



**ST0▶ key**

$$list \Rightarrow var$$
 $\pi/4 \rightarrow \text{myvar}$  
$$\frac{\pi}{4}$$
$$matrix \rightarrow var$$
$$2\cos(x) \rightarrow Y1(x) \quad \boxed{\text{ENTER}}$$

Done

$$expression \rightarrow fun\_name(parameter1, \dots)$$

$\{1,2,3,4\} \rightarrow \text{Lst5}$  **ENTER**

 $\{1 \ 2 \ 3 \ 4\}$ 

```
list → fun_name(parameter1,...)
```

*matrix*  $\rightarrow$  *fun name(parameter1,...)*

If variable *var* does not exist, creates *var* and initializes it to *expression*, *list*, or *matrix*.

$[1,2,3;4,5,6] \rightarrow \text{MatG}$  **ENTER**  $\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix}$

If *var* already exists and if it is not locked or protected, replaces its contents with *expression*, *list*, or *matrix*.

```
"Hello"→str1 [ENTER]
```

"Hello"

**Hint:** If you plan to do symbolic computations using undefined variables, avoid storing anything into commonly used, one-letter variables such as `a`, `b`, `c`, `x`, `y`, `z`, etc.

**Program Editor/Control menu** on

**TI-89:**   **key**

**TI-92 Plus:**  $\boxed{2^{nd}}$  X key

© [text]

Program segment:

- processes *text* as a comment line, which can be used to annotate program instructions.

```

:
:
: Get 10 points from the Graph

```

• can be at the beginning or anywhere in the line. Everything to the right of •, to the end of the line, is the comment.

```
screen
:For i,1,10
times
:
```

**TI-89:**   [B] keys    **TI-92 Plus:**  B keys

**TI-89: [0] [alpha] [B] keys    TI-92 Plus: [0] B keys**

**TI-89:**  $\boxed{0}$   $\boxed{\alpha}$   $\boxed{H}$  keys    **TI-92 Plus:**  $\boxed{0}$  H keys

**0b** *binaryNumber*

In Dec base mode:

## 0h hexadecimalNumber

0b10+0hF+10 [ENTER]

27

In Bin base mode:

0b10+0hF+10 [ENTER]

0b11011

Denotes a binary or hexadecimal number, respectively. To enter a binary or hex number, you must enter the 0b or 0h prefix regardless of the Base mode. Without a prefix, a number is treated as decimal (base 10).

In Hex base mode:

0b10+0hF+10 

0h1B

Results are displayed according to the Base mode.

## P (continued)

### point

- change, **PtChg**, 307 482
- off, **PtOff**, 307 483
- on, **PtOn**, 307 483
- test, **ptTest()**, 307 483
- text, **PtText**, 307 483

### polar

- coordinate, **R>Pθ()**, 487
- coordinate, **R>Pr()**, 487
- graphing, 133–138
- vector display, **►Polar**, 480

**polyEval()**, evaluate polynomial, 480

polynomials, 9 72, 76

- activity, 402
- evaluate, **polyEval()**, 480
- random, **randPoly()**, 488

**PopUp**, popup menu, 301, 481

power of ten, **10^()**, 537

power, **^**, 534 569

**PowerReg**, power regression, 262, 481, 571

pretty print, 6, 11, 23, 29

Pretty Print mode, 29, 41, 552

previews. *See* examples, previews, activities

**Prgm**, execute program, 276, 287, 481

prime number test, **isPrime()**, 459

prime numbers, 8

prime, **'**, 536

problems (new), **NewProb**, 43, 472

problems in operation. *See* errors and troubleshooting

product code, upgrading, 373, 374

product ID, 55

**product()**, product, 482

product, **Π()**, 75, 533

programs and programming, 275–314

- arguments, 284
- assembly language, 313, 314
- branching, 283, 295, 296
- calling another program, 287
- CBL 2/CBL, 309, 399
- CBR, 309, 399
- clear error, **ClrErr**, 310 420
- clear graph, **ClrGraph**, 205 305, 420
- clear home, **ClrHome**, 421
- clear I/O, **ClrIO**, 279 302 421
- clear table, **ClrTable**, 421
- comment, **⓪**, 282 539
- conditional tests, 294
- copying, 281
- custom toolbar off, **CustmOff**, 37 302 428
- custom toolbar on, **CustmOn**, 37 302 428
- debugging, 310
- define dialog box **Dialog**, 302 437
- define toolbar, **Custom**, 302 429
- define toolbar, **Toolbar**, 302 515
- define, **Define**, 287 305 384 433

deleting, 281

display graph, **DispG**, 302 305 438

display Home screen, **DispHome**, 302 438

display I/O screen, **Disp**, 277 283 302 310 437 555

display table, **DispTbl**, 302 305 438

drop-down menu, **DropDown**, 302 440

else if, **Elseif**, 207 296 442

else, **Else**, 296 456

end custom, **EndCustm**, 302 429

end dialog, **EndDlog**, 302 437

end for, **EndFor**, 283 297 450

end function, **EndFunc**, 207 286 451

end if, **EndIf**, 283 295 296 456

end loop, **EndLoop**, 299 466

end program, **EndPrgm**, 276 287 481

end toolbar, **EndTBar**, 302 515

end try, **EndTry**, 310 515

end while, **EndWhile**, 298 518

entering, 280, 281, 282, 283

execute assembly language, **Exec**, 314 444

execute program, **Prgm**, 276 287 481

exit, **Exit**, 444

for, **For**, 283 297 450

format string, **format()**, 302 450

function, **Func**, 207 286 451

functions, 280, 285, 286

get/return configuration, **getConfig()**, 300 452

get/return folder, **getFold()**, 300 453

get/return from calculator, **GetCalc**, 309 371 452

get/return key, **getKey()**, 301 453 556 559

get/return mode, **getMode()**, 300 453

get/return units, **getUnits()**, 300 454

go to, **Goto**, 287 296 299 455

graphical user interface, GUI, 302

graphs, 305

if, **If**, 207 283 295 296 456

input, 279, 283, 301

input, **Input**, 301 305 457

label, **Lbl**, 287 296 299 459

local, **Local**, 286 288 289 290 464

loop, **Loop**, 299 466

looping, 283, 297, 298

menu item, **Item**, 302 303 459

menus, 303, 304

multicommand lines, 282

operations, 412

output, 279, 283, 301, 302

output, **Output**, 302 476

pass error, **PassErr**, 310 479

passing values, 284

pause, **Pause**, 302 310 479

popup menu, **PopUp**, 301 481

prompt, **Prompt()**, 301 482

request, **Request**, 301 302 490