

Hypatia

Open Source
Programmable Text-based RPN Calculator
for Windows

Quick Reference Guide

This is not the manual!

For detailed information, see [hypatia-doc.pdf](#)

Version 4.2
www.hypatia-rpn.net

Table of Contents

Input Editor	2
Operators	3
1-Argument Operators	3
Large Number Units	4
1- and 2-Argument Unit Conversion Operators	4
2-Argument Operators	5
n-Argument Operators	6
n-Argument Delimiter	6
Pseudo Operators WHISK and DONE	6
Constants	7
Pseudo Constants	7
Variables	7
Result Variables	7
The Zero Threshold Variable \$zero	7
The Loop Exit Variable \$loop	7
User Defined Elements	8
Files	8
List of Commands	9
Mode Settings • Variables • Results	9
User Defined Elements • Clipboard • Files • Buffer	10
Scripts • Repeat and Loops • Help • Quit	11
List of Control Symbols	12

Input Editor

- ARROW LEFT, ARROW RIGHT, HOME and END keys move the cursor
- BACKSPACE deletes the character to the left of the cursor position
- DELETE deletes the character at the cursor position
- ENTER inputs the line as you see it, regardless of the cursor position

When the cursor is at the beginning of the line:

- ARROW UP and ARROW DOWN scroll through previous input lines, you can edit and reuse them

ESC (like HOME) moves the cursor to the beginning of the line. If you have scrolled up, pressing ESC twice, or pressing ESC at the beginning of the line, returns you to the current line.

Input is always in insert mode.

The length of the input line is limited by the width of the program window.

All Hypatia commands, variable and user defined element names, constants, file names etc. are case insensitive — upper or lower case characters make no difference.

Operators

1-Argument Operators

Syntax: a operator, where a is a number or a numerical expression

+-	changes algebraic sign
RCP	gives reciprocal value
ABS	absolute value
SIGN	1 if argument is positive, 0 if zero, -1 if negative
ISO	1 if argument is zero, 0 if not zero (positive or negative)
ISNOT0	1 if argument is positive or negative, 0 if zero
IS+	1 if argument is positive, 0 if zero or negative
ISO+	1 if argument is positive or zero, 0 if negative (you can also write IS+0)
IS-	1 if argument is negative, 0 if positive or zero
ISO-	1 if argument is negative or zero, 0 if positive (you can also write IS-0)
ISPOSINT	1 if argument is positive integer, 0 if not (error if argument > 1e18)
ISPRIME	1 if argument is prime number, 0 if not (error if argument not 0 or positive integer)
DIGITS	number of digits before decimal point, if negative: number of zeros after dec. point
INT	truncated integer value (for rounding to integer use 0 ROUND, error if argument $\geq 1e18$)
FRAC	fractional part of a decimal number (rounded to total of 18 digits, error if arg. $\geq 1e15$)
SQ	square
SQRT	square root
CUBE	cube
CBRT	cube root (argument can be negative)
LN	natural logarithm (base e), see 2-argument operator LOG^ for logarithm base b
LOG10	logarithm base 10
LOG2	logarithm base 2
EXP	e to the power of argument
EXP10	10 to the power of argument
EXP2	2 to the power of argument
!	factorial (argument must be 0 or positive integer, max. 449, rounded for arguments > 19)
RAD	converts angle from degrees to radians (do not confuse with USE RAD command)
DEG	converts angle from radians to degrees (do not confuse with USE DEG command)
SIN	sine (results less than $\pm 1e-18$ are rounded to zero)
COS	cosine (results less than $\pm 1e-18$ are rounded to zero)
TAN	tangent (abs. value of argument must be less than $\pi/2$ or 90 degrees)
ASIN	arcsine By default, all angles are in radians.
ACOS	arccosine You can change this to degrees by using the USE DEG command,
ATAN	arctangent or the -d command line option.

SIGN and the first six IS operators observe the zero threshold, by default $\pm 1e-16$ (see page 7).

FRAC rounds its result to a total of 18 digits of the argument because more decimal digits would be random, and then down to 0.999999999999999 (15 digits) if it is larger, to avoid the result 1.

