bc Precision calculator

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

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).

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

This sheet covers Gavin Howard's bc (BSD, macOS, many Linux distros). GNU bc lacks some extensions marked here.

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

RUNNING BC (cont.)

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

bc -q
Skip banner (GNU compatibility)

bc -z
Print leading zero (0.5, not .5)

bc -L
Do not wrap long output lines

PRECISION AND BASES

scale=*<n>*
Digits kept after the decimal point

scale
Show current scale

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

== != < <= > >=
Comparisons giving 1 or 0

&& || !
Boolean and, or, not

abs(*<x>***)**
Absolute value

length(*<x>***)**
Number of significant digits

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

MATH LIBRARY (-L) (cont.)

4*a(1)
Pi to current scale

pi(*<p>***)**
Pi at precision p

t(*<x>***)**
Tangent

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

log(*<x>***, *<b>*)**
Logarithm in any base

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

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

root(*<x>***, *<n>*)**
cbrt(*<x>***)**
Nth root; cube root

f(*<n>***)**
Factorial

gcd(*<a>***, *<b>*)** **lcm(***<a>***, *<b>*)**
Greatest common divisor; least common multiple

max(*<a>***, *<b>*)** **min(***<a>***, *<b>*)**
Larger or smaller of two

hex(*<x>***)** **binary(***<x>***)**
Print integer as hex or binary

PROGRAMMING

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

define void *<f>*()**
{ ... }
Function that returns no value

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

break **continue**
Leave or skip a loop iteration

a[*<i>***] = *<x>***
Array element (zero-based)

print "*<text>***", *<x>*, "\n"**
Print without implicit newline

"*<text>***"**
String literal prints as is

quit
Exit bc