

*Programming  
Your  
TI - 83, 84, or TI - 83plus*

Manual originally written for the TI-81 by Sarah Hayes, Dan McGuire, David Matthews, Elizabeth Blevins, and Robert Howard. Maintained and upgraded for the TI-83 and 84 by Steve Sigur and many of his students.



# Programming your Calculator

TIPS: In order to change the contrast of the screen

To lighten: press the 2nd key and press the down arrow

To darken: press the 2nd key and press the up arrow.

When subtracting, be sure to use the  $-$  button. When using negative numbers, be sure to use the  $(-)$  button.

To space when in Alpha mode, press zero. This is helpful when writing words in a program.

The last page of this handout has a nice diagram of the calculator menus for you.

## Things to do to get ready for your first program

Your first program will compute a random digit less than 10. Here are a few warm ups to get you used to the new calculator commands we will use.

Step 1, the random number command, **Rand**:

Go to the **MATH** button on your calculator. Move the cursor to the right using the right arrow until you get to **PRB**, the probability menu. Choose option 1 and press enter once. You will now see that the **rand** command is now on the home screen of your calculator. Push enter many times, each time produces a different decimal random number between 0 and 1.

Step 2: Now enter **rand \* 6**, where the "\*" means multiplication. Push enter many times to see that now the random numbers are between 0 and 6.

Step 3: Now we usually do not want all those decimals, we get rid of them with the command **IPart**, which is shorthand for "integer part." We find **IPart** under the **MATH** menu after we have moved the cursor over to **NUM**, which is short for "numbers."

Now enter this formula **IPart( rand \* 6 )**. Hit enter many times. You now get random integers from 0 to 6.

Summary: in the command **IPart( rand \* 6 )**, **IPart** makes the expression into an integer, **rand** creates a random number, and the **\* 6** makes the numbers go from 0 to 6. Every command has its purpose; it just makes sense.

---

### Your first program:

This program creates a random number between one and ten.

After turning the calculator on, press the **PRGM** key. Use the right arrow to go to the **NEW** submenu. Press enter. Now the calculator is waiting for you to name the program. The cursor has an A in it, indicating it is already in Alpha lock which allows you to write in letters. Name the program **RANDOM**.

Now you are at the first line of the program. This program that you will enter has two lines. Follow the instructions to the right to enter the various commands.

```
: IPart (10*rand) => X  
: Disp X
```

The program is now finished. To run the program, **QUIT** the editor. To see your program work go back to **PRGM**. Go to **EXEC** and press the number of your program. This puts the name of your program on the home screen. Press the **ENTER** button to run the program.

Keep pressing enter to get as many random numbers as you wish.

#### How to Key in This Program

First line: Don't type in the word **IPart**. Press **MATH** and use the right arrow key to go the **NUM** menu. Choose the second choice **IPart**. Now key in **(10\***. The star means multiplication. In order to get **rand**, go back to **MATH**; go to the submenu **PRB**. The first choice is **rand**. Close the parentheses. Press **STO** on the front of the calculator to get the arrow. Type **ALPHA X** to get the x. On completing a line, press **ENTER** to go to the next line.

Second line: Press **PRGM** button and go to **I/O**. **Disp** is the first option. Then press **ALPHA**, then **X**.

---

To change or edit a program, push the **PRGM** button, use the right arrow to choose **EDIT**, select the program **RANDOM** then put the cursor over the 10. The **DEL** button deletes any character or command you do not wish.

Questions: 1. After you have pushed enter many times, you should get many "random" numbers. Describe their pattern. What numbers do you get? What don't you get?  
2. What happens if you change the 10 in the program to a 6? Think; conjecture; then do.

## Dice

This program is intended to simulate the rolling of two dice. With this program, you can not only see the numbers rolled on the dice, but also get the sum of the numbers rolled.

Like the last program, this program uses the Rand command to generate a random number. A new feature uses a logical test to tell the user if they rolled doubles.

