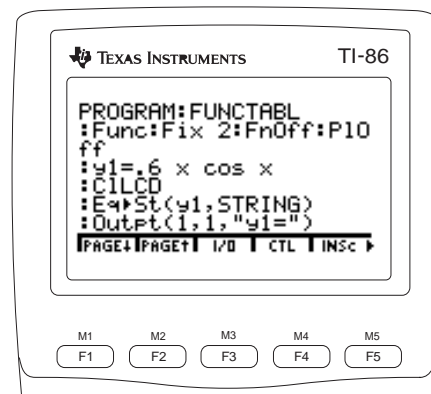


16 Programming

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

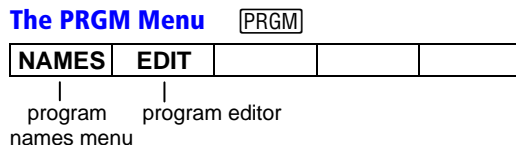


Writing a Program on the TI-86

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**

PAGE↓	PAGE↑	I/O	CTL	INSc	▶	DELc	UNDEL	:		
page down	page up	input/output menu	program control menu	insert a blank command line		delete (cut) a command line	undelete (paste) a deleted command line			

The PRGM I/O (Input/Output) Menu **PRGM** **F2** *programName* **ENTER** **F3**

PAGE↓	PAGE↑	I/O	CTL	INSc	▶	CITbl	Get	Send	getKy	CILCD
Input	Prompt	Disp	DispG	DispT						
					▶	"	Outpt	InpSt		

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).

Input**Input** *variable*

Displays the current graph and lets you use the free-moving cursor

Pauses a program, displays ? as a prompt, and then stores your response to *variable***Input** *promptString,variable*Pauses a program, displays *promptString* or *string* (up to 21 characters) as a prompt, and then stores your response to *variable***Input** "string",*variable*Although using **Get**(is preferred on the TI-86, you can use **Input** to receive *variable* from a CBL 2/CBL, CBR, or TI-86 (TI-85 compatible)**Input** "CBLGET",*variable*Displays each *variable* with ? to prompt you to enter a value for that *variable***Prompt** *variableA*[,*variableB,variableC,...*]

Displays the home screen

DispDisplays each *value***Disp** *valueA,valueB,...*Displays the value stored to each *variable***Disp** *variableA,variableB,...*Displays each *text* string on the left side of the current display line**Disp** "textA","textB",...

Displays the current graph

DispG

Displays the current table and temporarily halts the program

DispTClears the current table if **Indpnt: Ask** is set (Chapter 7)**CITbl**Gets data from a CBL 2/CBL, CBR, or another TI-86 and stores it to *variable***Get**(*variable*)Sends the contents of *listName* to a CBL 2/CBL or CBR**Send**(*listName*)

Returns a number corresponding to the last key pressed, according to the key code diagram (page 217); if no key was pressed, returns 0

getKey

Clears the home screen (LCD stands for liquid crystal display)

CILCD

"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*

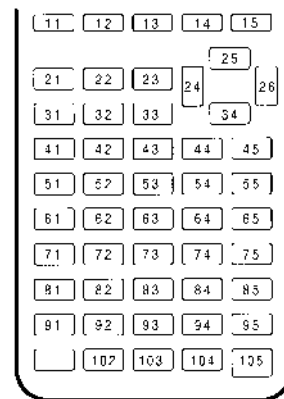
The TI-86 Key Code Diagram

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

Menu(*item#*,"*title1*",
label1 [*item#*,
"*title2*",*label2*,...])

