENTER**

ZStd **ENTER**

Done

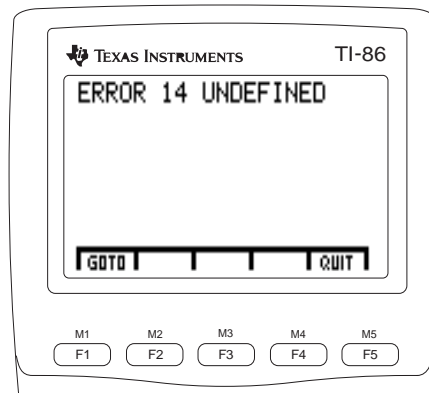


ZTrig **ENTER**



# A Appendix

TI-86 Menu Map.....	380
Handling a Difficulty .....	392
Error Conditions.....	393
Equation Operating System (EOS™) .....	397
TOL (The Tolerance Editor) [2nd] [MEM] [F4].....	398
Computational Accuracy.....	399
Support and Service Information .....	400
Warranty Information .....	402



## TI-86 Menu Map

This section presents the TI-86 menus as they appear on the TI-86 keyboard, starting at the top. If a menu has items that display other menus, the other menus follow directly below the main menu. In the program editor, the appearance of some menus changes slightly. The menu map omits user-created-name menus, such as the LIST NAMES and CONS USER menus.

### LINK Menu $\boxed{2\text{nd}} \boxed{[\text{LINK}]}$

SEND	RECV	SND85		
------	------	-------	--	--

*The link menus are not available in the program editor.*

### LINK SEND Menu $\boxed{2\text{nd}} \boxed{[\text{LINK}]} \boxed{[\text{F1}]}$

BCKUP	PRGM	MATRX	GDB	ALL	▶	LIST	VECTR	REAL	CPLX	EQU	▶	CONS	PIC	WIND	STRNG	
-------	------	-------	-----	-----	---	------	-------	------	------	-----	---	------	-----	------	-------	--

### SEND BCKUP Menu $\boxed{2\text{nd}} \boxed{[\text{LINK}]} \boxed{[\text{F1}]} \boxed{[\text{F1}]}$

XMIT				
------	--	--	--	--

### LINK SEND Selection Screen Menu $\boxed{2\text{nd}} \boxed{[\text{LINK}]} \boxed{[\text{F1}]}$ *data type*

XMIT	SELCT	ALL+	ALL-	
------	-------	------	------	--

### LINK SND85 Menu $\boxed{2\text{nd}} \boxed{[\text{LINK}]} \boxed{[\text{F3}]}$

MATRX	LIST	VECTR	REAL	CPLX	▶	CONS	PIC	STRNG		
-------	------	-------	------	------	---	------	-----	-------	--	--

*In the program editor, DrEqu is available as a GRAPH menu item.*

### GRAPH Menu $\boxed{[\text{GRAPH}]}$ in Func graphing mode

y(x)=	WIND	ZOOM	TRACE	GRAPH	▶	MATH	DRAW	FORMT	STGDB	RCGDB	▶	EVAL	STPIC	RCPIC		
-------	------	------	-------	-------	---	------	------	-------	-------	-------	---	------	-------	-------	--	--



**GRAPH Menu**  $\boxed{\text{GRAPH}}$  in Pol graphing mode

r( $\theta$ )=	WIND	ZOOM	TRACE	GRAPH	▶	MATH	DRAW	FORMT	STGDB	RCGDB	▶	EVAL	STPIC	RCPIC		
----------------	------	------	-------	-------	---	------	------	-------	-------	-------	---	------	-------	-------	--	--

**GRAPH Menu**  $\boxed{\text{GRAPH}}$  in Param graphing mode

E(t)=	WIND	ZOOM	TRACE	GRAPH	▶	MATH	DRAW	FORMT	STGDB	RCGDB	▶	EVAL	STPIC	RCPIC		
-------	------	------	-------	-------	---	------	------	-------	-------	-------	---	------	-------	-------	--	--

**GRAPH Menu**  $\boxed{\text{GRAPH}}$  in DifEq graphing mode

Q'(t)=	WIND	INITC	AXES	GRAPH	▶	FORMT	DRAW	ZOOM	TRACE	EXPLR	▶	EVAL	STGDB	RCGDB	STPIC	RCPIC
--------	------	-------	------	-------	---	-------	------	------	-------	-------	---	------	-------	-------	-------	-------

**Equation Editor Menu**  $\boxed{\text{GRAPH}}$   $\boxed{\text{F1}}$  in Func graphing mode

y(x)=	WIND	ZOOM	TRACE	GRAPH	▶	ALL+	ALL-	STYLE		
x	y	INSf	DELf	SELCT						

**Equation Editor Menu**  $\boxed{\text{GRAPH}}$   $\boxed{\text{F1}}$  in Pol graphing mode

r( $\theta$ )=	WIND	ZOOM	TRACE	GRAPH	▶	ALL+	ALL-	STYLE		
$\theta$	r	INSf	DELf	SELCT						

**Equation Editor Menu**  $\boxed{\text{GRAPH}}$   $\boxed{\text{F1}}$  in Param graphing mode

E(t)=	WIND	ZOOM	TRACE	GRAPH	▶	INSf	ALL+	ALL-	STYLE	
t	xt	yt	DELf	SELCT						

**Equation Editor Menu**  $\boxed{\text{GRAPH}}$   $\boxed{\text{F1}}$  in DifEq graphing mode

Q'(t)=	WIND	INITC	AXES	GRAPH	▶	ALL+	ALL-	STYLE		
t	Q	INSf	DELf	SELCT						



**GRAPH MATH Menu** [GRAPH] [MORE] [F1] in Param graphing mode

MATH	DRAW	FORMT	STGDB	RCGDB				
DIST	dy/dx	dy/dt	dx/dt	ARC	▶	TANLN		

**GRAPH DRAW Menu** [GRAPH] [MORE] [F2]

MATH	DRAW	FORMT	STGDB	RCGDB				
Shade	LINE	VERT	HORIZ	CIRCL	▶	DrawF	PEN	PTON
						PTOFF	PTCHG	▶
						CLDRW	PxOn	PxOff
						PxChg	PxTest	
						▶	TEXT	TanLn
							DrInV	

DrInV is available only in  
Func graphing mode.

DrEq is available only in  
DifEq graphing mode.

**SOLVER Menu** [2nd] [SOLVER] *equation* [ENTER]

GRAPH	WIND	ZOMM	TRACE	SOLVE
-------	------	------	-------	-------

**SOLVER ZOOM Menu** [2nd] [SOLVER] *equation* [ENTER] [F3]

BOX	ZINT	ZOUT	ZFACT	ZSTD
-----	------	------	-------	------

**TABLE Menu** [TABLE]

TABLE	TBLST			
-------	-------	--	--	--

**TABLE SETUP Menu** [TABLE] [F2]

TABLE				
-------	--	--	--	--

**Table Screen Menu** [TABLE] [F1]  
in Func graphing mode

TBLST	SELCT	x	y	
-------	-------	---	---	--

## in Param graphing mode

TBLST	SELCT	t	xt	yt
-------	-------	---	----	----

## in Pol graphing mode

TBLST	SELCT	$\theta$	r	
-------	-------	----------	---	--

## in DifEq graphing mode

TBLST	SELCT	t	Q	
-------	-------	---	---	--

**SIMULT ENTRY Menu** [2nd] [SIMULT] (*integer*  $\geq 2$  &  $\leq 30$ ) [ENTER]

PREV	NEXT	CLRq		SOLVE
------	------	------	--	-------

**SIMULT RESULT Menu** [F5]

COEFS	STOa	STOb	STOx	
-------	------	------	------	--

**PRGM Menu** [PRGM]

NAMES	EDIT			
-------	------	--	--	--

**Program Editor Menu** [PRGM] [F2] *program name* [ENTER]

PAGE↓	PAGE↑	I/O	CTL	INSc	▶	DELc	UNDEL	:		
-------	-------	-----	-----	------	---	------	-------	---	--	--

**PRGM I/O (Input/Output) Menu** [PRGM] [F2] *program name* [ENTER] [F3]

PAGE↓	PAGE↑	I/O	CTL	INSc	▶	CITbl	Get	Send	getKy	CILCD	▶	"	Outpt	InpSt		
Input	Prompt	Disp	DispG	DispT	▶						▶					

**PRGM CTL (Control) Menu** [PRGM] [F2] *programName* [ENTER] [F4]

PAGE↓	PAGE↑	I/O	CTL	INSc	▶	While	Repea	Menu	Lbl	Goto	▶	IS>	DS<	Pause	Retur	Stop	
If	Then	Else	For	End	▶						▶						
												▶	DelVa	GrStl	LCust		

**POLY ENTRY Menu** [2nd] [POLY] (*integer ≥ 2 & ≤ 30*) [ENTER]

CLRq				SOLVE
------	--	--	--	-------

**POLY RESULT Menu** [F5]

COEFS	STOa			
-------	------	--	--	--

**CUSTOM Menu** [CUSTOM]

					▶						▶					
--	--	--	--	--	---	--	--	--	--	--	---	--	--	--	--	--

Use the CUSTOM menu to create your own menu (Chapter 2).