Large Number Units

LAKH multiplies value by one hundred thousand (Indian numbering system)
 MILLION multiplies value by one million
 CRORE multiplies value by ten million (Indian numbering system)

Note that LAKH, MILLION and CRORE are 1-argument operators:

2 1 MILLION + is 1000002, 2 1 + MILLION is 3000000.

LAKH, MILLION and CRORE differ from other 1-argument operators in two ways:

- you can use them in MAXLOOP n, DO n and REPEAT n commands
- you can use them after numbers at the beginning of lines in chain calculations

1-Argument Unit Conversion Operators

:F degrees Celsius to Fahrenheit
 :C degrees Fahrenheit to Celsius
 :MI km to international miles
 :KM international miles to km
 :NAUTMI km to nautical miles, or km/h to knots
 :NAUTKM nautical miles to km, or knots to km/h
 :IN cm to inches
 :CM inches to cm
 :FT meter to feet (1 ft = 12 in)
 :M feet to meter (1 ft = 12 in)
 :LB kg to avoirdupois pounds
 :KG avoirdupois pounds to kg
 :OZ g to avoirdupois ounces (1 lb = 16 oz)
 :G avoirdupois ounces to g (1 lb = 16 oz)
 :TOZ g to troy ounces
 :TG troy ounces to g
 :GAL liters to US liquid gallons
 :L US liquid gallons to liters
 :MPG miles per gallon to liters per 100km, and vice versa

2-Argument Unit Conversion Operator

:MSDEC minutes and seconds to decimal hours or degrees

2-Argument Operators

Syntax: a b operator, where a and b are numbers or numerical expressions

+	addition, a + b
-	subtraction, a - b
*	multiplication, a * b
/	division, a / b
//	(a - b) / b, a if b = 0, set to 0 if within $\pm 1e-16$ (this is useful as a loop exit condition)
\	integer division, a \ b (absolute value of result is rounded off to the next integer)
^	power, a ^ b
ROOT	b th root of a (a must not be negative, only CBRT accepts negative radicand)
LOG^	logarithm of a base b (a must be positive, b must be > 1)
EQUAL	(or ==) 1 if a and b are equal, 0 if they differ
UNEQUAL	(or <>) 1 if a and b differ, 0 if they are equal
<<	1 if a is less than b, 0 if greater or equal
<=	1 if a is less than b or equal, 0 if greater
>>	1 if a is greater than b, 0 if less or equal
>=	1 if a is greater than b or equal, 0 if less
MULTIPLE	1 if a is positive multiple of b, a and b must be integer, 0 if not the same sign
ROUND	round a to b digits after decimal point; if b is 0, round to integer if b is negative, then round to b zeros before decimal point (b can be -9 to 15)
ROUNDS	round a to b significant digits (b can be 1 to 17)
GATE	zero if abs(a) is less than b, otherwise a (if threshold value b is 0, it is seen as 1e-16)
UPLIMIT	b is upper limit for a (result is a, but not higher than b)
LOLIMIT	b is lower limit for a (result is a, but not lower than b)
%	percent, b% of a
%+	percent +, a increased by b percent
%-	percent -, a diminished by b percent
%?	how many percent of a is b?
%N	net: if a is net amount plus b percent, what is the net amount?
%A	agio: if a is gross amount including b percent, what is the agio amount?
MOD	remainder, a modulo b (a and b must have the same sign)
IMOD	remainder, a modulo b (a and b must be integer and have the same sign)
BIN	binomial coefficient (a must be positive integer, b must be 0 or positive integer)
RANDINT	random integer, within the range of a to b (do not confuse with RAND pseudo constant)
RANDND	normal distributed random number, a is mean, b = standard deviation 0 0 = standard normal distribution, same as 0 1

The compare operators EQUAL or ==, UNEQUAL or <>, <<, <=, >> and >= observe the zero threshold, by default $\pm 1e-16$ (see page 7).

