2 Math Operations

Keyboard Math Operations

The following sections explain how to use the math functions, including [2nd] functions, found on the TI-73 keyboard. All of the examples in these sections assume that you are on the Home screen and that defaults are selected (unless specified otherwise).

Real numbers include fractions unless specified otherwise.

Basic Operations [+], [−], [×], [÷]

Returns the sum ([+]), difference ([−]), product ([×]), or quotient ([÷]) of *valueA* and *valueB*, which can be real numbers, expressions, or lists.

If both *values* are lists, they must have the same number of elements. If one *value* is a list and the other is a non-list, the non-list is paired with each element of the list, and a list is returned.

valueA + *valueB*
valueA − *valueB*
valueA ∗ *valueB*
valueA ÷ *valueB*

Add ⁻456 + 123.

[CLEAR] [(−)] **456** [+] **123** [ENTER]

```
-456+123       -333
```

Divide 45.68 ÷ 123.

45.68 [÷] **123** [ENTER]

```
-456+123       -333
45.68/123
        .3713821138
```

Multiply log(20) × cos(60).

[MATH] [▶] [▶] [▶] **1** **20** [)] [×] [2nd] [TRIG] **3** **60** [)] [ENTER]

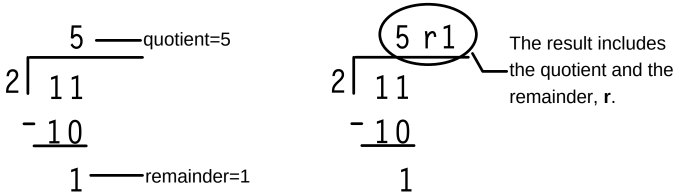
```
-456+123       -333
45.68/123
        .3713821138
log(20)*cos(60)
        .6505149978
```

In **Degree** mode

Integer Division [2nd] [INT÷]

[2nd] [INT÷] divides two positive integers and displays the quotient and the remainder, **r**.

posintegerA **Int/** *posintegerB*

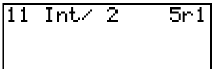


You can include integer division in an expression, but the remainder may not be displayed as part of the final answer.

After a calculation with [2nd] [INT÷] is completed, only the quotient from the result is stored in **Ans** (last answer). Therefore, if you use the result in another calculation, the remainder is ignored.

Calculate 11 ÷ 2 using integer division.

[CLEAR] **1 1** [2nd] [INT÷] **2**
[ENTER]

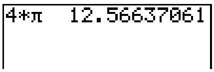


π [2nd] [π]

Represents the value for the constant, π, in calculations. The calculator uses π=3.1415926535898, although it only displays 3.141592654 on the screen. π acts as a real number in any calculation.

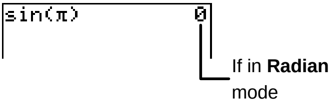
Multiply 4 × π.

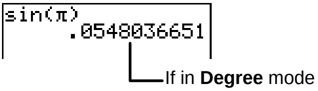
[CLEAR] **4** [×] [2nd]
[π] [ENTER]



Calculate sin(π).

[CLEAR] [2nd] [TRIG] **1** [2nd] [π] [)] [ENTER]





Percent [%]

Changes a *real_number* to percent. Results display according to the Decimal Notation mode setting.

real_number%

Convert ⁻30.6% to a decimal.

1. Select Float Decimal setting.

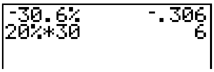
   [MODE] [▼] [ENTER]
   [2nd] [QUIT]

2. Convert ⁻30.6% to a decimal.

   [CLEAR] [(–)] **3 0 . 6** [%]
   [ENTER]

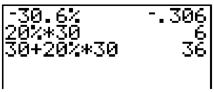
Calculate 20% of 30.

2 0 [%] [×] **3 0** [ENTER]



Calculate 30 + 20% of 30.

3 0 [+] **2 0** [%] [×] **3 0** [ENTER]



Inverse Function [2nd] [x^{-1}]

Returns the inverse, x^{-1}, of *value*, which is the equivalent of the reciprocal, $^1/_x$, of a real number, expression, or each element in a list.

value $^{-1}$

Important: To ensure that results are displayed as simple fractions instead of mixed numbers, select **b/c** Display Format mode.

Calculate $^5/_8{}^{-1}$.

[CLEAR] **5** [b/c] **8** [▶] [2nd] [x^{-1}] [ENTER]

Calculate -2.5^{-1}.

[(-)] **2 . 5** [2nd] [x^{-1}] [ENTER]

Square [x^2]

Finds the square of a real number, an expression, or each element in a list. **Note**: Using parentheses with [x^2] ensures that you get the correct answer. Refer to Appendix B: Reference Information for Equation Operating System (EOS) calculation rules.

value^2

Calculate 5^2.