**CATLG-VARS Menu** [2nd] [CATLG-VARS]

CATLG	ALL	REAL	CPLX	LIST	▶	VECTR	MATRX	STRNG	EQU	CONS	▶	PRGM	GDB	PIC	STAT	WIND
-------	-----	------	------	------	---	-------	-------	-------	-----	------	---	------	-----	-----	------	------

**CATLG-VARS Selection Menu** [2nd] [CATLG-VARS] [F1] **or select a data type**

PAGE↓	PAGE↑	CUSTM	BLANK	
-------	-------	-------	-------	--

**CALC Menu** [2nd] [CALC]

evalF	nDer	der1	der2	fnInt	▶	fMin	fMax	arc		
-------	------	------	------	-------	---	------	------	-----	--	--

**MATRIX Menu** [2nd] [MATRX]

NAMES	EDIT	MATH	OPS	CPLX
-------	------	------	-----	------

**Matrix Editor Menu** [2nd] [MATRX] [F2] *matrixName* [ENTER]

INSr	DELr	INSc	DELc	▶REAL
------	------	------	------	-------

**MATRIX MATH Menu** [2nd] [MATRX] [F3]

NAMES	EDIT	MATH	OPS	CPLX
det	τ	norm	eigVl	eigVc

▶	rnorm	cnorm	LU	cond	
---	-------	-------	----	------	--

**MATRIX OPS (Operations) Menu** [2nd] [MATRX] [F4]

NAMES	EDIT	MATH	OPS	CPLX
dim	Fill	ident	ref	rref

▶	aug	rSwap	rAdd	multR	mRAdd	▶	randM				
---	-----	-------	------	-------	-------	---	-------	--	--	--	--

**MATRIX CPLX Menu** [2nd] [MATRX] [F5]

NAMES	EDIT	MATH	OPS	CPLX
conj	real	imag	abs	angle

**VECTR Menu** [2nd] [VECTR]

NAMES	EDIT	MATH	OPS	CPLX
-------	------	------	-----	------

**Vector Editor Menu** [2nd] [VECTR] [F2] *vectorName* [ENTER]

INSi	DELi	▶REAL		
------	------	-------	--	--

**VECTR MATH Menu** [2nd] [VECTR] [F3]

NAMES	EDIT	MATH	OPS	CPLX
cross	unitV	norm	dot	

**VECTR OPS (Operations) Menu**  $\boxed{2\text{nd}}$  [VECTR]  $\boxed{F4}$ 

NAMES	EDIT	MATH	OPS	CPLX						
dim	Fill	►Pol	►Cyl	►Sph	►	►Rec	li►vc	vcli		

**VECTR CPLX Menu**  $\boxed{2\text{nd}}$  [VECTR]  $\boxed{F5}$ 

NAMES	EDIT	MATH	OPS	CPLX
conj	real	imag	abs	angle

**CPLX (Complex Number) Menu**  $\boxed{2\text{nd}}$  [CPLX]

conj	real	imag	abs	angle	►	►Rec	►Pol			
------	------	------	-----	-------	---	------	------	--	--	--

**MATH Menu**  $\boxed{2\text{nd}}$  [MATH]

NUM	PROB	ANGLE	HYP	MISC	►	INTER				
-----	------	-------	-----	------	---	-------	--	--	--	--

**MATH NUM (Number) Menu**  $\boxed{2\text{nd}}$  [MATH]  $\boxed{F1}$ 

NUM	PROB	ANGLE	HYP	MISC						
round	iPart	fPart	int	abs	►	sign	min	max	mod	

**MATH PROB (Probability) Menu**  $\boxed{2\text{nd}}$  [MATH]  $\boxed{F2}$ 

NUM	PROB	ANGLE	HYP	MISC						
!	nPr	nCr	rand	randIn	►	randN	randBi			

**MATH ANGLE Menu**  $\boxed{2\text{nd}}$  [MATH]  $\boxed{F3}$ 

NUM	PROB	ANGLE	HYP	MISC
◦	r	◄	►DMS	

**MATH HYP (Hyperbolic) Menu** [2nd] [MATH] [F4]

NUM	PROB	ANGLE	HYP	MISC
sinh	cosh	tanh	$\sinh^{-1}$	$\cosh^{-1}$

▶	$\tanh^{-1}$				
---	--------------	--	--	--	--

**MATH MISC (Miscellaneous) Menu** [2nd] [MATH] [F5]

NUM	PROB	ANGLE	HYP	MISC
sum	prod	seq	lcm	gcd

▶	▸Frac	%	pEval	$x\sqrt{\quad}$	eval
---	-------	---	-------	-----------------	------

**CONS (Constants) Menu** [2nd] [CONS]

BLTIN	EDIT	USER		
-------	------	------	--	--

**CONS BLTIN (Built-In Constants) Menu** [2nd] [CONS] [F1]

BLTIN	EDIT	USER		
Na	k	Cc	ec	Rc

▶	Gc	g	Me	Mp	Mn	▶	$\mu 0$	$\varepsilon 0$	h	c	u
---	----	---	----	----	----	---	---------	-----------------	---	---	---

**CONV (Conversions) Menu** [2nd] [CONV]

LNGTH	AREA	VOL	TIME	TEMP	▶	MASS	FORCE	PRESS	ENERGY	POWER	▶	SPEED				
-------	------	-----	------	------	---	------	-------	-------	--------	-------	---	-------	--	--	--	--

**CONV LNGTH (Length) Menu** [2nd] [CONV] [F1]

LNGTH	AREA	VOL	TIME	TEMP
mm	cm	m	in	ft

▶	yd	km	mile	nmile	lt-yr	▶	mil	Ang	fermi	rod	fath
---	----	----	------	-------	-------	---	-----	-----	-------	-----	------

**CONV AREA Menu** [2nd] [CONV] [F2]

LNGTH	AREA	VOL	TIME	TEMP
ft <sup>2</sup>	m <sup>2</sup>	mi <sup>2</sup>	km <sup>2</sup>	acre

▶	in <sup>2</sup>	cm <sup>2</sup>	yd <sup>2</sup>	ha	
---	-----------------	-----------------	-----------------	----	--

**CONV VOL (Volume) Menu** [2nd] [CONV] [F3]

LNGTH	AREA	VOL	TIME	TEMP
liter	gal	qt	pt	oz

› 

cm <sup>3</sup>	in <sup>3</sup>	ft <sup>3</sup>	m <sup>3</sup>	cup
-----------------	-----------------	-----------------	----------------	-----

 › 

tsp	tbsp	ml	galUK	ozUK
-----	------	----	-------	------

**CONV TIME Menu** [2nd] [CONV] [F4]

LNGTH	AREA	VOL	TIME	TEMP
sec	mn	hr	day	yr

› 

week	ms	μs	ns	
------	----	----	----	--

**CONV TEMP (Temperature) Menu** [2nd] [CONV] [F5]

LNGTH	AREA	VOL	TIME	TEMP
°C	°F	°K	°R	

**CONV MASS Menu** [2nd] [CONV] [MORE] [F1]

MASS	FORCE	PRESS	ENRGY	POWER
gm	kg	lb	amu	slug

› 

ton	mton			
-----	------	--	--	--

**CONV FORCE Menu** [2nd] [CONV] [MORE] [F2]

MASS	FORCE	PRESS	ENRGY	POWER
N	dyne	tonf	kgf	lbf

**CONV PRESS (Pressure) Menu** [2nd] [CONV] [MORE] [F3]

MASS	FORCE	PRESS	ENRGY	POWER
atm	bar	N/m <sup>2</sup>	lb/in <sup>2</sup>	mmHg

› 

mmH <sub>2</sub>	inHg	inH <sub>2</sub> O		
------------------	------	--------------------	--	--

**CONV ENRGY (Energy) Menu** [2nd] [CONV] [MORE] [F4]

MASS	FORCE	PRESS	ENRGY	POWER
J	cal	Btu	ft-lb	kw-hr

› 

eV	erg	l-atm		
----	-----	-------	--	--



**CONV POWER Menu** [2nd] [CONV] MORE [F5]

MASS	FORCE	PRESS	ENRGY	POWER
hp	W	ftlb/s	cal/s	Btu/m

**STRNG Menu** [2nd] [STRNG]

"	sub	Ingth	EqSt	StEq
---	-----	-------	------	------

**LIST Menu** [2nd] [LIST]

{	}	NAMES	EDIT	OPS
---	---	-------	------	-----

**List Editor Menu** [2nd] [LIST] [F4]

{	}	NAMES	"	OPS	▶	REAL				
---	---	-------	---	-----	---	------	--	--	--	--

**LIST OPS (Operations) Menu** [2nd] [LIST] [F5]

{	}	NAMES	EDIT	OPS	▶	sum	prod	seq	livc	vc i	▶	Fill	aug	cSum	Deltal	Sortx	
												▶	Sorty	Select	SetLE	Form	

**The (Number) BASE Menu** [2nd] [BASE]

A-F	TYPE	CONV	BOOL	BIT
-----	------	------	------	-----

**BASE TYPE Menu** [2nd] [BASE] [F2]

A-F	TYPE	CONV	BOOL	BIT
b	h	o	d	

**CONV SPEED Menu** [2nd] [CONV] MORE MORE [F1]

SPEED				
ft/s	m/s	mi/hr	km/hr	knot

**LIST NAMES Menu** [2nd] [LIST] [F3]

{	}	NAMES	EDIT	OPS
fStat	xStat	yStat		

**BASE A-F (Hexadecimal) Menu** [2nd] [BASE] [F1]

A	TYPE	CONV	BOOL	BIT
B	C	D	E	F

**BASE CONV (Conversions) Menu** [2nd] [BASE] [F3]

A-F	TYPE	CONV	BOOL	BIT
▶Bin	▶Hex	▶Oct	▶Dec	

**BASE BOOL (Boolean) Menu**  $\boxed{2^{\text{nd}}}$  [BASE]  $\boxed{F4}$ 

A-F	TYPE	CONV	BOOL	BIT
and	or	xor	not	

**BASE BIT Menu**  $\boxed{2^{\text{nd}}}$  [BASE]  $\boxed{F5}$ 

A-F	TYPE	CONV	BOOL	BIT
rotR	rotL	shftR	shftL	

**TEST (Relational) Menu**  $\boxed{2^{\text{nd}}}$  [TEST]

==	<	>	≤	≥	▶	≠				
----	---	---	---	---	---	---	--	--	--	--

**MEM (Memory) Menu**  $\boxed{2^{\text{nd}}}$  [MEM]

RAM	DELET	RESET	TOL	ClrEnt
-----	-------	-------	-----	--------

**MEM DELET (Delete) Menu**  $\boxed{2^{\text{nd}}}$  [MEM]  $\boxed{F2}$ 

ALL	REAL	CPLX	LIST	VECTR	▶	MATRX	STRNG	EQU	CONS	PRGM	▶	GDB	PIC				
-----	------	------	------	-------	---	-------	-------	-----	------	------	---	-----	-----	--	--	--	--

**MEM RESET Menu**  $\boxed{2^{\text{nd}}}$  [MEM]  $\boxed{F3}$ 

RAM	DELET	RESET	TOL	ClrEnt
ALL	MEM	DFLTS		

**MEM RESET Are You Sure? Menu**

			YES	NO
--	--	--	-----	----

**STAT (Statistics) Menu**  $\boxed{2^{\text{nd}}}$  [STAT]

CALC	EDIT	PLOT	DRAW	VARs	▶	FCST				
------	------	------	------	------	---	------	--	--	--	--

When you press  $\boxed{2^{\text{nd}}}$  [STAT]  $\boxed{F2}$ , the list editor and list menu are displayed.