### Prgm2: DICE

```
: IPart (Rand*6) + 1
: Ans ➡ A
: Disp Ans
: IPart (Rand*6) + 1
: Ans ➡ B
: Disp Ans
: A+B ➡ C
: Disp "SUM OF NUMBERS ="
: Disp C
: If A = B
: Disp "YOU HAVE ROLLED DOUBLES"
```

## Slope and y intercept: a program that uses formulas to compute numbers

Given two points (A, B) and (C, D) the program computes the slope  $m$  and the y intercept  $b$ , the slope and y intercept of the line. The new command in this program is the "Input" command. It allows the user to enter numbers of their choice into the program.

```
: Prgm3: ST LINE
: Disp "ENTER FIRST POINT"
: Input A
: Disp "ENTER B"
: Input B
: Disp "ENTER C"
: Input C
: Disp "ENTER D"
: Input D
:
: ((D-B)/(C-A)) ➡ M
: B - M * A ➡ Z
:
: Disp "M ="
: Disp M
: Disp "B ="
: Disp Z
```

Comments on this Program  
some commands explained in last program.

The command **Disp** found under the **I/O** menu after pushing the **PRGM** button.

**Ans** is found on above the (-) key at the bottom on the front of the calculator.

**A** is the value of the first "die." The program stores this number so it can be used later. **B** is the value of the second.

**IPart (Rand\*6) + 1** produces a random integer between 1 and 6. It is used twice to produce the two random numbers. (Note: the \* means multiply)

Notice the use of **Disp** to display text output using quotation marks. The "=" is found in the **TEST** menu.

The command **If** is also found under the **PRGM** button. Notice the use of the **If A = B** statement to find out if the two rolls are equal. If yes, the program informs you that doubles have been rolled. If no, the program skips to the next line which is the end of the program.

### Question2 for Program 2:

1. What would happen if you changed the two numbers 6 to 8's?

Comments on Program 3

The **Input** command is found by pressing **PRGM** then moving the cursor right to **I/O** which is shorthand for "Input/Output." This command causes the program to display a "?" on the users screen. The program pauses, waiting for the user to enter a number. This number is then the value of the named variable. The **Disp** command often appears just before **Input** to display text explaining what is to be entered.

These two lines compute the slope and the y intercept. The y intercept is stored as the variable **Z** since **B** is already used in the program. The user is never aware of this.

The last 4 lines display the slope and the y intercept.  
(note: the \* means multiplication)

**Question 1:** Find the equation of the line through points (5,6) and (1,2).

**Question 2:** Find the equation of the line through (-3, -4) and (12, -3).

**Question 3:** Find the equation of the line through (6000, 8396) and (3000, 4567).

## Directing Traffic, Learning the *If* Command

The *If* command allows your program to make a choice. After the *If* command is a logical **TEST**, using the "=" sign found in the **TEST** menu. The test is a statement that can be true or false. If the test is true, the program goes to the next line. If false, the program skips a line. The *If* command is followed by a **Goto** command directing the program to a certain label. Here is a simple example.

### Prgm4: FLOW

```
:Input "ENTER A NUMBER", N
: If N = 7
: Goto 1
: Goto 2
: Lbl 1
: Disp "THIS IS A 7"
: Stop
: Lbl 2
: Disp "THIS IS NOT A 7"
: Stop
```

## Quadratic Formula

This program solves the quadratic equation  $y = ax^2 + bx + c$  when you enter the three coefficients.

### Prgm5: QUAD FORMULA

```
: Disp "ENTER A, B, C"
: Prompt A, B, C

: If B^2 - 4AC < 0
: Goto 1
: (-B + √(B^2 - 4AC))/ (2A)
: Disp Ans
: (-B - √(B^2 - 4AC))/ (2A)
: Disp Ans
: Stop

: Lbl 1
: (B^2 - 4AC) ➡ Y
: -B/(2A)
: Disp Ans
: Disp "+ OR -"
: √-Y/(2A)
: Disp Ans
: Disp "I"
: Stop
```

### Comments on this Program

Line 1 asks the user for a number. Note that we have introduced a new feature of Input here. You can put the directions (a string of text, in quotes) before the letter of the variable (not in quotes).