[CLEAR] **5** [x^2] [ENTER]

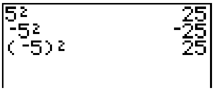
Compare the results of -5^2 and $(-5)^2$.

1. Calculate -5^2.

   [(-)] **5** [x^2] [ENTER]

2. Calculate $(^-5)^2$.

   [(] [(-)] **5** [)] [x^2] [ENTER]



Power [^]

Raises *value* to any *power*. *value* and *power* can be real numbers, expression, or lists. If both are lists, they must have the same number of elements. If one argument is a list and the other a non-list, the non-list is paired with each element of the list, and a list is returned.

value^*power*

value is limited by mathematical rules. For example, (⁻4)^.5 results in an error because this is the equivalent of (⁻4)^$^1/_2$, which is $\sqrt{^-4}$, a complex number.

Calculate 2^5.

[CLEAR] **2** [^] **5** [ENTER]

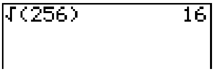
Square Root [2nd] [√]

Calculates the square root of *value*, which can be a positive real number, an expression that results in a positive real number, or a list of positive numbers.

√(*value*)

Calculate $\sqrt{256}$.

[CLEAR] [2nd] [√] **2 5 6** [)]
[ENTER]



Test Operations [2nd] [TEXT]

The two types of test operations included in the Text editor are relational operators (**=**, ≠, **>**, ≥, **<**, and ≤) and logic (Boolean) operators (**and** and **or**).

Both relational and logic operators often are used in programs to control program flow and in graphing to control the graph as a function over specific values.

Relational Operators

Relational operators compare *conditionA* and *conditionB* and return **1** if the conditional statement is true. They return **0** if the conditional statement is false. *conditionA* and *conditionB* can be real numbers, expressions, or lists.

If both *conditions* are lists, they must have the same number of elements. If one *condition* is a list and the other a non-list, the non-list is compared with each element of the list, and a list is returned.

Test operations are frequently used in programs.

conditionA **relational_operator** *conditionB*

Relational operators are evaluated after mathematical functions according to EOS rules (Appendix B: Reference Information). Therefore, for 2+2=2+3, the TI-73 returns **0**. It compares 4 with 5 and returns 0, because the operation is false. For 2+(2=2)+3, the TI-73 returns **6**. The relational test in parentheses returns 1, because the operation is true. Then it adds 2+(1)+3.

Operator:	Returns true (1) if:
= (equal)	Two conditions are equal.
≠ (not equal to)	Two conditions are not equal.
> (greater than)	*conditionA* is greater than *conditionB*.
≥ (greater than or equal to)	*conditionA* is greater than or equal to *conditionB*.
< (less than)	*conditionA* is less than *conditionB*.
≤ (less than or equal to)	*conditionA* is less than or equal to *conditionB*.

Logic (Boolean) Operators

Logic (Boolean) operators compare *conditionA* and *conditionB* and return **1** if the conditional statement is true. They return **0** if the conditional statement is false. *conditionA* and *conditionB* can be real numbers, expressions, or lists.

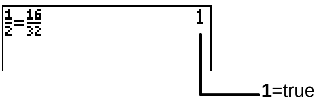
If both *conditions* are lists, they must have the same number of elements. If one *condition* is a list and the other a non-list, the non-list is compared with each element of the list, and a list is returned.

conditionA **and** *conditionB*
conditionA **or** *conditionB*

Operator:	Returns true (1) if:
and	Both conditions are nonzero.
or	At least one condition is nonzero.

Test $^1/_2 = ^{16}/_{32}$.

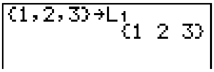
[2nd] [QUIT]
[CLEAR] **1** [b/c] **2** [▸]
[2nd] [TEXT] **=** [ENTER] **Done**
[ENTER] **1 6** [b/c] **3 2** [ENTER]



For **L1={1,2,3}**, test **L1**>**log(30)**.

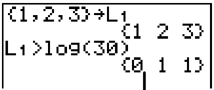
1. Define **L1**.

   [CLEAR] [2nd] [TEXT]
   { [ENTER] **1** [,] **2** [,] **3 }**
   [ENTER] **Done** [ENTER]
   [STO▶] [2nd] [STAT] **1** [ENTER]



2. Test **L1 > log(30)**.

   [2nd] [STAT] **1** [2nd] [TEXT]
   > [ENTER] **Done** [ENTER]
   [MATH] [◀] **1** **3 0** [)] [ENTER]



1>log(30) is false;
2>log(30) is true;
3>log(30) is true.

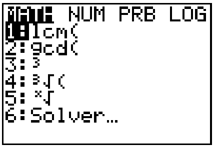
Test **cos(90) and sin(0)**.