**STAT CALC (Calculations) Menu**  $\boxed{2^{\text{nd}}}$  [STAT]  $\boxed{F1}$ 

CALC	EDIT	PLOT	DRAW	VARs													
OneVa	TwoVa	LinR	LnR	ExpR	▶	PwrR	SinR	LgstR	P2Reg	P3Reg	▶	P4Reg	StReg				

**STAT PLOT Menu** [2nd] [STAT] [F3]

PLOT1	PLOT2	PLOT3	PIOn	PIOff
-------	-------	-------	------	-------

**Plot Type Menu** [2nd] [STAT] [F3] ( [F1], [F2], or [F3] ) ▾

PLOT1	PLOT2	PLOT3	PIOn	PIOff
SCAT	xyLINE	MBOX	HIST	BOX

**Plot Mark Menu** [2nd] [STAT] [F3] ( [F1], [F2], or [F3] ) ▾ ( [F1], [F2], or [F3] ) ▾ ▾ ▾

PLOT1	PLOT2	PLOT3	PIOn	PIOff
□	+	•		

**STAT DRAW Menu** [2nd] [STAT] [F4]

CALC	EDIT	PLOT	DRAW	VARS	
HIST	SCAT	xyLINE	BOX	MBOX	DRREG CLDRW DrawF STPIC RCPIC

**STAT VARS (Statistical Result Variables) Menu** [2nd] [STAT] [F5]

CALC	EDIT	PLOT	DRAW	VARS	
$\bar{x}$	$\sigma_x$	Sx	$\bar{y}$	$\sigma_y$	Sy   $\Sigma x$   $\Sigma x^2$   $\Sigma y$   $\Sigma y^2$   $\Sigma xy$   RegEq   corr   a   b
					n   minX   maxX   minY   maxY   Med   PRegC   Qrt1   Qrt3   tolMe

**CHAR (Character) Menu** [2nd] [CHAR]

MISC	GREEK	INTL		
------	-------	------	--	--

**CHAR MISC (Miscellaneous) Menu** [2nd] [CHAR] [F1]

MISC	GREEK	INTL			
?	#	&	%	'	!   @   \$   ~       $\acute{c}$   Ñ   ñ   Ç   ç

Ñ, ñ, Ç, and ç are valid as the first letter of a variable name.

%, ' , and ! can be functions.

All CHAR GREEK menu items are valid variable-name characters, including the first character.  $\pi$  ( $\overline{2nd}$  [ $\pi$ ]) is not valid as a character;  $\pi$  is a constant on the TI-86.

### CHAR GREEK Menu $\overline{2nd}$ [CHAR] $\overline{F2}$

MISC	GREEK	INTL			
$\alpha$	$\beta$	$\gamma$	$\Delta$	$\delta$	

▶

$\epsilon$	$\theta$	$\lambda$	$\mu$	$\rho$
------------	----------	-----------	-------	--------

▶

$\Sigma$	$\sigma$	$\tau$	$\phi$	$\Omega$
----------	----------	--------	--------	----------

### CHAR INTL (International Letter Symbols) Menu $\overline{2nd}$ [CHAR] $\overline{F3}$

MISC	GREEK	INTL		
		$\Lambda$		

## Handling a Difficulty

- 1 If you cannot see anything on the screen, you may need to adjust the contrast (Chapter 1).
  - ◆ To darken the screen, press and release  $\overline{2nd}$ , and then press and hold  $\blacktriangle$ .
  - ◆ To lighten the screen, press and release  $\overline{2nd}$ , and then press and hold  $\blacktriangledown$ .
- 2 If an error menu is displayed, follow the steps in Chapter 1. Refer to the Error Conditions section of the Appendix (page 393) for details about specific errors, if necessary.
- 3 If a checkerboard cursor ( $\blacksquare$ ) is displayed, then either you have entered the maximum number of characters in a prompt or memory is full. If memory is full, press  $\overline{2nd}$  [MEM]  $\overline{F2}$ , select a data type, and then delete some items from memory (Chapter 17).
- 4 If the busy indicator (dotted line) is displayed in the top-right corner, a graph or program has paused; the TI-86 is waiting for input. Press  $\overline{ENTER}$  to continue or press  $\overline{ON}$  to break.
- 5 If the calculator does not seem to work at all, be sure the batteries are fresh and that they are installed properly. Refer to battery information in Chapter 1.

## Error Conditions

When the TI-86 detects an error, it displays an error message **ERROR # type** and the error menu. Chapter 1 describes how to correct an error. This section describes possible causes for the errors and examples. To find the proper arguments for a function or instruction, as well as restrictions on those arguments, refer to Chapter 20: A to Z Function and Instruction Reference.

*Errors 1 through 5 do not occur during graphing. The TI-86 allows for undefined values on a graph.*

- |                        |   |
|------------------------|---|
| <b>01 OVERFLOW</b>     | <ul style="list-style-type: none"> <li>◆ You attempted to enter a number that is beyond the calculator's range.</li> <li>◆ You attempted to execute an expression with a result that is beyond the calculator's range.</li> </ul>   |
| <b>02 DIV BY ZERO</b>  | <ul style="list-style-type: none"> <li>◆ You attempted to divide by zero.</li> <li>◆ You attempted a linear regression with a vertical line.</li> </ul>   |
| <b>03 SINGULAR MAT</b> | <ul style="list-style-type: none"> <li>◆ You attempted to use a singular matrix (determinate = 0) as the argument for <math>^{-1}</math>, <b>Simult</b>, or <b>LU</b>.</li> <li>◆ You attempted a regression with at least one inappropriate list.</li> <li>◆ You attempted to use a matrix with repeated eigenvalues as the argument for <b>exp</b>, <b>cos</b>, or <b>sin</b>.</li> </ul> |
| <b>04 DOMAIN</b>       | <ul style="list-style-type: none"> <li>◆ You attempted to use an argument that is out of the range of valid values for the function or instruction.</li> <li>◆ You attempted a logarithmic or power regression with a -x or an exponential regression with a -y.</li> </ul>   |
| <b>05 INCREMENT</b>    | The increment in <b>seq</b> is <b>0</b> or has the wrong sign; the increment for a loop is <b>0</b> .   |
| <b>06 BREAK</b>        | You pressed <b>[ON]</b> to break a program, DRAW instruction, or expression evaluation.   |
| <b>07 SYNTAX</b>       | You entered a value; look for misplaced functions, arguments, parentheses, or commas; check the syntax description in the A to Z Reference.   |

<b>08 NUMBER BASE</b>	<ul style="list-style-type: none"><li>◆ You entered an invalid digit in a number base, such as <b>7b</b>.</li><li>◆ You attempted an operation that is not allowed in <b>Bin</b>, <b>Oct</b>, or <b>Hex</b> base mode.</li></ul>
<b>09 MODE</b>	You attempted to store to a window variable of a noncurrent graphing mode, or to use an instruction valid only in noncurrent graphing modes; for example, using <b>DrInV</b> in <b>Pol</b> , <b>Param</b> , or <b>DifEq</b> graphing mode.
<b>10 DATA TYPE</b>	<ul style="list-style-type: none"><li>◆ You entered a value or variable that is an inappropriate data type.</li><li>◆ You entered an argument that is an inappropriate data type for a function or an instruction, such as a program name for <b>sortA</b>.</li><li>◆ In an editor, you entered a data type that is not allowed; check the appropriate chapter.</li><li>◆ You attempted to store data to a protected data type, such as a constant, program, picture, or graph database.</li><li>◆ You attempted to store inappropriate data to a restricted built-in variable, such as the list names <b>xStat</b>, <b>yStat</b>, and <b>fStat</b>.</li></ul>
<b>11 ARGUMENT</b>	You attempted to execute a function or instruction without all the arguments.
<b>12 DIM MISMATCH</b>	You attempted to use two or more lists, matrices, or vectors as arguments, but the dimensions of all arguments are not equal, such as <b>{1,2}+{1,2,3}</b> .
<b>13 DIMENSION</b>	<ul style="list-style-type: none"><li>◆ You entered an argument with an inappropriate dimension.</li><li>◆ You entered a matrix or vector dimension <math>&lt; 1</math> or <math>&gt; 255</math> or a noninteger.</li><li>◆ You attempted to invert a matrix that is not a square matrix.</li></ul>
<b>14 UNDEFINED</b>	You are referencing a variable that currently is not defined.
<b>15 MEMORY</b>	Memory is insufficient to perform the desired command; you must delete items from memory (Chapter 17) before executing this command.
<b>16 RESERVED</b>	You attempted to use a built-in variable inappropriately.
<b>17 INVALID</b>	You attempted to reference a variable or use a function where it is not valid.