Do not confuse the compare operator == (same as EQUAL) with the == command!

n-Argument Operators

Syntax: $a_1 a_2 a_3 \dots a_n$ operator, where a_1 to a_n are numbers or numerical expressions.

n-Argument operators consider everything to their left as their arguments, either to the beginning of the calculation line, or to the n-argument delimiter | (see below).

N	number of arguments
N+	number of positive values
N0	number of zeros
N-	number of negative values
SUM	sum
MEAN	mean
GMEAN	geometric mean (all arguments must be > 0)
HMEAN	harmonic mean (all arguments must be > 0)
MED	median
SDEV	standard deviation
SQSUM	sum of squares
RCPSUM	sum of reciprocals
PROD	product
MIN	minimum
MAX	maximum
i ITEM	the value of the i^{th} item in the argument list (i must be an integer in the range from 1 to n) when i is negative, counts backwards from the end of the list

N+, N0 and N- operators observe the zero threshold, by default $\pm 1\text{e-}16$ (see page 7).

N-argument-operators (also called statistical operators) need at least 1 argument, except for N, which can have 0 or more, and SDEV and ITEM, which need at least 2.

n-Argument Delimiter |

While n-argument operators take anything to their left as their arguments, the vertical bar | can serve as a delimiter, hiding anything to its left from the next n-argument operator.

There are no restrictions to what stands left of the delimiter — it remains hidden until an n-argument operator is encountered, and then becomes visible again. There can be more than one delimiter in a calculation line, but each delimiter must be followed by a corresponding n-argument operator.

Pseudo Operators WHISK and DONE

They are called “pseudo operators” because like operators they manipulate the stack, but do not perform any calculations.

WHISK removes the two values left of it from the stack, and stores them under the names A and B, which can then be used in the calculation line.

DONE preserves the value immediately to its left, but deletes all prior values from the stack.

Constants

PI	3.141592653589793238
TAU	$\tau = 2\pi = 6.28318530717958648$
E	2.718281828459045235
PHI	1.618033988749894848 (the “golden ratio”)

Pseudo Constants

RAND	random decimal number in the range of 0 to 1 (do not confuse this with RANDINT and RANDND, which are 2-argument operators)
DUP	duplicates the preceding number, constant or variable in the input line
I	in DO loops: loop index, in REPEAT n loops: loop index + 1, outside of loops always 1
ISLOOP	in DO and REPEAT loops 1, outside of loops (when a script is called directly) 0
TIME	in DO and REPEAT loops the time since start in seconds, outside of loops always 0

Variables

Variable names begin with \$, and must not contain spaces.

Variables are created by assigning them a value, either through \$myvar = ... or STO \$myvar

See “List of Commands — Variables”, page 9.

Result Variables:

\$	the result of the previous completed calculation
\$\$	the result of the previous but one completed calculation

The Zero Threshold Variable \$zero

\$zero = ...	sets the zero threshold (default zero threshold is 1e-16) certain operators consider absolute values below this threshold to be zero
\$zero = 0	sets the threshold to 0
DEL \$zero	resets the threshold to the default value of 1e-16

The following operators are *not* affected by \$zero:

- a b // (a minus b divided by b) — it always uses a zero threshold of 1e-16
- a 0 GATE — uses a zero threshold of 1e-16
- SIN and COS always round their results to 0 if they are less than $\pm 1e-16$

The Loop Exit Variable \$loop

\$loop = ...	setting it to exactly zero in a loop exits the loop after the last line in the script see “List of Commands — Repeat, Loops and Scripts”, page 11
--------------	--

User Defined Elements

User defined element names begin with @, and must not contain spaces.

User defined elements (UDEs) are created by assigning them a text through @myude =

UDEs can contain data or operators. Their syntax is similar to that for variables, but they are used in input lines like insert files.

While a variable always is a number, a user defined element always is a text, though this text can consist of, or contain, numbers.

See “List of Commands — User Defined Elements”, page 10.

Files

All files are located in Hypatia’s program folder.

File names are lower case, they can not contain spaces or parentheses, and can not begin with @.

