GNU Calc Reference Card

(for version 2.03 of November 2001)

Starting and Stopping

start/stop standard Calc	`M-# c`
start/stop X keypad Calc	`M-# k`
start/stop either: `M-# #` or `M-# M-#`	
stop standard Calc	`q`
Calc tutorial	`M-# t`
run Calc in other window	`M-# o`
quick calculation in minibuffer	`M-# q`

Getting Help

The `h` prefix key is Calc's analogue of `C-h` in Emacs.

quick summary of keys	`?`
describe key briefly	`h c`
describe key fully	`h k`
describe function or command	`h f`
read on-line manual	`h i` or `M-# i`
read full Calc summary	`h s` or `M-# s`

Error Recovery

abort command in progress	`C-g`
display recent error messages	`w`
undo last operation	`U`
redo last operation	`D`
recall last arguments	`M-RET`
edit top of stack	`` ` ``
reset Calc to default state	`M-# 0` (zero)

Transferring Data

grab region from a buffer	`M-# g`
grab rectangle from a buffer	`M-# r`
grab rectangle, summing columns	`M-# :`
grab rectangle, summing rows	`M-# _`
yank data to a buffer	`M-# y`

Also, try `C-k`/`C-y` or X cut and paste.

Examples

In RPN, enter numbers first, separated by `RET` if necessary, then type the operator. To enter a calculation in algebraic form, press the apostrophe first.

	RPN style:	**algebraic style:**
Example:	`2 RET 3 +`	`' 2+3 RET`
Example:	`2 RET 3 + 4 *`	`' (2+3)*4 RET`
Example:	`2 RET 3 RET 4 + *`	`' 2*(3+4) RET`
Example:	`3 RET 6 + Q 3 ^`	`' sqrt(3+6)^3 RET`
Example:	`P 3 / n S`	`' sin(-pi/3) RET =`

© 2001 Free Software Foundation, Inc. Permissions on back. v2.03

Arithmetic

add, subtract, multiply, divide	`+`, `-`, `*`, `/`
raise to a power, nth root	`^`, `I ^`
change sign	`n`
reciprocal $1/x$	`&`
square root $\sqrt{x}$	`Q`
set precision	`p`
round off last two digits	`c 2`
convert to fraction, float	`c F`, `c f`
enter using algebraic notation	`' 2+3*4`
refer to previous result	`' 3*$^2`
refer to higher stack entries	`' $1*$2^2`
finish alg entry without evaluating	`LFD`
set mode where alg entry used by default	`m a`

Stack Commands

Here S_n is the nth stack entry, and N is the size of the stack.

key	*no prefix*	*prefix n*	*prefix −n*
`RET`	copy S_1	copy $S_{1..n}$	copy S_n
`LFD`	copy S_2	copy S_n	copy $S_{1..n}$
`DEL`	delete S_1	delete $S_{1..n}$	delete S_n
`M-DEL`	delete S_2	delete S_n	delete $S_{1..n}$
`TAB`	swap $S_1 \leftrightarrow S_2$	roll S_1 to S_n	roll S_n to S_N
`M-TAB`	roll S_3 to S_1	roll S_n to S_1	roll S_N to S_n

With a 0 prefix, these copy, delete, or reverse $S_{1..N}$.

Display

scroll horizontally, vertically	`< >`, `{ }`
home cursor	`o`
line numbers on/off	`d l`
trail display on/off	`t d`
scientific notation	`d s`
fixed-point notation	`d f`
floating-point (normal) notation	`d n`
group digits with commas	`d g`

For display mode commands, `H` prefix prevents screen redraw and `I` prefix temporarily redraws top of stack.

Notations

scientific notation	`6.02e23`
minus sign in numeric entry	`_23` or `23 n`
fractions	`3:4`
complex numbers	(*x*, *y*)
polar complex numbers	(*r*; *θ*)
vectors (commas optional)	`[1, 2, 3]`
matrices (or nested vectors)	`[1, 2; 3, 4]`
error forms (`p` key)	`100 +/- 0.5`
interval forms	`[2 .. 5)`
modulo forms (`M` key)	`6 mod 24`
HMS forms	`5@ 30' 0"`
date forms	`<Jul 4, 1992>`
infinity, indeterminate	`inf`, `nan`