<b>18 ILLEGAL NEST</b>	You attempted to use an invalid function in an argument for <b>seq(</b> or a <b>CALC</b> function; for example, <b>der1(der1(x^3,x),x)</b> .
<b>19 BOUND</b>	You defined an upper bound that is less than the specified lower bound or a lower bound that is greater than the specified upper bound.
<b>20 GRAPH WINDOW</b>	<ul style="list-style-type: none"> <li>◆ One or more window variable values is incompatible with the others for defining the graph screen; for example, you defined <b>xMax &lt; xMin</b>.</li> <li>◆ Window variables are too small or too large to graph correctly; for example, you attempted to zoom out beyond the calculator's range.</li> </ul>
<b>21 ZOOM</b>	A <b>ZOOM</b> operation resulted in an error; you attempted to define <b>ZBOX</b> with a line.
<b>22 LABEL</b>	In programming, the <b>Goto</b> instruction label is not defined with a <b>Lbl</b> instruction.
<b>23 STAT</b>	<ul style="list-style-type: none"> <li>◆ You attempted a stat calculation with at least one inappropriate list, such as a list with less than two data points.</li> <li>◆ At least one element of a frequency list is <math>&lt; 0</math>.</li> <li>◆ <math>(xMax - xMin)/xScl \leq 63</math> must be true when plotting a histogram.</li> </ul>
<b>24 CONVERSION</b>	When converting measurements, the units are incompatible, as in volts to liters.
<b>25 SOLVER</b>	<ul style="list-style-type: none"> <li>◆ In the solver editor, the equation does not contain a variable.</li> <li>◆ You attempted to graph with the cursor positioned on bound.</li> </ul>
<b>26 SINGULARITY</b>	In the solver editor, the equation contains a singularity, which is a point at which the function is not defined.
<b>27 NO SIGN CHNG</b>	The solver did not detect a sign change.
<b>28 ITERATIONS</b>	The solver has exceeded the maximum permitted number of iterations.
<b>29 BAD GUESS</b>	<ul style="list-style-type: none"> <li>◆ The initial guess was outside the specified bounds.</li> <li>◆ The initial guess and several points around the guess are undefined.</li> </ul>

*Errors 26 through 29 occur during the solving process. Examine a graph of the function or a graph of the variable vs. left-rt in the SOLVER. If the equation has a solution, change bounds and/or the initial guess.*

<b>30 DIF EQ SETUP</b>	In <b>DifEq</b> graphing mode, equations in the equation editor must be from <b>Q'1</b> to <b>Q'9</b> and each must have an associated initial condition from <b>QI1</b> to <b>QI9</b> .
<b>31 DIF EQ MATH</b>	The step size used by the fitting algorithm has become too small; check the equations and initial values; try a larger value for the window variable <b>difTol</b> ; try changing <b>tMin</b> or <b>tMax</b> to examine a different region of the solution.
<b>32 POLY</b>	All coefficients are <b>0</b> .
<b>33 TOL NOT MET</b>	The algorithm cannot return a result accurate to the requested tolerance.
<b>34 STAT PLOT</b>	You attempted to display a stat plot that references an undefined list.
<b>35 AXES</b>	You attempted to plot a <b>DifEq</b> graph with improper axes set.
<b>36 FLD/ORDER</b>	<ul style="list-style-type: none"><li>◆ You attempted to plot a 2nd-order or higher differential equation with <b>SlpFld</b> field format set; change field format or modify the order.</li><li>◆ You attempted to plot a 3rd-order or higher differential equation with <b>DirFld</b> field format set; change field format or modify the order.</li></ul>
<b>37 LINK MEMORY FULL</b>	You attempted to transmit an item with insufficient available memory in the receiving unit; skip the item or cancel the transmission.
<b>38 LINK TRANSMISSION ERROR</b>	<ul style="list-style-type: none"><li>◆ Unable to transmit item; check to see that the cable is firmly connected to both units and the receiving unit is ready to receive data (Chapter 18).</li><li>◆ You pressed <b>[ON]</b> to break during transmission.</li></ul>
<b>39 LINK DUPLICATE NAME</b>	You attempted to transmit an item when an item with the same name already exists in the receiving unit.



## Equation Operating System (EOS™)

The Equation Operating System (EOS) governs the order of evaluation on the TI-86. Calculations within parentheses are evaluated first, and then EOS evaluates functions within an expression in this order:

*Within a priority level, EOS evaluates functions from left to right.*

*Multi-argument functions, such as **nDeriv(A2,A,6)**, are evaluated as they are encountered.*

*TI-86 implied multiplication rules differ from those of the TI-85. For example, the TI-86 evaluates  $1/2x$  as  $(1/2)*x$ , while the TI-85 evaluates  $1/2x$  as  $1/(2*x)$ .*

- 1st Functions that are entered after the argument, such as  $^2$ ,  $^{-1}$ ,  $!$ ,  $^{\circ}$ ,  $^{\circ}$ ,  $^{\circ}$ , and conversions
- 2nd Powers and roots, such as  $2^5$  or  $5^{\sqrt{32}}$
- 3rd Single-argument functions that precede the argument, such as  $\sqrt{}$ , **sin**(, or **log**(
- 4th Permutations (**nPr**) and combinations (**nCr**)
- 5th Multiplication, implied multiplication, and division
- 6th Addition and subtraction
- 7th Relational functions, such as  $>$  or  $\leq$
- 8th Logic operator **and**
- 9th Logic operators **or** and **xor**

### Implied Multiplication

The TI-86 recognizes implied multiplication, so you need not press  $\square$  to express multiplication in all cases. For example, the TI-86 interprets  $2\pi$ ,  $4\sin(46)$ ,  $5(1+2)$ , and  $(2*5)7$  as implied multiplication.

### Parentheses

All calculations inside a pair of parentheses are completed first. For example, in the expression  $4(1+2)$ , EOS evaluates  $1+2$  inside the parentheses first, and then multiplies 3 by 4.

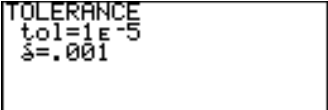
$4*1+2$	6
$4(1+2)$	12

You can omit the close parenthesis ( ) at the end of an expression. All open parenthetical elements are closed automatically at the end of an expression. This is also true for open parenthetical elements that precede the store or display-conversion instructions.

Open parentheses after list names, matrix names, or equation function names are not interpreted as implied multiplication. Arguments that follow these open parentheses are specified list elements, matrix elements, or values for which to solve the equation function.

## TOL (The Tolerance Editor) 2nd [MEM] F4

On the TI-86, the computational accuracy of some functions is controlled by the variables **tol** and  $\delta$ . The values stored to these variables may affect the speed at which the TI-86 calculates or plots.



```
TOLERANCE
tol=1E-5
delta=.001
```

The variable **tol** defines the tolerance in calculating the functions **fnInt**(, **fMin**(, **fMax**(, and **arc**(, and the GRAPH MATH operations  $\Sigma f(x)$ , **FMIN**, **FMAX**, and **ARC** (Chapter 6). **tol** must be a positive value  $\geq 1E-12$ .

The value stored to  $\delta$  must be a positive real number.  $\delta$  defines the step size the TI-86 uses to calculate the functions **arc** in **dxNDer** mode; **nDer**; and the operations **dy/dx**, **dr/dθ**, **dy/dt**, **dx/dt**, **INFLC**, **TANLN**, and **ARC**, all in **dxNDer** mode (Chapter 6).

To store a value to **tol** or  $\delta$  on the home screen or in a program, use STO►. You can select **tol** and  $\delta$  from the CATALOG. Also, you can enter **tol** directly and select  $\delta$  from the CHAR GREEK menu.

## Computational Accuracy

To maximize accuracy, the TI-86 carries more digits internally than it displays. Values are stored in memory using up to 14 digits with a 3-digit exponent.

- ◆ You can store values up to 12 digits long to most window variables. To **xSci**, **ySci**, **tStep**, and **θStep**, you can store values up to 14 digits long.
- ◆ When a value is displayed, the displayed value is rounded as specified by the mode setting (Chapter 1), with a maximum of 12 digits and a 3-digit exponent.
- ◆ Chapter 4 describes calculations in hexadecimal, octal, and binary number bases.

## Support and Service Information

### Product Support

#### ***Customers in the U.S., Canada, Puerto Rico, and the Virgin Islands***

For general questions, contact Texas Instruments Customer Support:

phone: **1-800-TI-CARES (1-800-842-2737)**

e-mail: **[ti-cares@ti.com](mailto:ti-cares@ti.com)**

For technical questions, call the Programming Assistance Group of Customer Support:

phone: **1-972-917-8324**

#### ***Customers outside the U.S., Canada, Puerto Rico, and the Virgin Islands***

Contact TI by e-mail or visit the TI **Calculator** home page on the World Wide Web.

e-mail: **[ti-cares@ti.com](mailto:ti-cares@ti.com)**

Internet: **[education.ti.com](http://education.ti.com)**

## Product Service

### *Customers in the U.S. and Canada Only*

Always contact Texas Instruments Customer Support before returning a product for service.

### *Customers outside the U.S. and Canada*

Refer to the leaflet enclosed with this product or contact your local Texas Instruments retailer/distributor.

## Other TI Products and Services

Visit the TI **Calculator** home page on the World Wide Web.

[education.ti.com](http://education.ti.com)

## Warranty Information

### Customers in the U.S. and Canada Only

#### One-Year Limited Warranty for Commercial Electronic Product

This Texas Instruments electronic product warranty extends only to the original purchaser and user of the product.

**Warranty Duration.** This Texas Instruments electronic product is warranted to the original purchaser for a period of one (1) year from the original purchase date.

**Warranty Coverage.** This Texas Instruments electronic product is warranted against defective materials and construction. **THIS WARRANTY IS VOID IF THE PRODUCT HAS BEEN DAMAGED BY ACCIDENT OR UNREASONABLE USE, NEGLIGENCE, IMPROPER SERVICE, OR OTHER CAUSES NOT ARISING OUT OF DEFECTS IN MATERIALS OR CONSTRUCTION.**

**WARRANTY DISCLAIMERS. ANY IMPLIED WARRANTIES ARISING OUT OF THIS SALE, INCLUDING BUT NOT LIMITED TO THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE, ARE LIMITED IN DURATION TO THE ABOVE ONE-YEAR PERIOD. TEXAS INSTRUMENTS SHALL NOT BE LIABLE FOR LOSS OF USE OF THE PRODUCT OR OTHER INCIDENTAL OR CONSEQUENTIAL COSTS, EXPENSES, OR DAMAGES INCURRED BY THE CONSUMER OR ANY OTHER USER.**

Some states/provinces do not allow the exclusion or limitation of implied warranties or consequential damages, so the above limitations or exclusions may not apply to you.

**Legal Remedies.** This warranty gives you specific legal rights, and you may also have other rights that vary from state to state or province to province.

**Warranty Performance.** During the above one (1) year warranty period, your defective product will be either repaired or replaced with a reconditioned model of an equivalent quality, (at TI's option) when the product is returned, postage prepaid, to Texas Instruments Service Facility. The warranty for the repaired or replacement unit will continue for the warranty of the original unit or six (6) months, whichever is longer. Other than the postage requirement, no charge will be made for such repair and/or replacement. TI strongly recommends that you insure the product for value prior to mailing.

**Software.** Software is licensed, not sold. TI and its licensors do not warrant that the software will be free from errors or meet your specific requirements. **All software is provided "AS IS."**

**Copyright.** The software and any documentation supplied with this product are protected by copyright.

## Australia & New Zealand Customers only

### One-Year Limited Warranty for Commercial Electronic Product

This Texas Instruments electronic product warranty extends only to the original purchaser and user of the product.