   [CLEAR] [2nd] [TRIG] **3**
   9 0 [)] [2nd] [TEXT] **and**
   [ENTER] **Done** [ENTER]
   [2nd] [TRIG] **1** **0** [)] [ENTER]

The [MATH] MATH Menu

The [MATH] **MATH** menu includes various math functions.

[MATH]



1:lcm(	Finds the least common multiple, which is the smallest number that two integers can divide into evenly.
2:gcd(	Finds the greatest common divisor, which is the largest number that divides into two integers evenly.
3:³	Calculates the cube.
4:³√(	Calculates the cube root.
5:ˣ√	Calculates the x^{th} root.
6:Solver…	Displays the Equation Solver.

lcm([MATH] 1

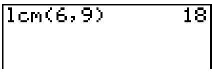
The least common multiple (LCM) function returns the smallest number that two positive whole numbers can divide into evenly, of two positive whole numbers or lists of positive whole numbers. If both arguments are lists, they must have the same number of elements. If one argument is a list and the other a non-list, the non-list is paired with each element of the list, and a list is returned.

lcm(is frequently used with fractions to find a common denominator. See Chapter 3: Fractions for more information on entering fractions.

lcm(*valueA*,*valueB***)**

Find the LCM of 6 and 9.

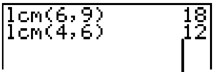
[CLEAR] [MATH] **1**
6 [,] **9** [)] [ENTER]



Add $^1/_4 + {^5/_6}$ (using LCM).

1. Find the LCM of the denominators.

   [MATH] **1**
   4 [,] **6** [)] [ENTER]



Therefore, 12 is the common denominator.

2. Use the LCM to convert $^1/_4$ and $^5/_6$ to fractions where 12 is the common denominator (without using the calculator).

$$\frac{1}{4} \times \frac{3}{3} = \frac{3}{12}$$

$$\frac{5}{6} \times \frac{2}{2} = \frac{10}{12}$$

LCM=12

3. Add the newly converted fractions (without using the calculator).

$$\frac{3}{12} + \frac{10}{12} = \frac{13}{12}$$

4. Verify your answer by adding the original fractions on the calculator. Select the **b/c** Display Format mode setting and clear the Home screen, if desired.

   [MODE] [▼] [▼] [▼] [▶] [ENTER]
   [2nd] [QUIT] [CLEAR]
   1 [b/c] **4** [▶] [+] **5** [b/c] **6**
   [ENTER]

gcd([MATH] 2

The greatest common divisor (GCD) function returns the largest number that divides into two positive whole numbers or lists of positive whole numbers evenly. If both arguments are lists, they must have the same number of elements. If one argument is a list and the other a non-list, the non-list is paired with each element of the list, and a list is returned.

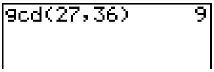
This is frequently used with fractions to reduce them to lowest terms. See Chapter 2: Fractions for more information on entering fractions.

gcd(*valueA*,*valueB***)**

Find the greatest common divisor for the fraction, $^{27}/_{36}$.

1. Find the GCD of $^{27}/_{36}$.

   [MATH] **2**
   2 7 [,] **3 6** [)] [ENTER]



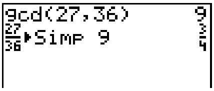
2. Simplify the fraction completely using the GCD (without using the calculator).

GCD=9

$$\frac{27}{36} \div \frac{9}{9} = \frac{3}{4}$$

3. Verify your answer by simplifying $^{27}/_{36}$ by 9 on the calculator. You must be in **Mansimp** mode setting.

   [MODE] [▼] [▼] [▼] [▼] [▶]
   [ENTER] [2nd] [QUIT]
   2 7 [b/c] **3 6** [▶] [SIMP] **9**
   [ENTER]



3 [MATH] 3

Calculates the cube of n, which is equivalent to $n \times n \times n$ of any real number, expression, or each element in a list.

n^3

Calculate 5^3.

   5 [MATH] **3**
   [ENTER]

$5^3 = 5 \times 5 \times 5 = 125$

$^3\sqrt{}$([MATH] 4

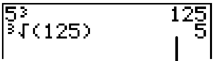
Calculates the cube root of *value*, which is equivalent to *n* where n^3=*value*. *value* can be a real number, expression, or list.

For n^3=value, $^3\sqrt{}$value=n

$^3\sqrt{}$(*value*)

Calculate $^3\sqrt{}(125)$.

[MATH] **4** **1 2 5** [)] [ENTER]



$^3\sqrt{}$(125) = 5
because 5^3=125

$^x\sqrt{}$ [MATH] 5

Calculates the x^{th} root of *value*, which is equivalent to *n* where n^x=*value*. *value* can be a real number, expression, or list. *x* can be any real number.

For n^x=value, $^x\sqrt{}$value=n

x **$^x\sqrt{}$(*value*)**

Calculate $^6\sqrt{}64$.