Hypatia creates the following files:

hy.ini	script file that is run when the program starts created as an empty file when it does not exist, you can edit it later
hy	result file, contains the most recent result, or many results through accumulation mode results written to hy are formatted according to the currently chosen number format
hyin	contains the most recent calculation or RUN command input line

Results can be written to a buffer instead of the file hy — see “List of Commands — Buffer”, page 10

The commands SAVE filename and SAVE@ filename write variables and user defined elements to files with the specified names, from where they can be restored with _filename.

Insert Files:

The content of a file can be inserted into an input line by writing its name in parentheses, (filename)

Script Files:

Script files are files in which each line is an input line.

Scripts can be used in loops, see “List of Commands — Repeat and Loops”, page 11.

RUN filename	executes a script file
_filename	short for RUN filename, but only end result will be displayed

List of Commands

Mode Settings and Output Format Commands

AUTO\$ ON OFF	set auto include \$ mode ON (default) OFF
ECHO ON OFF	set echo mode ON OFF (default)
LOG ON OFF	start stop (default) logging input and output to file hy.log
INTEGER ON OFF	set integer bias ON (default) OFF
USE DEG RAD	angle unit is degrees radians (default) — compare DEG and RAD operators
AUTO\$, ECHO, LOG, INTEGER, USE — show current mode	
FSHORT, FLONG	show up to 9 digits (does not affect FDEC n) up to 15 digits (18 for integers)
FDEC	(default) show results in decimal format, scientific notation for large or small numbers
FDEC n	show results with n digits after decimal point (before decimal point if n is negative)
FSCI	show results in scientific notation, 15 digits (9 digits with FSHORT) and exponent
FHEX	use hexadecimal format for displaying results (only valid for positive integer numbers)
FHEX n	use hexadecimal format, n digits (or more if the number is larger, min. 2, max. 12)
FBIN	use binary format for displaying results (only valid for positive integer numbers)
FBIN n	binary format, n digits (or more if the number is larger, min. 2, max. 48)
FLAKH	decimal format, numbers $\geq 1E5$ are shown in lakh, numbers $\geq 1E7$ in crore
FMILLION	decimal format, numbers $\geq 1E6$ are shown in million
F' ON OFF	apostrophe format ON OFF (F' toggles apostrophe format ON/OFF)
RESET	delete all variables, reset all options to default or to command line parameters

For accumulation mode, silent mode and debug mode see “List of Control Symbols” (page 12).

Variables

STO \$var	assign value of last result to variable (create variable if it doesn't already exist)
\$var = ...	assign number or calculation result to variable (create if it doesn't already exist)
PROMPT \$var [comment]	prompt user for value of variable (number or calculation), comment text will be displayed
DEL \$var	delete variable
SHOW	display all variables; if angle unit is set to degrees, this will be displayed
SHOW \$var1 \$var2 ...	display these variables (can include loop index I and loop timer TIME)
SAVE filename [comment]	save variables to file (anything after filename is a comment line)
_filename	or RUN filename: retrieve variables from file

Results

=	show last result (value of \$) in currently chosen format
==	show last result (value of \$) in decimal format with up to 18 digits
HY	show content of result file hy
\$	write last result (value of \$) to hy in the currently specified format

User Defined Elements

@ude = ...	assign numbers and/or operators to UDE (create UDE if it doesn't exist)
DEL@ @ude	delete user defined element
SHOW@	display all user defined elements
SHOW@ @ude1 @ude2 ...	display these user defined elements
SAVE@ filename	save user defined elements to file (anything after filename is a comment line)
_filename	or RUN filename: retrieve user defined elements from file

Clipboard

COPY	copy the result file hy to the clipboard
COPYALL ON	set copy to clipboard mode ON (all results will be copied to the clipboard)
COPYALL OFF	set copy to clipboard mode OFF (default)
COPYALL	show current copyall mode
COPIN	copy last calculation input line to clipboard