**Warranty Duration.** This Texas Instruments electronic product is warranted to the original purchaser for a period of one (1) year from the original purchase date.

**Warranty Coverage.** This Texas Instruments electronic product is warranted against defective materials and construction. This warranty is void if the product has been damaged by accident or unreasonable use, neglect, improper service, or other causes not arising out of defects in materials or construction.

**Warranty Disclaimers.** Any implied warranties arising out of this sale, including but not limited to the implied warranties of merchantability and fitness for a particular purpose, are limited in duration to the above one-year period. Texas Instruments shall not be liable for loss of use of the product or other incidental or consequential costs, expenses, or damages incurred by the consumer or any other user.

Some jurisdictions do not allow the exclusion or limitation of implied warranties or consequential damages, so the above limitations or exclusions may not apply to you.

**Legal Remedies.** This warranty gives you specific legal rights, and you may also have other rights that vary from jurisdiction to jurisdiction.

**Warranty Performance.** During the above one (1) year warranty period, your defective product will be either repaired or replaced with a new or reconditioned model of an equivalent quality (at TI's option) when the product is returned to the original point of purchase. The repaired or replacement unit will continue for the warranty of the original unit or six (6) months, whichever is longer. Other than your cost to return the product, no charge will be made for such repair and/or replacement. TI strongly recommends that you insure the product for value if you mail it.

**Software.** Software is licensed, not sold. TI and its licensors do not warrant that the software will be free from errors or meet your specific requirements. All software is provided "AS IS."

**Copyright.** The software and any documentation supplied with this product are protected by copyright.

### **All Customers outside the U.S. and Canada**

For information about the length and terms of the warranty, refer to your package and/or to the warranty statement enclosed with this product, or contact your local Texas Instruments retailer/distributor.



# Index

" (string), 216, 227  
" (List Editor menu), 156  
! (factorial), 294  
➤, 362  
≥ (greater than or equal to), 56, 301  
≤ (less than or equal to), 55, 312  
≠ (not equal to), 56, 326  
π (pi), 48  
√ (square root), 360  
[√] (square root) key, 48  
x̄ (statistical result variable), 193  
ȳ (statistical result variable), 193  
<sup>-1</sup> (inverse), 48, 309  
➤dim, 184, 281  
➤dimL, 282  
f(x) (function numerical integral), 96, 98  
ΔTbl (table step), 113  
ox (statistical result variable), 193

Σx<sup>2</sup> (statistical result variable), 193  
oy (statistical result variable), 193  
% (percent), 52, 334  
< (less than), 55, 312  
= (assign to), 270  
= (equals), 290  
== (relational equals), 55, 291  
> (greater than), 55, 300  
[ ], 319, 369  
^ (exponent), 48  
{ }, 316  
10<sup>n</sup> (10 raised to *n* power), 48, 337

## A

abs (absolute value), 49, 71, 175, 185, 267  
addition (+), 267  
ALL, 43, 232  
ALL-, 77  
ALL+, 77  
ALPHA character, 22  
ALPHA cursor, 22

alpha cursor, 22  
ALPHA key, 21  
ALPHA-lock, 22, 44  
canceling, 22  
setting, 22  
and (Boolean), 69, 268  
angle, 71, 175, 185, 269  
expressed in degrees, 51  
angle modes, 35  
angle values, 35  
Ans (last answer), 29, 30, 41, 269  
answer  
displaying, 19  
storing to a variable, 41  
APD. *See* Automatic Power Down  
ARC, 96, 98  
arc(), 54, 269  
argument, 25  
Asm (assembly language program), 269  
AsmComp (compile assembly language program), 226, 270  
AsmPrgm (assembly language program), 226, 270

assembly language programs, 225  
assignment, 270  
attached formulas  
executing, 164  
resolving errors, 165  
attached-formula list  
comparing, 163  
creating, 162  
editing elements, 166  
aug(), 160, 184, 270  
Automatic Power Down, 17  
automatic regression equation  
storage, 191  
AXES, 137  
Axes editor, 137  
field formats, 137  
Axes(), 271  
AxesOff, 84, 271  
AxesOn, 84, 271

## B

b (binary), 271  
backup battery, 16

- BASE A-F (Hexadecimal) menu,  
67
- BASE BIT menu, 69
- BASE BOOL (Boolean) menu,  
68
- BASE CONV (Conversion)  
menu, 68
- BASE menu, 66
- BASE TYPE menu, 67
- base type symbol, 67
- batteries, 2, 16-18
- battery compartment, 16
- BCKUP (memory backup), 237
- Bin (binary), 35, 272
- Bin (to binary), 68, 272
- binary integer, 271
- binary number base, 35, 66
- Boolean operators, 68, 268,  
325, 328, 370
- bound={-1E99,1E99}, 204
- bounds, 204
- BOX (GRAPH ZOOM menu),  
14, 92, 93
- Box (stat plot), 272
- BOX (ZOOM menu), 208
- break (program), 222
- BREAK menu, 26
- built-in constants, 58
- built-in variables, 39, 45, 138
- busy indicator, 26, 85
- C**
- CALC (Calculus) menu, 54
- calculating derivatives, 7
- calculation  
interrupting, 26
- calculus functions, 54
- CATALOG, 25, 38
- Quick-Find Locator, 262
- CATLG (CATALOG), 43
- CATLG-VARS (CATALOG  
Variables) menu, 43
- changing TI-86 settings, 39
- CHAR (Character) menu, 45
- CHAR GREEK menu, 46
- CHAR INTL (International)  
menu, 46
- CHAR MISC (Miscellaneous)  
menu, 46
- characters, 19
- alpha, 22
- blue, 21, 22
- case, 22
- characters (continued)
- deleting, 23
- entering, 21
- second, 22
- yellow, 21
- check RAM screen, 230
- CIRCL (circle), 105, 106
- Circl(), 273
- circles  
drawing, 106
- CLDRW (clear drawing), 103,  
105, 273
- clearing CUSTOM menu items,  
45
- clearing ENTRY storage area,  
29
- CILCD (clear LCD), 216, 273
- ClrEnt (clear entry), 232, 273
- CITbl (clear table), 114, 216,  
273
- cnorm (column norm), 183, 273
- command line, 220
- complements (binary numbers),  
66
- complex matrix, 180
- Complex Number menu, 71
- complex number modes, 35
- complex number variables, 43
- complex numbers, 29, 70
- as list elements, 156
- displaying as result, 5
- entering, 20
- in results, 70
- separator, 70
- using in expressions, 71
- complex values, 48
- concatenation (+), 274
- cond (condition number), 183,  
274
- conj (complex conjugate), 71,  
175, 185, 275
- connecting instructions, 235
- CONS (constants), 43
- CONS (Constants) menu, 58
- CONS BLTIN (Built-In  
Constants) menu, 58
- CONS EDIT menu, 60
- consecutive entries, 26
- Constant Memory feature, 17, 34
- constants, 59
- built-in, 58
- defined, 58
- name, 61
- user-created, 58, 60

- contrast  
 adjusting, 2, 18
- CONV (Conversions) menu, 62
- CONV AREA menu, 63
- CONV ENRGY (Energy) menu, 64
- CONV FORCE menu, 64
- CONV LENGTH (Length) menu, 63
- CONV MASS menu, 64
- CONV POWER menu, 64
- CONV PRESS (Pressure) menu, 64
- CONV SPEED menu, 64
- CONV TEMP (Temperature) menu, 8, 63
- CONV TIME menu, 63
- CONV VOL (Volume) menu, 63
- conversions  
 ▶Bin, 272  
 ▶Dec, 279  
 ▶DMS, 51, 285  
 ▶Frac, 52, 298  
 ▶Hex, 303  
 ▶Oct, 327  
 ▶Pol, 336  
 ▶REAL, 156
- conversions (continued)  
 ▶Rec, 343  
 ▶Sph, 360  
 Eq▶St, 227  
 li▶vc, 160  
 St▶Eq(), 227, 361  
 vc▶li, 160
- converting a value expressed as a rate, 65
- converting Fahrenheit to Celsius, 8
- converting units of measure, 61
- CoordOff, 84, 275
- CoordOn, 84, 275
- copying variable value, 41
- corr (correlation coefficient), 193
- cos (cosine), 48, 186, 276
- cos<sup>-1</sup> (arccosine), 48, 276
- cosh (hyperbolic cosine), 51, 277
- cosh<sup>-1</sup> (inverse hyperbolic cosine), 51, 277
- CPLX (complex number variables), 43, 71
- cross(), 173, 277
- cSum( (cumulative sum), 160, 278
- current entry, 19  
 clearing, 23
- current item, 38
- cursor, 17, 22  
 ALPHA, 22  
 alpha, 22  
 changing, 23  
 direction keys, 23  
 entry, 22  
 free-moving, 128, 144, 205  
 full, 22  
 insert, 22  
 location, 19, 20, 21, 25  
 moving, 23  
 second, 22  
 selection, 38  
 trace, 90
- curves  
 drawing, 107
- CUSTOM menu, 44  
 clearing items, 45  
 copying items, 44
- Customer Support, 392
- ▶Cyl (to cylindrical), 174, 278
- CylV (cylindrical vector coordinate mode), 36, 278
- D**
- d (decimal), 278
- data type selection screen, 42
- Dec (decimal number base mode), 278
- Dec (decimal), 35, 65  
 ▶Dec (to decimal), 279
- decimal, 20
- decimal mode, 34, 35, 65  
 fixed (012345678901), 35  
 floating, 35
- decimal number, 278
- decimal number base, 35
- decimal point, 35
- degree angle mode, 35, 75, 279
- degree complex-number mode, 70
- degree entry (°), 279
- degrees°, 51
- degrees/minutes/seconds form, 51
- DELC (delete column), 179
- DELET, 60