[CLEAR] **6** [MATH] **5**
6 4 [ENTER]

6$^x\sqrt{}$64 = 2
because 2^6=64

Solver [MATH] 6

The Equation Solver allows you to solve for one unknown one-letter variable in an equation containing up to 5 one-letter variables. By default, the equation is assumed to be equal to 0; however, you can set the equation equal to any real number (or an expression that results in a real number).

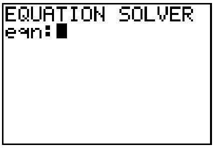
The screen you see when you select **Solver** depends on whether an equation has been defined previously.

To exit Solver and return to the Home screen, press [2nd] [QUIT].

The EQUATION SOLVER Screen

If no equation is currently defined, pressing [MATH] **6** takes you to the **EQUATION SOLVER** screen. Enter the equation at the cursor, using the Text editor ([2nd] [TEXT]) to enter the variable names.

[MATH] **6**



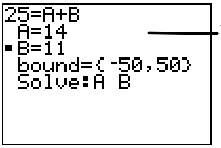
You can have more than one variable on each side of the equation. For example, **A + B = B + D + E**.

If you do not set the equation equal to a value, the calculator automatically sets it equal to 0. For example, to enter A+B=0, just enter **A+B** and press [ENTER]. You are limited to 5 variables per equation.

The Equation Variables Screen

If an equation has been defined previously, pressing [MATH] **6** takes you to the Equation Variables screen.

[MATH] **6**



Your screen may vary.

Equation	Displays the currently defined equation.
Equation Variables	Displays all equation variables and their values.
bound Default={⁻1E99,1E99}	Displays the **bound** limits that apply to the unknown variable value for which you are solving.
Solve	You select one variable, the one you want to solve for, from this list.

Equation

The first line of the Equation Variables screen displays the equation you defined on the **EQUATION SOLVER** screen.

If you would like to edit a defined equation, press [▲] until the **EQUATION SOLVER** screen is displayed. Edit the equation with [CLEAR], [DEL], or [2nd] [INS], as necessary. Then press [ENTER] to return to the Equation Variables screen.

Equation Variables

All variables included in the defined equation are displayed. If those variables have never been assigned a value, they are set equal to 0. If a variable has been defined previously (for example, from the Home screen), that value appears.

If a value extends beyond the screen, press [▶] to scroll to the end of the number. This is especially important if a number is in scientific notation and you need to see whether it has a negative or positive exponent.

For an equation with more than one variable, you must define all variables except the unknown variable for which you want to solve.

bound

bound limits apply to the unknown variable value for which you are solving. Default bounds are {⁻1E99,1E99}. Use these limits to narrow the unknown value solution to a specific range of numbers, especially if more than one answer exists.

Hint: For answers with many solutions (for example, trig functions), consider graphing the function first to get an idea of the most ideal (or specific) **bound** limits.

Solve

Specify the unknown variable from the **Solve** line. This prompts the calculator to solve for it.

To select a variable on the **Solve** line, highlight the unknown variable with the cursor, and then press [ENTER]. After you press [ENTER], a solid black square appears next to the solved (previously unknown) variable displayed in the Equation Variables section.

Hint: The Solver allows for a small tolerance when solving a result, which is noticeable especially when solving complex equations or those with multiple solutions. For example, a result of 3.99999999999999 (instead of 4) for the equation **16=x²** is considered a correct answer.

Solving Equations with Only One Possible Answer

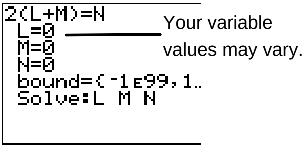
For 2(*L*+*M*)=*N*, solve for *L* when *N*=268, and *M*=40, ⁻14, and 307.

1. Define the equation on the **EQUATION SOLVER** screen.

   [MATH] **6**
   [▲] [CLEAR] (if necessary)

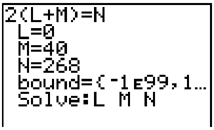
2. Enter the equation.

   [2nd] [TEXT]
   2 [(] **L** [ENTER] [+] **M** [ENTER]
   [)] **=** [ENTER] **N** [ENTER]
   Done [ENTER] [ENTER]



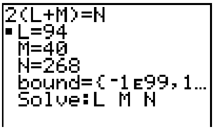
3. Enter the first value for **M**, 40, and **N**, 268.

   [▼] **4 0** [▼] **2 6 8**



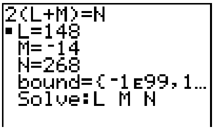
4. Solve for **L**.

   [▼] [▼] [ENTER]



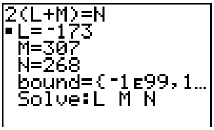
5. Solve for **L** when **M**=⁻14.