Lines 3-5 test "is this number equal to 7?" If so then go to label 1, the first option which is directly below the test. If not go to label 2. When the computer goes to a **Lbl**, it executes commands until it sees an **Stop** command, whence it goes back to the *Goto* statement and executes from there.

The **Stop** command is way down at the bottom of the **PRGM** menu choices.

### Comments on this Program

The first 2 lines prompt the user to enter A, B, and C. This uses the **Prompt** command (in the **I/O** menu) which allows us to ask for more than one variable at once.

Lines 3-4 use an **if** statement to test if the discriminant is negative. If so the program goes to label 1, where the program handles imaginary numbers. The "<" symbol is found under the **TEST** menu.

Make sure you get the parentheses right. Also make sure you use the (-) button for the -B. Notice that  $-B - \sqrt{B^2 - 4AC}$  uses two different ways to input the "-" sign.

Lines 5-8 display the two answers for the case where the answers are real numbers.

Lines 11 - 18 display the answer when there are imaginary numbers in the answer.

The **Ans** command is found above the (-) key.

The \* indicates multiplication.

Questions for the Quadratic Formula program. It is very important that this program gives correct answers. Do these three Quadratic formula problems both by hand and on your calculator.

$$x^2 + 5x + 4 = 0$$

$$x^2 - 5x + 4 = 0$$

$$x^2 + x + 1 = 0$$

## Algorithm for $\sqrt{\quad}$

An algorithm is a procedure to compute a known result. This clever algorithm computes the square root of a positive number. The **Lbl** command marks a special place in the program. The **Goto** command tells the program which label to go to. We use the label here to make the program do the same procedure over and over. You first enter a positive number  $N$ , and then try to guess its square root. Not a good guesser? The program does not care; any positive guess will work.

### Prgm6: SQROOT

```
:Input "ENTER NUMBER",N
:Input "GUESS SQ ROOT",A
:Lbl 1
:N/A -> B
:(A+B)/2 -> A
:Disp A
:Pause
:Goto 1
```

## Modified Algorithm for $\sqrt{\quad}$

This time the algorithm does not pause but computes the square root of a positive number to an accuracy specified in the program (4 decimal places). The **If** statement tests how accurate our answer is. When it is accurate enough the program stops and prints it out.

### Prgm7: SQROOT

```
:Input "ENTER NUMBER",N
:Input "GUESS SQ ROOT",A
:Lbl 1
:N/A -> B
:(A+B)/2 -> A
:Disp A
:If abs(A-B) > .0001
:Goto 1
```

---

#### Comments on this Program

Line 1 asks the user for a number to take the square root of. You then follow with your guess (good or bad the program does not care, it will find the answer.).

Hopefully your teacher will explain how this works. It is a neat way to get a square root.

The **Pause** command is in the **PRGM** menu choices. We **Pause** the program so that you can see each step the program takes towards guessing the square root.

---

#### Comments on this Program

This version of the program replaces **Pause** with an **If** statement. The purpose of the **Pause** statement was a teaching one; we **Paused** so that you could see what numbers were being computed and that those numbers were getting closer to the square root you were computing.

The **If** statement lets the program make a decision, We want the program to go as many steps as it needs to compute the square root to a given accuracy, 4 decimal places in this case. It does this by testing how far apart our guesses are. When they are very close together, we know we are close to the answer. If we want a more accurate square root, we make the **If** statement more accurate.

The **If** command is in the **PRGM** menu choices.

The **abs** command is in the **MATH** menu. This finds the absolute value.

---

## Fred and Steve's game for the really bored

An algorithm is a procedure to compute a known result. This clever algorithm computes the square root of a positive number.

```
Prgm8: NICEGAME
:iPart(100*Rand) -> B
:Lbl 1
:ClrHome
:Disp "HOW DO YOU KEEP", "A BORED
:STUDENT", "AMUSED?", "PRESS ENTER"
:iPart(100*Rand) -> A
:Disp A
:Pause
:If A ≠ B
:Goto 1
:Disp "SORRY", "YOU ARE DONE"
```

---

## Fred's better Quadratic Formula

Since the answers that come from the quadratic equation can be complex numbers, Fred wrote a very nice a short version of the quadrati equation program

```
Prgm9: SUPRQUAD
:ClrHome
:a+bi
:Disp "AX2+BX+C=0"
:Prompt A,B,C
:Disp ( -B+√(B2-4AC))/(2A)
:Disp ( -B-√(B2-4AC))/(2A)
```

Comments on this Program

The ≠ is found in the **TEST** menu.