Files

FILES	show all files in Hypatia's program folder
EDIT	open result file hy to view or edit (short for EDIT hy)
EDIT filename	open editor to view, edit or create a file in Hypatia's program folder
EDIN	open editor to view or edit file hyin (short for EDIT hyin)
EDINI	open editor to view or edit file hy.ini (short for EDIT hy.ini)
	default editor is Windows Notepad (notepad.exe)
EXTEDITOR filename	replace notepad.exe with an editor of your choice
RUN filename	execute a script file, each line is treated as an input line
_filename	same as RUN filename, but without displaying intermediate results
(filename)	insert content of file in input line
&	clear result file hy (if buffer mode is on, clear buffer)
&&	add a line break to result file hy (if buffer mode is on, to buffer)
HY	show content of result file hy

Buffer

BUFFER START	turn buffer mode on, clear buffer
BUFFER SHOW	display buffer content
BUFFER SAVE filename [comment]	save buffer to file (anything after filename is a comment line)
BUFFER FLUSH	write buffer to hy, clear buffer, turn buffer mode off
BUFFER DISCARD	delete content of buffer and turn buffer mode off
BUFFER	display buffer mode (ON, ON but empty, or OFF)
(buffer)	if buffer mode is on, insert content of buffer in input line

Scripts

RUN filename	execute script file
_filename	same as RUN filename, but only final result will be shown

Repeat and Loops

REPEAT	repeat the most recent calculation
MAXLOOP n	set max. number of passes in a loop (by default 100000, max. 1E7)
REPEAT n	loop command, repeat the most recent calculation n times
REPEAT *	loop command, n = maximum (by default 100000) minus 1
DO n :: ...	loop command
DO * ::	loop command, n = maximum (by default 100000)
* _filename	short for DO * :: _filename (run script in loop with max. n of passes)
I1: ...	loops are exited when variable \$loop is set to 0 within the loop
I*: ...	execute line only when loop index is 1, or when not in a loop
IF ... THEN ...	execute line only at the end of loop
IF ... THEN ENDLOOP	execute command or calculation only when condition is met
IF ... THEN SKIP	exit loop after last line in the script when condition is met
IF ... THEN ABORT	skip the rest of the script when condition is met
ALSO: ...	combines ENDLOOP and SKIP
ELSE: ...	skip the rest of the script when condition is met, then exit loop
\$loop =	execute when preceding IF/THEN condition was met
	execute when preceding IF/THEN condition was not met
	setting variable \$loop to 0 exits the loop, the same as ENDLOOP

IF does *not* observe the zero threshold, it only returns false for values exactly zero.

\$loop only exits the loop when its value is exactly zero.

Help

	display program version and basic help info
?	display program version and settings
HELP	display help overview
HELP ?	display available help topics

Quit

Q or QUIT or EXIT quits the calculator.

You can also just close the console window, Hypatia keeps no files open that could be corrupted.

List of Control Symbols

comment line (will not be displayed in a loop)

Comments following I1:, I*:, IF ... THEN, ALSO: or ELSE: will always be displayed if true

#: comment line

Comment lines in a script that start with #: will also be displayed in loops

comment line

Comment lines in a script that start with ## will not be displayed, except in echo mode

... ## comment

Anything in a command or calculation line after ## is a comment

calculation line **&**

& at the end of a calculation line sets accumulation mode for this line:

instead of overwriting hy the present result will be appended, separated by a space

calculation line **&&**

Same as &, but sets “new line” accumulation mode for this line:

the new result will be added to hy in a new line

calculation line **#**

at the end of a calculation line sets silent mode for this line:

result file hy will not be updated

calculation line **?**

Debug mode, intermediate results will be shown

Permitted combinations (&& can be used instead of &):

... ? # ... ? &... ... & ?

Accumulation, silent and debug modes are only set for the particular line

In scripts hy only gets updated at the end, or by lines with accumulation mode set

& or # can be used in the script's last calculation line

REPEAT n ?

DO n ? :: ...

Show results for each loop pass

_filename

Same as RUN filename, but only final result will be shown

... (filename) ...

Insert content of file into the input line

Do not confuse the & and && control symbols at the end of calculation lines with the commands & and && (& clears the file hy, && adds a line break), nor with the &h and &b notation of hexadecimal and binary numbers.

Do also not confuse the control symbol ? at the end of a calculation line, or that after the REPEAT command, with the command ? which shows current program settings.