   [▼] [(-)] **1 4** [▼] [▼] [▼] [ENTER]



6. Solve for **L** when **M**=307.

   [▼] **3 0 7** [▼] [▼] [▼] [ENTER]



Solving Equations with More Than One Answer

The calculator only returns one solution even if more than one possible solution exists. When this is the case, you can first enter a guess by assigning a value to that variable and then asking the calculator to solve your equation. The TI-73 always chooses the solution closest to that guess. However, the guess must be within the bound limits; otherwise, you get an error.

Find the negative solution to the equation, $16 = X^2$.

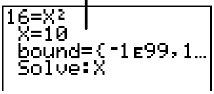
1. Define the equation on the **EQUATION SOLVER** screen.

   [MATH] **6**
   [▲] [CLEAR] (if necessary)

2. Enter the equation.

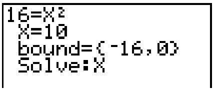
   [2nd] [TEXT]
   1 6 = [ENTER] [x] [x^2] **Done**
   [ENTER] [ENTER]

Your **X** value may vary.



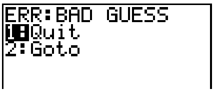
3. Use **bound** to limit your answer to a negative one (between -16 and 0).

   [▼] [▶] [(-)] **16** [DEL] [DEL] [▶] **0**
   [DEL] [DEL] [DEL]



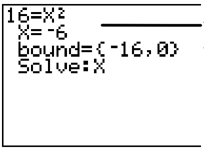
4. Solve for **X**.

   [▼] [ENTER]



5. The guess, **X=10**, is not between the limit bounds. You must clear or change it. (This step uses a different guess, -6.)

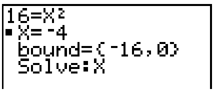
   2 [CLEAR] [(-)] **6**



-6 is between the bounds.

6. Solve for **X**.

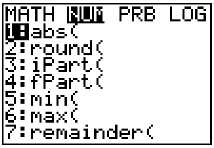
   [▼] [▼] [ENTER]



The [MATH] NUM Menu

The [MATH] **NUM** (number) menu includes seven different math functions.

[MATH] [▶]



1:abs(	Calculates the absolute value of a real number, list, or expression.
2:round(	Rounds a real number, list, or expression.
3:iPart(	Returns only the integer part of a result.
4:fPart(	Returns only the fractional part of a result.
5:min(	Returns the minimum of two real numbers, lists, or expressions.
6:max(	Returns the maximum of two real numbers, lists, or expressions.
7:remainder(	Returns the remainder resulting from the division of two real numbers or lists.

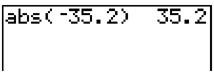
abs([MATH] [▶] 1

Returns the absolute value of a real number, expression, or each element in a list. For an expression, the expression is calculated and the absolute value of that result is returned.

abs(*value***)**

Find the absolute value of ⁻35.2.

[MATH] [▶] **1** [(–)] **3 5 . 2** [)]
[ENTER]



round([MATH] [▸] 2

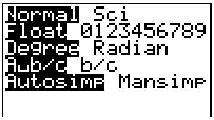
Returns a number, expression, or each element in a list rounded to 10 digits or #*decimal_places* (≤9), if specified. The final result is always displayed according to the Decimal Notation mode ([MODE]) unless #*decimal_places* is specified, which overrides the current setting. Notice that the Decimal Notation mode settings *do* change the display but not the value of the result. Therefore, the entire result is stored in the calculator ready to use for future calculations, as applicable.

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

Round π to different numbers of decimal places using different Decimal Notation mode settings.

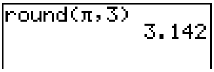
1. Set Decimal Notation mode to **Float**, if necessary.

   [MODE] [▾] [ENTER]
   [2nd] [QUIT] [CLEAR]



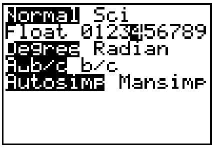
2. Round π to 3 decimal places.

   [MATH] [▸] 2 [2nd] [π]
   [,] 3 [)] [ENTER]



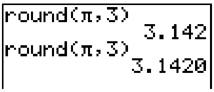
3. Set Decimal Notation mode to **4**.

   [MODE] [▾] [▸] [▸] [▸] [▸] [▸]
   [ENTER] [2nd] [QUIT]



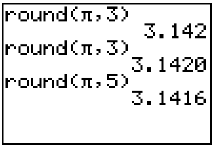
4. Round π to 3 decimal places.

   [2nd] [ENTRY] [ENTER]



5. Leave the Decimal Notation mode at **4** and round π to 5 digits.

   [2nd] [ENTRY] [◄] [◄] **5**
   [ENTER]



iPart(and fPart([MATH] [►] 3 and 4

iPart(returns the integer part of a real number, expression, or each element in a list. For an expression, the expression is calculated and the integer part of the result is displayed.

iPart(*value***)**

fPart(returns the fractional part of a real number, expression, or each element in a list. For an expression, the expression is calculated and the fractional part of the result is displayed.