**ClrHome** is found in the **DRAW** menu.

A new feature of *Disp* is shown here, if quotes are separated by a comma, they are printed on a new line.

---

Comments on this Program

Line 1 clears the home screen. **ClrHome** is found in the **DRAW** menu.

Line 2 sets the calculator inot complex number mode; find **a+bi** under **MODE**. This lets the calculator give answers as complex numbers.

**Disp** and **Prompt** are under **PRGM** button in **I/O**.

Remember that the "-" in from of the B is not the same as the "-" after it. Remember that 2A must be in parentheses so that you will divide by all of it.

---

## Drawing a Geometric Figure

This program is more elaborate and will teach you some of the graphics commands in your calculator. The program commands are divided into sections so that you can understand them better.

The first group of four commands are called the initialization section. Here we set up the graphing screen as we wish it to be by clearing old graphics and sizing it properly. Then follows three lines where data is entered; ending with the computation needed to place a triangle and a circle on the drawing.

ClrDraw

:ZStandard

:ZSquare

:Zoom In

:1→A:2→B

:3→C:4→D

:-1→E:3→F

:Ymin+3→Ymin

:Ymax+3→Ymax

:Line(A,B,C,D)

:Line(A,B,E,F)

:Line(C,D,E,F)

: $\sqrt{(A-C)^2+(D-B)^2}$ →N

: $\sqrt{(A-E)^2+(F-B)^2}$ →M

: $\sqrt{(E-C)^2+(D-F)^2}$ →L

:(L+M+N)/2→S

: $\sqrt{S(S-L)(S-M)(S-N)}$ →T

:T/S→R

:(L\*A+M\*C+N\*E)/(2\*S)→X

:(L\*B+M\*D+N\*F)/(2\*S)→Y

:Pt-On(X,Y,2)

:Circle(X,Y,R)

## Comments on this Program

This program computes the quantities needed to draw a triangle with vertices (1,2), (3,4), (-1, 3) and its inscribed circle. You should know what is being done until the last 6 lines where the circle is being computed. Do not worry that you do not yet know the formulas, you will!

The **ClrDraw** command is found in the **DRAW** menu.

**ZStandard** under the **ZOOM** button set the graphing screen as it was from the factory. **ZSquare** sets the x and y axes to the same scale so that squares look square and circles circular. Finally **ZoomIn** makes the picture larger as appropriate for this drawing.

These three lines store the x and y coordinates of the three vertices of the triangle. Remember that the arrow is entered using **STO** at the lower left of the calculator. Remember to use the minus (-) rather than the subtraction operation when you enter negative numbers. The colon is used to separate two separate commands.

These two lines reset the window of the graph. Both the bottom and the top of the window are moved up 3. Moving the window helps us see our drawing better. Ymin and Ymax are found in the **VARS** menu, option 1, **Window...**

These three lines each use the **Line** command found in the **DRAW** menu to draw the three sides of the triangle.

These three lines each use the distance formula to compute the lengths of the sides of the triangle.

This line computes perimeter divided by 2 and stores in S.

This line computes the area of the triangle using Heron's formula and stores it in T.

This line computes the radius of the inscribed circle.

These two lines compute the x and y coordinates of the center of the circle. This formula is my gift to you and is difficult to explain.

This line marks a point at the center of the circle. **Pt-On** is found in the **DRAW** menu. The 2 gives specifies the shape of the point.

The final line draws the **Circle** centered at (X, Y) and of radius R. **Circle** is found in the **DRAW** menu.

# Selected Calculator Menus

This page will help you find commands hidden in your calculator. This page is specifically designed to help you with the programming in this brochure.