Scientific Functions

ln, $\log_{10}$, $\log_b$	`L`, `H L`, `B`
exponential e^x, 10^x	`E`, `H E`
sin, cos, tan	`S`, `C`, `T`
arcsin, arccos, arctan	`I S`, `I C`, `I T`
inverse, hyperbolic prefix keys	`I`, `H`
two-argument arctan	`f T`
degrees, radians modes	`m d`, `m r`
pi (π)	`P`
factorial, double factorial	`!`, `k d`
combinations, permutations	`k c`, `H k c`
prime factorization	`k f`
next prime, previous prime	`k n`, `I k n`
GCD, LCM	`k g`, `k l`
random number, shuffle	`k r`, `k h`
minimum, maximum	`f n`, `f x`
error functions erf, erfc	`f e`, `I f e`
gamma, beta functions	`f g`, `f b`
incomplete gamma, beta functions	`f G`, `f B`
Bessel J_ν, Y_ν functions	`f j`, `f y`
complex magnitude, arg, conjugate	`A`, `G`, `J`
real, imaginary parts	`f r`, `f i`
convert polar/rectangular	`c p`

Financial Functions

enter percentage	`M-%`
convert to percentage	`c %`
percentage change	`b %`
present value	`b P`
future value	`b F`
rate of return	`b T`
number of payments	`b #`
size of payments	`b M`
net present value, int. rate of return	`b N`, `b I`

Above computations assume payments at end of period. Use `I` prefix for beginning of period, or `H` for a lump sum investment.

straight-line depreciation	`b S`
sum-of-years'-digits	`b Y`
double declining balance	`b D`

Units

enter with units	`' 55 mi/hr`
convert to new units, base units	`u c`, `u b`
convert temperature units	`u t`
simplify units expression	`u s`
view units table	`u v`

Common units:

distance: `m`, `cm`, `mm`, `km`; `in`, `ft`, `mi`, `mfi`; `point`, `lyr`
volume: `l` or `L`, `ml`; `gal`, `qt`, `pt`, `cup`, `floz`, `tbsp`, `tsp`
mass: `g`, `mg`, `kg`, `t`; `lb`, `oz`, `ton`
time: `s` or `sec`, `ms`, `us`, `ns`, `min`, `hr`, `day`, `wk`
temperature: `degC`, `degF`, `K`

GNU Calc Reference Card

Programmer's Functions

binary, octal, hex display	`d 2`, `d 8`, `d 6`
decimal, other radix display	`d 0`, `d r`
display leading zeros	`d z`
entering non-decimal numbers	`16#7FFF`
binary word size	`b w`
binary AND, OR, XOR	`b a`, `b o`, `b x`
binary DIFF, NOT	`b d`, `b n`
left shift	`b l`
logical right shift	`b r`
arithmetic right shift	`b R`
integer quotient, remainder	`\`, `%`
integer square root, logarithm	`f Q`, `f I`
floor, ceiling, round to integer	`F`, `I F`, `R`

Variables

Variable names are single digits or whole words.

store to variable	`s t`
store and keep on stack	`s s`
recall from variable	`s r`
shorthands for digit variables	`t` *n*, `s` *n*, `r` *n*
unstore, exchange variable	`s u`, `s x`
edit variable	`s e`

Vector Operations

vector of 1, 2, ..., *n*	`v x` *n*
vector of *n* counts from *a* by *b*	`C-u v x`
vector of copies of a value	`v b`
concatenate into vector	`|`
pack many stack items into vector	`v p`
unpack vector or object	`v u`
length of vector (list)	`v l`
reverse vector	`v v`
sort, grade vector	`V S`, `V G`
histogram of vector data	`V H`
extract vector element	`v r`
matrix determinant, inverse	`V D`, `&`
matrix transpose, trace	`v t`, `V T`
cross, dot products	`V C`, `*`
identity matrix	`v i`
extract matrix row, column	`v r`, `v c`
intersection, union, diff of sets	`V ^`, `V V`, `V -`
cardinality of set	`V #`
add vectors elementwise (i.e., map `+`)	`V M +`
sum elements in vector (i.e., reduce `+`)	`V R +`
sum rows in matrix	`V R _ +`
sum columns in matrix	`V R : +`
sum elements, accumulate results	`V U +`