If *value* is a mixed number, the fractional part is returned and displayed according to the current Simplification mode setting.

fPart(*value***)**

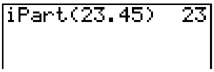
Find the integer and fractional part of 23.45.

1. Set Decimal Notation mode to **Float**.

   [MODE] [▼] [ENTER]
   [2nd] [QUIT]

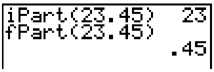
2. Find the integer part.

   [CLEAR] [MATH] [►] **3**
   2 3 . 4 5 [)] [ENTER]



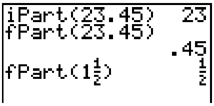
3. Find the fractional part.

   [MATH] [►] **4**
   2 3 . 4 5 [)] [ENTER]



Find the fractional part of 1 $^1/_2$.

[MATH] [▶] **4** **1** [UNIT]
1 [b/c] **2** [▶] [)] [ENTER]



min(and max([MATH] [▶] 5 and 6

These are identical to the **min(** and **max(** commands found on the [2nd] [STAT] **MATH** menu.

min((minimum) returns the smaller of two *values* or the smallest element in one *list*. *value* can be a real number, expression, or a list.

If both arguments are lists, they must have the same number of elements. If one argument is a list and the other a non-list, the non-list is compared with each element of the list, and a list is returned.

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

max((maximum) functions exactly like **min(**, but it always returns the *larger* of two *values* or the largest element in a list.

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

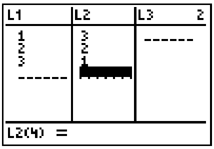
For this example, the Decimal mode settings are set to **Float**.

Compare **L1** and **L2** to find the **min(** and **max(**. **L1={1,2,3}**, and **L2={3,2,1}**.

1. Define **L1** and **L2** in the List editor.

   [LIST]

   For more information on entering lists, see Chapter 5: Lists.



2. Find the list minimums.

   [2nd] [QUIT] [CLEAR]
   [MATH] [▶] **5**
   [2nd] [STAT] **1** [,]
   [2nd] [STAT] **2** [)] [ENTER]

```
min(L1,L2)
        {1 2 1}
```

3. Find the list maximums.

   [MATH] [▶] **6**
   [2nd] [STAT] **1** [,]
   [2nd] [STAT] **2** [)] [ENTER]

```
min(L1,L2)
        {1 2 1}
max(L1,L2)
        {3 2 3}
```

remainder([MATH] [▶] 7

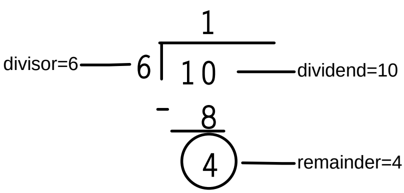
Returns the remainder resulting from the division of two positive whole numbers, *dividend* and *divisor*, each of which can be a list of positive whole numbers. They also are subject to mathematical rules. For example, *divisor*≠0.

remainder(*dividend*,*divisor***)**

If both arguments are lists, they must have the same number of elements. If one argument is a list and the other a non-list, the non-list is paired with each element of the list, and a list of remainders is returned.

remainder(*list*,*divisor***)**
remainder(*dividend*,*list***)**
remainder(*list*,*list***)**

Calculate 10÷6 and return the remainder only.



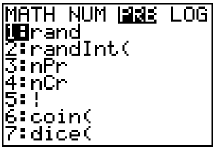
[CLEAR] [MATH] [▶] **7**
1 0 [,] **6** [)] [ENTER]

```
remainder(10,6)
              4
```

The [MATH] PRB Menu

The [MATH] **PRB** (probability) menu lets you select functions that are often used to calculate probabilities.

[MATH] [▶] [▶]



1:rand	Generates a random number between 0 and 1.
2:randInt(	Generates a random integer between two values.
3:nPr	Calculates the number of permutations for a group of items.
4:nCr	Calculates the number of combinations for a group of items.
5:!	Calculates the factorial of a positive integer.
6:coin(	Simulates one or more coin tosses.
7:dice(	Simulates one or more dice rolls.

rand [MATH] [▶] [▶] 1

Generates a random real number between 0 and 1 (0<*number*<1). **rand** takes no arguments.

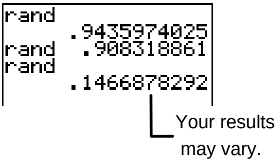
rand

If you want to control a sequence of random numbers, first store an integer “seed value” to **rand**. The calculator generates a specific sequence of random numbers from each seed value. To get a different sequence, use a different seed value. The default seed value is 0.

seed [STO▶] **rand**

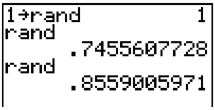
Generate a sequence of random numbers using whatever value happens to be the current seed.

