bc Precision calculator

CONCEPTS

Division truncates to scale digits and scale starts at 0, so 10/3 prints 3. Set scale=N first, or use -l (scale 20).

Input read in ibase
↓ compute
Arbitrary precision scale decides digits
↓ print
Output shown in obase

Set obase before ibase when both change: after ibase=16, the number 10 means sixteen. Use A for ten while ibase is 16.

RUNNING BC

bc
Start interactive calculator

bc -l
Math library and scale 20

echo '<expr>' | bc -l
Evaluate an expression from a pipe

bc -e '<expr>'
Evaluate an expression argument

bc <file>
Run a script file, then read stdin

bc -f <file>
Run a file, exit afterwards

bc -L
Do not wrap long output lines

PRECISION AND BASES

scale=<n>
Digits kept after the decimal point

obase=<n>
Output base (2 to max)

ibase=<n>
Input base (2 to 36)

obase=16; <n>
Convert decimal to hex

ibase=16; <HEX>
Read hex as input (uppercase digits)

last or .
Last printed result

OPERATORS

+ - * / %
Basic arithmetic and remainder

^
Power with integer exponent

sqrt(<x>)
Square root

++x x-- x+=1
Increment and compound assignment

MATH LIBRARY (-L)

s(<x>)
Sine in radians

c(<x>)
Cosine in radians

a(<x>)
Arctangent in radians

l(<x>)
Natural logarithm

e(<x>)
e to the power x

4*a(1)
Pi to current scale

l(<x>)/l(10)
Log base 10 (or l10(x))

p(<x>, <y>)
Power with non-integer exponent

r(<x>, <p>)
Round to p places

PROGRAMMING

define <f>(<a>, <b>) { ... }
Define a function

auto <i>, <j>
Local variables (first line in function)

return (<x>)
Return a value

if (<c>) { } else { }
Conditional

while (<c>) { ... }
Loop while true

for (i=0; i<10; i++) { ... }
Counting loop

quit
Exit bc