Algebra

enter an algebraic formula	`' 2x+3y^2`
enter an equation	`' 2x^2=18`
symbolic (vs. numeric) mode	`m s`
fractions (vs. float) mode	`m f`
suppress evaluation of formulas	`m O`
simplify formulas automatically	`m S`
return to default evaluation rules	`m D`
"Big" display mode	`d B`
C, Pascal, FORTRAN modes	`d C`, `d P`, `d F`
TEX, eqn modes	`d T`, `d E`
Unformatted mode	`d U`
Normal language mode	`d N`
simplify formula	`a s`
put formula into rational form	`a n`
evaluate variables in formula	`=`
evaluate numerically	`N`
let variable equal a value in formula	`s l` *x*`=`*val*
declare properties of variable	`s d`
Common decls: `pos`, `int`, `real`, `scalar`, [*a*..*b*].	
expand, collect terms	`a x`, `a c`
factor, partial fractions	`a f`, `a a`
polynomial quotient, remainder, GCD	`a \`, `a %`, `a g`
derivative, integral	`a d`, `a i`
taylor series	`a t`
principal solution to equation(s)	`a S`
list of solutions	`a P`
generic solution	`H a S`
apply function to both sides of eqn	`a M`
rewrite formula	`a r`
Example: `a r a*b + a*c := a*(b+c)`	
Example: `a r sin(x)^2 := 1-cos(x)^2`	
Example: `a r cos(n pi) := 1 :: integer(n) :: n%2 = 0`	
Example: `a r [f(0) := 1, f(n) := n f(n-1) :: n > 0]`	
Put rules in `EvalRules` to have them apply automatically.	
Put rules in `AlgSimpRules` to apply during `a s` command.	
Common markers: `opt`, `plain`, `quote`, `eval`, `let`, `remember`.	

Numerical Computations

sum formula over a range	`a +`
product of formula over a range	`a *`
tabulate formula over a range	`a T`
integrate numerically over a range	`a I`
find zero of formula or equation	`a R`
find local min, max of formula	`a N`, `a X`
fit data to line or curve	`a F`
mean of data in vector or variable	`u M`
median of data	`H u M`
geometric mean of data	`u G`
sum, product of data	`u +`, `u *`
minimum, maximum of data	`u N`, `u X`
sample, pop. standard deviation	`u S`, `I u S`

Selections

select subformula under cursor	`j s`
select *n*th subformula	`j` *n*
select more	`j m`
unselect this, all formulas	`j u`, `j c`
copy indicated subformula	`j RET`
delete indicated subformula	`j DEL`
commute selected terms	`j C`
commute term leftward, rightward	`j L`, `j R`
distribute, merge selection	`j D`, `j M`
isolate selected term in equation	`j I`
negate, invert term in context	`j N`, `j &`
rewrite selected term	`j r`

Graphics

graph function or data	`g f`
graph 3D function or data	`g F`
replot current graph	`g p`
print current graph	`g P`
add curve to graph	`g a`
set number of data points	`g N`
set line, point styles	`g s`, `g S`
set log vs. linear *x*, *y* axis	`g l`, `g L`
set range for *x*, *y* axis	`g r`, `g R`
close graphics window	`g q`

Programming

begin, end recording a macro	`C-x (`, `C-x )`
replay keyboard macro	`X`
read region as written-out macro	`M-# m`
if, else, endif	`Z [`, `Z :`, `Z ]`
equal to, less than, member of	`a =`, `a <`, `a {`
repeat *n* times, break from loop	`Z <`, `Z >`, `Z /`
"for" loop: start, end; body, step	`Z (`, `Z )`
save, restore mode settings	`Z ‘`, `Z ’`
display message during macro	`Z =`
query user during macro	`Z #`
put finished macro on a key	`Z K`
define function with formula	`Z F`
edit definition	`Z E`
record user-defined command permanently	`Z P`
record variable value permanently	`s p`
record mode settings permanently	`m m`

Copyright © 2001 Free Software Foundation, Inc.
designed by Dave Gillespie and Stephen Gildea, v2.03
for GNU Emacs Calc version 2.03 (November 2001)

Permission is granted to make and distribute copies of this card provided the copyright notice and this permission notice are preserved on all copies.

For copies of the GNU Emacs Calc manual, write to the Free Software Foundation, Inc., 675 Massachusetts Ave, Cambridge MA 02139.