[CLEAR] [MATH] [▸] [▸] **1**
[ENTER] [ENTER] [ENTER]



Generate a sequence of random numbers using *seed*=1.

[CLEAR] **1** [STO▸] [MATH] [▸]
[▸] **1** [ENTER] [MATH] [▸] [▸] **1**
[ENTER] [ENTER]



randInt([MATH] [▸] [▸] 2

Generates a random integer between *lower* and *upper* (both integers) boundaries.

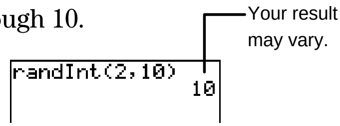
The random integer returned may be one of the boundaries. For example, **randInt(1,5)** may return 1, 2, 3, 4, or 5.

To generate more than one random integer, specify *#ofIntegers*, a positive whole number >0.

randInt(*lower*,*upper*[,*#ofIntegers*]**)**

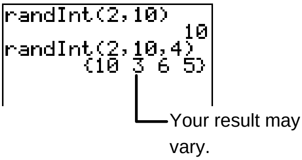
Find a random integer from 2 through 10.

[CLEAR] [MATH] [▸] [▸] **2**
2 [,] **1 0** [)] [ENTER]



Find 4 random integers from 2 through 10. (Recall and edit the last entry.)

[2nd] [ENTRY] [◂] [,] **4** [)]
[ENTER]



nPr [MATH] [▶] [▶] 3

Returns the number of permutations of n *items* taken r *number* at a time. The order in which you select the items DOES matter. *items* and *number* can be nonnegative integers or lists of nonnegative integers.

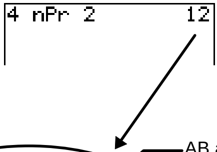
If both arguments are lists, they must have the same number of elements. If one argument is a list and the other a non-list, the non-list is paired with each element in the list, and a list of permutations is returned.

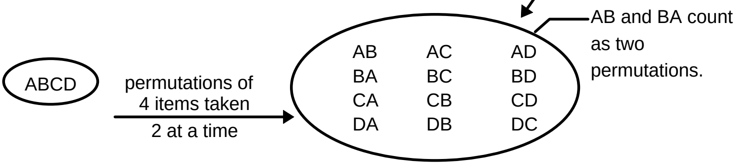
items **nPr** *number*

From a group of 4 items (ABCD), how many ways can you select 2 of the items if the order does matter?

Find 4 **nPr** 2.

[CLEAR] **4** [MATH] [▶] [▶] **3**
2 [ENTER]





nCr [MATH] [▶] [▶] 4

Returns the number of combinations of n *items* taken r *number* at a time. In combinations, the order in which you select the items DOES NOT matter. *items* and *number* can be nonnegative integers or lists of nonnegative integers.

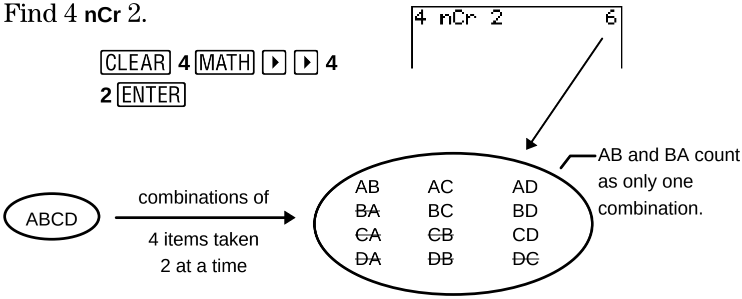
If both arguments are lists, they must have the same number of elements. If one argument is a list and the other a non-list, the non-list is paired with each element in the list, and a list of combinations is returned.

items **nCr** *number*

From a group of 4 items (ABCD), how many ways can you select 2 of the items if the order does not matter?

Find 4 **nCr** 2.

[CLEAR] **4** [MATH] [▸] [▸] **4**
2 [ENTER]



! [MATH] [▸] [▸] 5

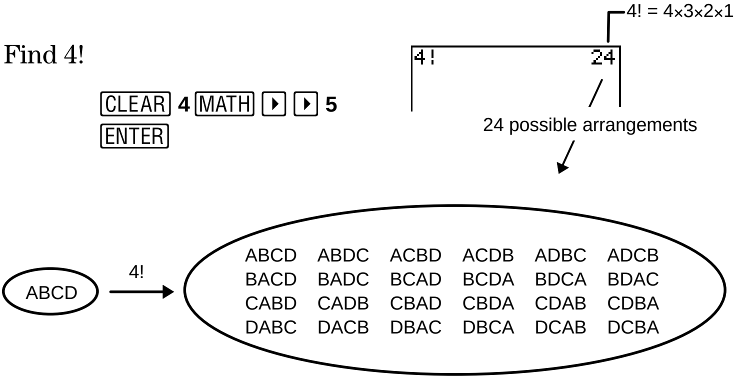
Returns the factorial of *value*. *value* can be an integer or list of integers between 0 and 69. By definition, 0! = 1.