## MATH Menus

MATH	NUM	CPX	PRB
1: $\blacktriangleright$ Frac	to Fraction		
2: $\blacktriangleright$ Dec	to Decimal		
3: $^3$	Cube		
4: $^{\sqrt[3]{}}$	Cube Root		
5: $^{\sqrt{x}}$	xth Root		
6: fMin(	Function Min		
7 $\downarrow$ fMax	Function Max		
8: NDerive(	Num Derivative		
10: Solver_	Solve		

TEXAS INSTRUMENTS TI-83Plus

MATH	NUM	CPX	PRB
1: abs(	Absolute Value		
2: round(	Round		
3: iPart	Integer Part		
4: fPart	Fractional Part		
5: int	Integer		
6: min(	Minimum		
7 $\downarrow$ max(	Maximum		
8: lcm(	lowest multiple		
9: gcd(	greatest divisor		

TEXAS INSTRUMENTS TI-83Plus

MATH	NUM	CPX	PRB
1: conj(	Conjugate		
2: real(	Real Part		
3: imag(	Imaginary Part		
4: angle(	Complex Angle		
5: abs(	Magnitude		
6: $\blacktriangleright$ Rect	to Rectangular		
7: $\blacktriangleright$ Polar	to Polar		

TEXAS INSTRUMENTS TI-83Plus

MATH	NUM	HYP	PRB
1: Rand	Random decimal		
2: nPr	Permutation		
3: nCr	Combination		
4: !	Factorial		
5: randInt(	Random Integer		
6: randNorm	Random Normal		
7: randBin(	Random Binary		

## PRGM Menus

EXEC	EDIT	NEW
1: Prgm1		
2: Prgm2		
3: Prgm3		
4: Prgm4		
5: Prgm5		
6: Prgm6		
7 $\downarrow$ Prgm7		

These 3 menus let you  
EXEC - run program  
EDIT - change a program  
NEW - create a program

If you know the name of a command, but do not know where it is, 2nd Catalog will find it for you.

TEXAS INSTRUMENTS TI-83Plus

CTL	I/O	EXEC
1: If	If test	
2: Then	Then statement	
3: Else	Else statement	
4: For(	For( test)	
5: While	While test	
6: Repeat	Repeat statement	
7 $\downarrow$ End	End program	
8: Pause	Stop and go Home	
9: Lbl	Set Label	
0: Goto	Goto label	
A: IS>(	Increment and skip	

TEXAS INSTRUMENTS TI-83Plus

CTL	I/O	EXEC
1: Input	Input number	
2: Prompt	Prompt for number	
3: Disp	Display num or txt	
4: DispGraph	Show graph	
5: Disp Table	Show Table	
6: Output(	Output	
7 $\downarrow$ getKey	read key pressed	
8: ClrHome	Clear home	

## TEST Menu

TEST
1: =
2: $\neq$
3: >
4: $\geq$
5: <
6: $\leq$

These are used with the If, While, and For programming commands to control the flow of the program

## DRAW Menu

DRAW	POINTS	STO
1: ClrDraw	Clear Drawing	
2: Line(	line( 2 pts)	
3: Horizontal	Horizontal line	
4: Vertical	Vertical line	
5: Tangent(	tangent ( Yn)	
6: DrawF	draws fn on grph	
7 $\downarrow$ Shade(	shades	
8: DrawInv	inverse	
9: Circle(	Circle(x,y,r)	

TEXAS INSTRUMENTS TI-83Plus

DRAW	POINTS	STO
1: Pt-On(	point at (x,y)	
2: Pt-Off(	Turn point off (x,y)	
3: Pt-Chang	change color	
4: Pxl-On(	turn pixel on	
5: Pxl-Off(	turn pixel off	
6: Pxl-Chang	change pixel color	
7: Pxl-Test		

## VARS Menu

VARS	
1: Window	Very important
2: Zoom_	menu; gives access
3: GDB_	to many different
4: Picture_	variables.
5: Statistics_	
6: Table_	
7: String_	