- DELf (delete function), 77  
DELi (delete element), 170  
DELr (delete row), 179  
Deltalst( (delta list), 160, 279  
DelVar( (delete variable), 219, 280  
der1( (first derivative), 54, 280  
der2( (second derivative), 54, 280  
derivatives  
    calculating, 7  
det (determinant), 183, 281  
DFLTS (defaults), 232  
DiffEq (differential equation mode), 35, 74, 239, 281  
differential equation editor, 134  
differential equation graphs, 74  
    displaying, 138  
    drawing, 145  
    mode, 35  
differential equations  
    changing to first order, 142  
    defining graph, 132  
    drawing solutions, 148  
    DrEqu(, 287  
    editor, 134  
    EXPLR, 148  
differential equations (continued)  
    graphing, 132, 137, 139, 141, 142  
    initial conditions editor, 136  
    mode, 144  
    Q'n equation variables, 135  
    setting axes, 137  
    setting graph format, 133  
    setting graphing mode, 132  
    solving, 139  
    tracing, 144  
    using EVAL, 150  
    window variables, 135  
differentiation modes, 36  
diffTol (tolerance), 136  
dim (dimension), 173, 184, 281  
dimL (dimension of list), 159, 282  
DirFld (direction field), 134, 282  
Disp (display), 216, 283  
DispG (display graph), 283  
display, 17  
display contrast  
    adjusting, 17, 18  
displaying a menu, 31  
DispT (display table), 284  
DIST (distance), 96, 98  
division (/), 284  
division symbol, 3  
►DMS (to degrees/  
    minutes/seconds), 51, 285  
dot(, 173, 285  
dr/dθ, 122  
DRAW, 75, 88  
DrawDot, 84, 285  
DrawF (draw function), 103, 107, 286  
drawing  
    circles, 106  
    differential equation graphs, 145  
    freehand points, lines, curves, 107  
    function, tangent line, inverse function, 107  
    line segments, 105  
    lines, 105, 106  
    parametric graphs, 130  
    points, 108  
    polar graphs, 122  
drawing tools, 101  
drawings  
    clearing, 103  
    drawings (continued)  
        recalling, 102  
        saving, 102  
    DrawLine, 84, 286  
    DrEqu( (draw equation), 145, 287  
    DrInv (draw inverse), 103, 107, 287  
    DS<( (decrement and skip), 219, 288  
    DUPLICATE NAME menu, 241  
    dx/dt, 130  
    dxDer1 (exact differentiation), 36, 75, 288  
    dxNDer (numeric differentiation), 36, 75, 288  
    dy/dt, 130  
    dy/dx, 96, 99, 130  
**E**  
E (exponent), 48, 292  
e^ (e raised to power), 288  
editing equations, 205  
editor menu, 33  
eigVc (eigenvector), 183, 289  
eigVl (eigenvalue), 183, 289

- element  
 matrix, 181
- ellipsis  
 at end of line, 19  
 in matrix row, 179
- Else, 218, 306
- e-mail address (TI Customer Support), 392
- End, 218, 290, 297, 306
- Eng (engineering notation), 34, 20, 290
- entry  
 executing, 19  
 storing to, 29
- entry cursor, 18, 22, 23
- [ENTRY] key, 19
- ENTRY Storage Area, 28, 29
- EOS. *See* Equation Operating System
- EqSt( (equation to string), 227, 290
- eqn (equation) variable, 54, 203, 205
- EQU (equation variables), 43
- equal (=), 290
- equal to (==), 291
- equation  
 entering, 203  
 evaluating, 122, 130
- equation coefficients  
 storing to a variable, 210
- equation editor, 74, 75, 76, 80  
 entering a function, 77  
 graph styles, 77  
 parametric, 126  
 polar, 118
- Equation Editor menu, 76
- Equation Operating System, 397
- equation results  
 storing to a variable, 210
- equation solver, 40, 202  
 graph tools, 207
- equation storage  
 automatic regression, 191
- equation variables, 40, 43, 78
- equation-entry editor, 203
- equations  
 editing, 205  
 solving, 206
- error conditions, 393
- error menu, 31
- error message, 27
- error type, 27
- errors, 17, 27  
 correcting, 27  
 diagnosing, 27  
 from attached formulas, 165
- EStep, 136
- Euler method, 133, 291
- eval, 52, 76, 88, 101, 122, 130, 150, 291
- evalF(, 54, 292
- evaluating a function for x, 101
- evaluating equations, 122, 130
- $e^x$  (constant e raised to a power), 48
- exact differentiation, 36
- EXIT (cancel data transmission), 241
- exiting a menu, 6, 33
- exp variable, 54, 203
- EXPLR (explore), 148
- exponent (E), 292
- ExpR (exponential regression), 190, 293
- expression, 18, 20, 24, 25, 26, 30, 48  
 editing, 4  
 entering, 24
- expression (continued)  
 entering a list, 153  
 evaluating, 29, 30  
 using a complex number, 71  
 using a vector, 172  
 using matrix, 181
- F**
- factorial (!), 50, 294
- Fahrenheit  
 converting to Celsius, 8
- family of curves  
 graphing, 86  
 in parametric graphs, 129  
 in polar graphs, 120
- fcstx (forecast x), 294
- fcsty (forecast y), 294
- feature symbol, 39
- field formats, 134
- Fill, 184
- Fill(, 160, 173, 295
- Fix, 295
- FldOff (slope and direction fields off), 134, 295
- fldPic (field) variable, 138
- Float, 35, 295

- FMAX (function maximum), 96, 97  
fMax( (function maximum), 54, 296  
FMIN (function minimum), 96, 97  
fMin( (function minimum), 54, 296  
fnInt( (function integral), 54, 296  
FnOff (functions off), 296  
FnOn (functions on), 297  
For(, 218, 297  
Form(, 161, 298  
FORMT (graph format), 76  
formulas  
    attaching, 163  
    attaching to list name, 162  
    detaching, 166  
fPart (fractional part), 49, 176, 186, 298  
►Frac (to fractions), 52, 298  
fraction, 3, 19  
free-moving cursor, 84, 144  
    parametric graphs, 128  
    polar graphs, 119  
fStat (frequency list), 189  
full cursor, 22  
Func (function mode), 35, 74, 239, 299  
function graphs, 73, 74  
    mode, 35  
functions, 25, 38  
    deleting, 77  
    deselecting, 13  
    drawing, 107  
    entering, 25  
    entering in the equation  
        editor, 76, 77, 78  
    evaluating, 101  
    keyboard, 48  
    plotting, 11  
    tracing, 11  
    using with lists, 5, 161  
**G**  
gcd( (greatest common denominator), 52, 299  
GDB (graph database), 43  
GDB variable, 102  
Get(, 299  
getKey (get key), 216, 300  
    key code diagram, 217  
GOTO, 26, 27, 300  
Goto (PRGM CTL menu), 219, 224  
graph, 75  
    defining, 74  
    displaying, 85  
    family of curves, 86  
    interrupting, 26  
    modifying, 85  
    pausing, 85  
    shading, 104  
    stopping, 85  
GRAPH (Solver menu), 206  
graph database (GDB), 102  
    recalling, 76  
GRAPH DRAW menu, 75, 103, 122, 145  
graph format  
    differential equations, 133, 137  
    parametric graphs, 128  
    polar graphs, 119  
    screen, 76  
    setting, 83  
GRAPH LINK, 235  
GRAPH MATH menu, 75, 95, 122, 130  
GRAPH MATH operations  
    effect of other settings, 96  
    using ff(x), DIST, or ARC, 98  
    using dy/dx or TANLN, 99  
    using ISECT, 100  
    using ROOT, FMIN, FMAX, or INFLC, 97  
    using YICPT, 100  
GRAPH menu, 27, 31, 75, 88, 117, 126, 133  
graph modes, 35  
    setting, 74  
    differential equations, 144  
    function,  
        parametric, 126  
        polar, 35, 117  
graph screen, 75  
    setting window variables, 81  
graph screen dimensions, 75  
graph styles, 79  
    changing, 10  
    GrStl(, 302  
    setting, 79  
graph tools  
    in differential equation  
        graphs, 144  
    in equation solver, 207

- graph tools (continued)
  - in parametric graphs, 128
  - in polar graphs, 119
- graph zoom
  - defining custom, 93
  - defining screen, 92
  - setting zoom factors, 93
  - Smart Graph, 94
  - zooming in, 92, 93
  - zooming out, 92, 93
- GRAPH ZOOM menu, 75, 91, 147
- graphing accuracy, 89
- greater than (>), 300
- greater than or equal to ( $\geq$ ), 301
- grid points, 84
- GridOff, 84, 301
- GridOn, 84, 302
- GrStl( (graph style), 220, 302
- Guess, 204
  - in interactive solver editor, 205
- H**
- h (hexadecimal), 302
- Hex (hexadecimal), 35, 302
  - ▶Hex (to hexadecimal), 68, 303
- hexadecimal characters menu, 67
- hexadecimal number base, 35, 66
- Hist (histogram), 303
- home screen, 17, 18, 23, 24, 26, 27
  - displaying entries and answers, 18
- Horiz, 304
- HORIZ (horizontal line), 105, 106
- hyperbolic functions, 51
- I**
- IAsk, 304
- IAuto, 304
- ident (identity), 184, 304
- If, 218, 305, 306
- imag (imaginary), 71, 175, 185, 306
- imaginary portion of complex number, 71
- implied multiplication, 397
- INFLC (inflection point), 96, 97
- INITC (initial conditions), 136
- InpSt, 217, 307
- Input (PRGM I/O menu), 216, 307
- Input CBLGET, 216
- INSc (insert column), 179
- insert cursor, 22, 23
  - canceling, 23
- INSf (insert function), 77
- INSi (insert element), 170
- INSr (insert row), 179
- installing batteries, 16
- instructions, 25
  - entering, 25
  - executing, 19
- int (integer), 49, 176, 186, 308
- integer part, 49
- integer part of real numbers
  - displaying, 6
- inter( (interpolate), 309
- interactive-solver editor, 204
  - bounds, 204
- international letters, 46
- Internet
  - downloading programs, 235
  - e-mail address (TI Customer Support), 392
- interpolate/extrapolate editor, 53
- interrupting a calculation, 26
- interrupting a graph, 26, 27
- interrupting a program, 222
- inverse, 309
- inverse function
  - drawing, 107
- IPart (integer part), 6, 49, 176, 186, 309
- IS>( (increment and skip), 219, 310
- ISECT (intersection), 96, 100
- items on menus, 31
- K**
- keys, 48
  - 2nd, 21
  - ALPHA, 21
  - primary function, 19, 21, 22
- key code diagram, 217
- L**
- LabelOff, 84, 310
- LabelOn, 84, 310