Factorials are similar to permutations because the order DOES matter. You can think of 4! as the total number of ways that 4 items can be arranged.

value **!**

Find 4!

[CLEAR] **4** [MATH] [▸] [▸] **5**
[ENTER]



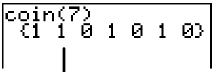
coin([MATH] [▶] [▶] 6

Returns a random list of 0s and 1s that represents heads and tails for one or more coin *tosses*. *tosses* is a positive whole number.

coin(*tosses***)**

Simulate tossing a coin 7 times.

[CLEAR] [MATH] [▶] [▶] **6**
7 [ )] [ENTER]



4 heads and 3 tails (or 3 heads and 4 tails). Your result may vary.

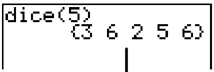
dice([MATH] [▶] [▶] 7

Returns a random list of numbers (between 1 and 6) that represents dice rolls. **dice(** takes one optional argument, *#ofdice*, a positive whole number>1. If *#ofdice* is specified, each list element is the total sum of one roll's results.

dice(*rolls*[,*#ofDice*]**)**

Simulate 5 dice rolls for one die.

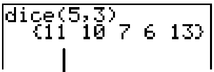
[CLEAR] [MATH] [▶] [▶] **7**
5 [ )] [ENTER]



Your result may vary.

Simulate 5 rolls of 3 dice.

[CLEAR] [2nd] [ENTRY]
[◀] [,] **3** [ )] [ENTER]

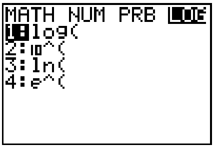


The three dice totaled 11 on the first roll, 10 on the second roll, etc. Your result may vary.

The [MATH] LOG Menu

The [MATH] **LOG** (logarithm) menu lets you select functions that are used to calculate base-10 and base-*e* logarithms and powers.

[MATH] [▶] [▶] [▶]
-or-
[MATH] [◀]



1:log(	Returns the base-10 logarithm of a value.
2:10^(	Raises 10 to a power.
3:ln(	Calculates the natural logarithm of a value.
4:e^(	Raises e to a power (e = 2.71828182846).

log([MATH] [▶] [▶] [▶] 1

The logarithm is the exponent, x, indicating the power which a fixed number (using base 10) must be raised to in order to produce a given number, a.

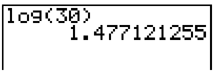
For $10^x = a$, $\log_{10} a = x$

log(returns the logarithm of a positive real number, an expression that results in a positive real number, or a list of positive real numbers.

log(*value***)**
log(*list***)**

Calculate log(30).

[CLEAR] [MATH] [▶] [▶] [▶] **1**
3 0 [)] [ENTER]



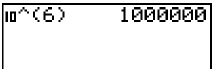
10^([MATH] ► ► ► 2

Raises 10 to a power of x, where x is an integer, an expression that results in an integer, or a list of integers. If $x \leq 10^{-4}$ or $\geq 10^{10}$, the result is displayed in scientific notation.

10^(*integer***)**
10^(*x***)**

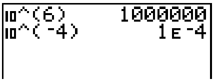
Calculate 10^(6), which is often written as 10^6.

[CLEAR] [MATH] ► ► ► **2**
6 [)] [ENTER]



Calculate 10^(⁻4).

[MATH] ► ► ► **2**
[(-)] **4** [)] [ENTER]



ln([MATH] ► ► ► 3

The natural logarithm is the exponent, x, indicating the power which the base, e, must be raised to in order to produce a given number, a.

```
For eˣ=a, ln(a)=x
```

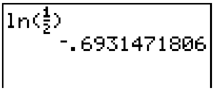
The calculator uses e=2.718281828459, although it only displays 2.718281828 on the screen.

ln(returns the natural logarithm of a positive real number, an expression that results in a positive real number, or a list of positive real numbers.

ln(*value***)**
ln(*list***)**

Calculate ln($^1/_2$).

[CLEAR] [MATH] [▶] [▶] [▶] **3**
1 [b/c] **2** [▶] [)] [ENTER]



e^([MATH] [▶] [▶] [▶] 4

Raises e to a power of x, where x is a real number, an expression that results in an real number, or a list of real numbers.

The calculator uses e=2.718281828459, although it only displays 2.718281828 on the screen.

e^(x**)**
e^(*list***)**

Calculate e^5, which is often written as e^5.

[CLEAR] [MATH] [▶] [▶] [▶] **4**
5 [)] [ENTER]