Lbl *label*

Goto *label*

IS>(*variable*,*value*)

DS<(*variable*,*value*)

Pause

Pause *value*

Return

Stop

tested when the **End** instruction is encountered

Sets up branching within a program as selected from menu keys **[F1]** through **[F5]**; when encountered, displays the first of up to 3 menu groups (up to 15 *titles*); when you select a *title*, the program branches to the *label* that the *title* represents; *item#* is an integer ≥ 1 and ≤ 15 that specifies *title*'s menu placement; *title* is a text string from one to eight characters long (may be abbreviated in the menu)

Assigns a *label* to a program command; label can be one to eight characters long, starting with a letter

Transfers control to the program branch labeled with *label*

Adds 1 to *variable*; if the answer is $> value$, the next command is skipped; if the answer is $\leq value$, the next command is executed; *variable* cannot be a built-in variable








Subtracts 1 from *variable*; if the answer is $< value$, the next command is skipped; if the answer is $\geq value$, the next command is executed; *variable* cannot be a built-in variable

Halts the program so that you can examine results, including displayed graphs and tables; to resume the program, press **[ENTER]**

Displays *value* on the home screen so that you can scroll large values, such as lists, vectors, or matrices; to resume, press **[ENTER]**

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 **Return** exits each subroutine upon completion and returns to the calling program)



Stops a program and returns to the home screen

DelVar (<i>variable</i>)	Deletes from memory <i>variable</i> (except program names) and its contents
GrStl (<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 5 in y5 ; <i>graphStyle#</i> is an integer ≥ 1 and ≤ 7 , where 1 =  (line), 2 =  (thick), 3 =  (shade above), 4 =  (shade below), 5 =  (path), 6 =  (animate), and 7 =  (dotted)
*LCust (<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 CUSTOM ; <i>item#</i> is an integer ≥ 1 and ≤ 15 ; <i>title</i> is a string with one to eight characters (may be abbreviated in the menu)

Entering a Command Line

A command line that is longer than the screen is wide automatically continues at the beginning of the next 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

- ❶ Paste the program name to the home screen. Either select it from the PRGM NAMES menu (PRGM F1) or enter individual characters.
- ❷ Press **ENTER**. The program begins to run.

*To resume the program after a pause, press **ENTER**.*

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

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 y1 into the string variable STRING (STRNG menu)
:Outpt(1,1,"y1=")	Display y1= at row 1, column 1 (PRGM I/O menu)
:Outpt(1,4,STRING)	Display value stored to STRING at row 1, col. 4 (PRGM I/O menu)
:Outpt(8,1,"PRESS ENTER")	Display PRESS ENTER 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 y2 as the first derivative of y1 (CALC menu)
:y3=der2(y1,x,x)	Define y3 as the second derivative of y1 (CALC menu)
:DispT	Display the table (PRGM I/O menu)
:GrSt1(1,1):GrSt1(2,2)	Set graph styles for y1 , y2 , and y3 (PRGM CTL menu)
:GrSt1(3,7)	
:2→xRes	Store 2 to the window variable xRes (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.

Calling program

```
PROGRAM:VOLCYC
:Input "D=",D
:Input "H=",H
:AREACIR
:A*H→U
:Disp U
```

Subroutine

```
PROGRAM:AREACIR
:D/2→R
:π*R²→A
:Return
```

Input/Output

```
VOLCYC
D=4
H=5
62.8318530718
Done
```

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 ([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 [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 (<i>AsciiAssemblyPrgmName</i> , <i>HexAssemblyPrgmName</i>)	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 **+**. The syntax is:

"stringA"+"stringB"+"stringC"+...

The STRNG (String) Menu 2nd [STRNG]

"	sub	lngh	Eq▶St	St▶Eq
---	-----	------	-------	-------

" 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)

lngh "string" or **lngh** stringName

Eq▶St(equationVariable,stringName)

St▶Eq(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

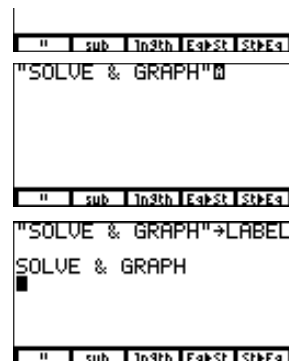
Begin these steps on a blank line on the home screen or in the program editor.

- 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] [↵]
2nd [CHAR] [F1] [F3] [↵]
[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 **St▶Eq** to convert it to an equation.