- last answer, 28, 29
    - storing to variable, 3
  - last entry, 26, 28
  - Lbl (label), 219, 224, 311
  - lcm( (least common multiple), 52, 311
  - LCust( (load custom menu), 220, 311
  - left-rt, 202
  - length of segment of curve, 54
  - less than (<), 312
  - less than or equal to ( $\leq$ ), 312
  - LgstR (logistic regression), 190, 193, 313
  - li $\rightarrow$ vc (list to vector), 160, 174, 316
  - LINE, 104, 105
  - Line(, 314
  - Lines
    - drawing, 107
  - LINK menu, 236
  - LINK SEND menu, 236
  - LINK SEND85 menu, 239
  - linking instructions, 235
  - linking options, 234
  - LinR (linear regression), 190, 315
  - list, 29, 43, 52
    - as an argument, 161
    - attached formulas, 165
    - attaching formula, 162, 166
    - braces { }, 316
    - comparing, 163
    - creating, 157
    - deleting an element, 158
    - deleting from memory, 154
    - detaching formulas, 166
    - displaying list elements, 154
    - editing elements, 166
    - entering in an expression, 153
    - inserting, 157
    - removing from list editor, 158
    - storing, 154
    - uses, 152
    - using with function, 5
  - list editor, 31, 67, 156, 188
    - attaching formulas, 163, 164
    - removing a list, 158
  - List Editor menu, 156
  - list element
    - complex, 156
    - deleting, 158
    - list element (continued)
      - displaying, 155, 158
      - editing, 158
      - storing a value to, 155
    - list entry { }, 316
    - LIST menu, 152
    - list names, 43
    - LIST NAMES menu, 153, 189
    - LIST OPS menu, 159
    - ln (natural log), 48, 316
    - lngh (length of string), 227, 316
    - LnR (logarithmic regression), 190, 317
    - log, 48, 318
    - low-battery message, 16, 18
    - lower menu, 32
    - LU( (lower-upper), 183, 318
- M**
- Macintosh
    - linking to, 235
  - MATH, 75
  - MATH (Graph menu), 88
  - MATH ANGLE menu, 51
  - MATH HYP (Hyperbolic) menu, 51
  - MATH menu, 31, 49
  - MATH MISC (Miscellaneous) menu, 52
  - MATH NUM (Number) menu, 31, 49
  - MATH PROB (Probability) menu, 50
  - mathematical functions, 48
    - using with lists, 161
    - with a matrix, 185
  - matrix, 29
    - brackets [ ], 180, 319
    - creating, 178, 180
    - defined, 178
    - deleting from memory, 180
    - displaying elements, rows, submatrices, 181
    - editing using  $\overrightarrow{\text{STO}}$ , 182
    - names, 43
    - using in expression, 181
    - using math functions, 185
  - Matrix Editor menu, 179
  - matrix entry [ ], 319
  - MATRX (matrix names), 43
  - MATRX (Matrix) menu, 178
  - MATRX CPLX (Complex) menu, 185



- MATRX MATH menu, 183  
 MATRX NAMES menu, 178  
 MATRX OPS (Operations)  
   menu, 184  
 max(, 49, 160, 319  
 maximum characters, 22  
 maxX, 193  
 maxY, 193  
 MBox, 319  
 Med (median), 193  
 MEM (clear memory), 232  
 MEM (Memory) menu, 29, 230  
 MEM RESET menu, 232  
 MEM DELET (Delete) menu,  
   231  
 MEM FREE (available  
   memory), 230  
 memory, 16, 17, 22, 28, 29, 223  
   available, 230  
   deleting items, 231  
   resetting, 3, 232  
 memory backup  
   initiating, 237  
   overwrite warning, 237  
 menus  
   displaying, 31  
   exiting, 6  
   menus (continued)  
     in editors, 33  
     keys, 32  
     lower, 32  
     removing, 6, 33  
     selecting items, 32  
     upper, 33  
   menu map, 380  
   Menu(, 219, 320  
   min(, 49, 160, 320  
   minX, 193  
   minY, 193  
   mod(, 49, 320  
 mode settings, 19, 20, 70  
   changing, 34  
   displaying, 34  
   number base, 65  
 modulo, 49  
 mRAdd, 184  
 mRAdd(, 321  
 multiple entries  
   retrieving, 29  
 multiplication (\*), 321  
 multR( (multiply row), 184,  
   322  
   type, designating, 67  
   modes, 35  
 numbers  
   entering, 19  
 numeric differentiation, 36  
 numerical derivative, 54
- N**  
 n (statistical results variable), 193  
 natural log, 48  
 nCr (number of combinations),  
   50, 322  
 nDer( (numerical derivative),  
   54, 323  
 negation symbol (-), 20  
 negative numbers  
   entering, 19  
 norm, 173, 183, 323  
 Normal, 34, 324  
 not (Boolean), 66, 69, 325  
 not equal to ( $\neq$ ), 326  
 notation modes, 34  
   engineering, 34  
   normal, 34  
   scientific, 34  
 notation of displayed answers,  
   20  
 nPr (number of permutations),  
   50, 326  
 number base, 65  
   designators, 65  
   ranges, 66
- O**  
 o, 326  
 Oct (octal), 35, 327  
 ►Oct (to octal), 327  
 octal integer, 326  
 octal number base, 35, 66  
 OneVa (OneVar), 189, 191, 327  
 operation  
   second, 22  
 operator  
   entering, 25  
 or (Boolean), 69, 328  
 order of operations, 56  
 order-of-evaluation rules, 20, 62  
 Outpt(, 217, 329  
 OVERW (overwrite), 241

- P**
- P2Reg (quadratic regression),  
190, 330
  - P3Reg (cubic regression), 190,  
331
  - P4Reg (quartic regression),  
190, 332
  - panning, 90
  - Par, 74
  - Param (parametric mode), 35,  
239, 333
  - parametric equation
    - deleting, 127
    - graphing, 126
    - selecting and deselecting,  
127
  - parametric graphs, 74
    - default graph style, 126
    - defining, 125
    - displaying, 128
    - drawing, 130
    - equation editor, 126
    - free-moving cursor, 128
    - graph format, 128
    - graph tools, 128
    - mode, 35, 126
    - tracing, 128
    - window variables, 127
    - Zoom, 129
  - parentheses, 20, 25, 56, 61, 397
  - pause, 26, 333
  - Pause (PRGM CTL menu), 219
  - pause indicator, 26
  - PC
    - linking to, 235
  - PEN, 105
  - percent (%), 334
  - permutations of items, 50
  - pEval(, 52, 334
  - phone (TI Customer Support),  
392
  - pi, 59
  - PIC (picture names), 43
  - PIC variable
    - entering, 76
    - storing graph, 102
  - pictures
    - recalling, 102
    - saving, 102
  - pixel resolution
    - for function graphs, 81
  - PIOff (plot off), 195, 334
  - PIOn (plot on), 195, 334
  - PLOT1, 195
  - Plot1(, 335
  - PLOT2, 195
  - Plot2(, 335
  - PLOT3, 195
  - Plot3(, 335
  - plotting functions, 9, 11
  - plotting statistical data, 194
  - points
    - drawing, 108
    - turning on and off, 108
  - Pol (polar mode), 35, 74, 239,  
336
  - Pol (to polar), 71, 174, 336
  - polar angle of complex number,  
72
  - polar complex ( $\angle$ ), 336
  - polar complex mode, 35, 336
  - polar complex number form,  
20, 70
  - polar equation
    - tracing, 120
  - polar graphs, 74, 84
    - default graph style, 118
    - defining, 117
    - displaying, 119
    - drawing, 122
    - equation editor, 118
    - free-moving cursor, 119
    - graph format, 119
    - graph tools, 119
  - polar graphs (continued)
    - mode, 35
    - trace cursor, 120, 121
    - tracing, 120
    - window editor, 118
    - Zoom, 121
  - PolarC (polar complex mode),  
35, 336
  - PolarGC (polar graph  
coordinates), 84, 336
  - poly, 337
  - polynomial coefficient
    - storing to a variable, 212
  - polynomial root
    - storing to a variable, 212
  - polynomial root-finder, 211
  - polynomial value, 52
  - power of 10 ( $10^{\wedge}$ ), 20, 34, 337
  - PRegC, 193
  - previous entries, 8
    - re-executing, 19
    - retrieving, 28

- reusing, 28
  - PRGM (program names), 43
  - PRGM CTL menu, 218
  - PRGM I/O (Input/Output) menu, 215
  - PRGM menu, 214
  - prod (product), 52, 160, 338
  - program editor, 214
    - menus and screens, 215, 220
  - program flow, 56
  - programming
    - assembly language, 225
    - calling a program, 224
    - copying a program, 225
    - creating programs, 214
    - defined, 214
    - deleting a program, 223
    - downloading assembly programs, 225
    - editing a program, 223
    - entering a command line, 220
    - getting started, 214
    - interrupting program, 222
    - running program, 221
    - using variables, 225
  - Prompt (PRGM I/O menu), 216, 338
  - prompts, 22
    - Eval x=, 76
    - Name=, 22, 39, 76
    - Rcl, 42
    - Sto, 212
  - PTCHG, 105
  - PtChg(), 338
  - PTOFF, 105, 108
  - PtOff(), 338
  - PTON, 105, 108
  - PtOn(), 338
  - PwrR (power regression), 190, 339
  - PxChg(), 103, 340
  - PxOff(), 103, 340
  - PxOn(), 103, 340
  - PxTest(), 103, 340
- Q**
- Q'n equation variables, 135
  - Qrt11, 193
  - Qrt13, 193
  - Quick Zoom, 91
    - in parametric graphing, 129
    - in polar graphing, 120
- Quick-Find Locator (A to Z Reference), 262
- R**
- r (radian entry), 341
  - rAdd, 184
  - rAdd(), 340
  - Radian (angle mode), 35
  - radian angle mode, 75, 341
  - radian complex-number mode, 70
  - radian entry (r), 341
  - rand (random), 50, 341
  - randBin( (random binomial), 50, 341
  - randInt( (random integer), 50, 342
  - randM( (random matrix), 184, 342
  - randNorm( (random normal), 50, 342
  - random number, 50
  - RCGDB (recall graph database), 76, 88, 343
  - RcPic (recall picture), 76, 102, 343
  - RCPIC menu, 76
  - REAL, 43, 175, 185, 343
  - REAL (to real number), 156, 170, 179
  - real number variables, 43
  - real numbers, 29
  - real portion of complex number, 71
  - Rec (to rectangular), 71, 174, 343
  - recalling variable values, 18, 42
  - receiving transmitted data, 241
  - rectangular complex mode, 35
  - rectangular complex numbers, 70
  - rectangular complex-number form, 20
  - rectangular graph, 84
  - rectangular vector coordinates, 36
  - RectC (rectangular complex), 35, 344
  - RectGC (rectangular graph coordinates), 84, 344
  - RectV (rectangular vector coordinate mode), 36, 344
  - RECV (LINK menu), 236

- RECV (LINK SND85 menu), 240  
redefining user-created  
  constants, 60  
ref (row echelon form), 184,  
  344  
regression models, 191  
relational functions, 55, 56  
RENAM (rename), 241  
Repeat (PRGM CTL menu),  
  218, 345  
replacing batteries, 16  
resetting memory, 232  
result, 20, 24  
result of last expression, 26  
Return (PRGM CTL menu), 219,  
  345  
RK (Runge-Kutta) method, 133,  
  345  
norm (row norm), 183, 346  
ROOT, 96, 97  
   $x\sqrt{\phantom{x}}$ , 346  
root-finder, 211  
RotL (rotate left), 69, 347  
RotR (rotate right), 69, 347  
round(, 49, 176, 348  
row  
  of matrix, 181  
  ref (reduced row echelon),  
    184, 348  
rSwap( (row swap), 184, 348  
running a program, 221
- S**  
Scatter (stat plot type), 349  
Sci (scientific notation), 20, 34,  
  349  
scrolling, 19  
seed value, 50  
SELECT, 112  
SELECT, 77  
Select(, 161, 350  
selection cursor, 38  
SEND (LINK menu), 236  
SEND WIND screen, 238  
Send(, 216, 350  
separator, 70  
seq( (sequence), 52, 160, 351  
SeqG (sequential graphing), 84,  
  351  
series of instructions  
  displaying, 18  
SetLE, 159  
SetLEdit, 161, 351  
  setting graph format, 83  
  setting graph style, 80  
Shade(, 103, 104, 352  
shading  
  pattern, 104  
  resolution, 104  
shading patterns, 80  
ShftL (shift left), 69, 353  
ShftR (shift right), 69, 353  
ShwSt (show string), 354  
sign, 49, 354  
SimulG (simultaneous  
  graphing), 84, 354  
SIMULT ENTRY menu, 208  
SIMULT order screen, 208  
SIMULT RESULT menu, 209  
simult(, 210, 354  
simultaneous equation solver,  
  208  
sin (sine), 48, 186, 355  
sin<sup>-1</sup> (arcsine), 48, 355  
sine  
  calculating, 3  
sinh (hyperbolic sine), 51, 356  
sinh<sup>-1</sup> (inverse hyperbolic sine),  
  51, 356  
SinR (sinusoidal regression),  
  190, 193, 357  
SKIP, 241  
SlpFld (slope field), 134, 358  
Smart Graph, 86  
  drawing tools, 102  
  in GRAPH MATH, 95  
  in GRAPH ZOOM, 94  
SND85 (LINK menu), 236  
solution method formats, 133  
solutions  
  drawing, 148  
SOLVE, 205  
solver graph, 207  
Solver menu, 206  
Solver ZOOM menu, 208  
Solver(, 358  
solving differential equations,  
  139  
solving for unknown variable,  
  206  
sortA, 159, 359  
sortD, 159, 359  
Sortx, 160, 359  
Sorty, 160, 359  
►Sph (to spherical), 174, 360

- SphereV (spherical vector coordinate mode), 36, 360
- square ( $\square$ ), 360
- square root ( $\sqrt{\quad}$ ), 7, 360
- StbEq (string to equation), 227, 361
- STAT (statistical result variables), 43
- STAT CALC (Calculations) menu, 189
- STAT menu, 188
- Stat Plot
  - changing on/off status, 81
  - setting up, 195
  - turning on and off, 195
- STAT PLOT menu, 195
- STAT PLOT status screen, 194
- STAT VARS (Statistical Variables) menu, 192
- statistical analysis, 188
  - results, 192
- statistical data
  - entering, 189
  - plotting, 194, 195
- STGDB (store graph database), 76, 88, 361
- STOa, 210
- STOb, 210
- Stop, 219, 362
- Store, 18
- store symbol, 22
- store to variable ( $\rightarrow$ ), 362
- storing a graph display, 102
- storing data, 39
- storing equation coefficients, 210
- storing equation results, 210
- STOx, 210
- STPIC (store picture), 76, 88, 362
- STPIC menu, 76
- StReg (store regression equation), 190, 362
- string, 29
  - concatenating, 226
  - creating, 226
  - defined, 226
  - storing, 226, 227
- string entry, 363
- STRNG (string variables), 43
- STRNG (String) menu, 227
- STYLE, 77
- sub( (subset of string), 227, 363
- submatrix
  - displaying, 181
- subroutines, 224
- subtraction ( $-$ ), 363
- sum, 52, 160, 364
- sum of elements of list, 52
- Sx (statistical result variable), 193
- syntax error, 27
- syntax of function, 25
- syntax of instruction, 25
- T**
- T (transpose), 367
- table, 110
  - clearing, 114
  - displaying, 110
  - navigating, 111
  - setting up, 113
  - setup editor, 113
- TABLE menu, 110
- Table menus, 112
- table setup editor, 113
- tan (tangent), 48, 364
- $\tan^{-1}$  (arctangent), 48, 365
- tangent line
  - drawing, 107
- tanh (hyperbolic tangent), 51, 365
- $\tanh^{-1}$  (inverse hyperbolic tangent), 51, 365
- TANLN (tangent line), 96, 99
- TanLn(, 103, 107, 366
- TBLST (table setup editor), 112, 113
- TEST menu, 55
- TEXT, 105
- Text(, 366
- Then, 218, 305, 306
- TI-GRAPH LINK, 235
- tMax, 127, 136
- tMin, 127, 136
- TOL (Tolerance Editor), 398
- tPlot, 136
- TRACE, 88
- TRACE (cursor), 75
- TRACE (Graph menu), 367
- TRACE (Solver menu), 207
- trace cursor, 75, 90, 144, 205
  - in parametric graphing, 128
  - in polar graphing, 120
  - moving, 90, 121, 129
  - panning, 90
  - Quick Zoom, 91

stopping and resuming, 91  
tracing a function, 11  
transmitting data, 234, 240  
    error conditions, 242  
    insufficient memory, 242  
transmitting data (continued)  
    repeating to several devices,  
        242  
    selecting variables, 238  
    window variables, 239  
transpose ( $\overline{\phantom{x}}$ ), 367  
tStep, 127, 136, 138  
turning off TI-86, 2, 17  
turning on TI-86, 2, 17  
TwoVa (TwoVar), 189, 368

## U

unevaluated expression  
    storing, 9, 40  
units of measure  
    converting, 61  
unit-to-unit cable, 234, 235  
unitV (unit vector), 173, 368  
unknown variable  
    solving for, 206  
upper menu, 32

    selecting an item, 33  
user-created constants, 43, 58,  
    60  
user-created zoom variables,  
    239

## V

value, 24, 25, 29  
variable, 21  
    classifying as data types, 42  
    copying, 41  
    creating, 39  
    deleting, 45  
    displaying, 41  
    in expressions, 4  
    in table screen, 111  
    names, 44  
    recalling, 42  
    storing data to, 39  
    storing results to, 3, 30  
    uppercase and lowercase  
        names, 39  
    x variable, 77  
    y variable, 77  
variable equations  
    in a table, 114  
VARS CPLX (complex  
    variables) screen, 71

VARS EQU menu, 203  
vOli (vector to list), 160, 174,  
    369  
vector, 29  
    brackets [ ], 369  
    complex, 171, 180  
    creating, 170  
    defined, 168  
    deleting from memory, 170  
    displaying, 171  
    editing dimension and  
        elements, 172  
    forms, 168  
    operations, 173  
    using in an expression, 172  
    with math functions, 176  
vector coordinate modes, 36  
vector editor, 168  
Vector Editor menu, 170  
vector entry [ ], 369  
VECTR (vector names), 43  
VECTR CPLX (Complex) menu,  
    175  
VECTR MATH menu, 173  
VECTR menu, 169  
VECTR NAMES menu, 169

VECTR OPS (Operations)  
    menu, 173  
VERT (vertical line), 104, 106,  
    369

## W

warranty information, 400, 402  
While, 218, 369  
WIND (Solver menu), 206  
WIND (window variables), 43,  
    35, 75, 238  
window editor, 75  
    polar, 118  
window variables, 82  
     $\Delta x$  and  $\Delta y$ , 83  
    changing, 12, 82  
    differential equations, 135  
    graph screen, 81

## X

x variable, 77  
XMIT (transmit), 237, 240  
Xor (Boolean), 69, 370  
xRes (resolution), 81  
xScl (scale), 81  
xStat (x-variable list), 189

xyline, 370

## Y

y variable, 77

$y(x)=$ , 75

YICPT (y-intercept), 96, 100

yScl (scale), 81

yStat (y-variable list), 189

## Z

ZData, 371

ZDATA (GRAPH ZOOM menu),

92

ZDecm, 372

ZDECM (GRAPH ZOOM menu),  
92

ZFACT (ZOOM FACTOR), 92,  
208

ZFit, 129, 373

ZFIT (GRAPH ZOOM menu), 92

ZIn (zoom in), 373

ZIN (zoom in), 92, 208

ZInt, 374

ZINT (GRAPH ZOOM menu), 92

ZOOM, 14, 75, 88

custom, 93

parametric graphs, 129

polar graphs, 121

ZOOM operations, 147

zoom window variables

storing and recalling, 95

ZOOMX (GRAPH ZOOM menu),  
92

ZOOMY (GRAPH ZOOM menu),  
92

ZOUT (zoom out), 92, 208, 375

ZPREV (zoom previous), 92,  
375

ZRCL (GRAPH ZOOM menu),  
92, 95

user-created zoom variables,  
239

ZRcl (zoom recall), 376

ZSqr, 376

ZSQR (GRAPH ZOOM menu),  
92

ZSTD (GRAPH ZOOM menu),  
92

ZSTD (standard defaults), 208,  
377

ZSTO (GRAPH ZOOM menu),  
92, 95

ZTrig, 378

ZTRIG (GRAPH ZOOM menu),  
92