

**BC(I)**

**BC(I)**

NAME

bc — arbitrary precision interactive language

SYNOPSIS

**bc** [ **-l** ] [ file ... ]

DESCRIPTION

*Bc* is an interactive processor for a language which resembles C but provides unlimited precision arithmetic. It takes input from any files given, then reads the standard input. The '**-l**' argument stands for the name of a library of mathematical subroutines which contains sine (named 's'), cosine ('c'), arctangent ('a'), natural logarithm ('l'), and exponential ('e'). The syntax for *bc* programs is as follows; E means expression, S means statement.

Comments

are enclosed in /\* and \*/.

Names

letters a-z  
array elements: letter[E]  
the words 'ibase', 'obase', and 'scale'

Other operands

arbitrarily long numbers with optional sign and decimal point  
( E )  
sqrt ( E )  
<letter> ( E , ... , E )

Operators

+ - \* / % ^  
++ -- (prefix and postfix; apply to names)  
== <= >= != < >  
= += -= \*= /= %= ^

Statements

E  
{ S ; ... ; S }  
if ( E ) S  
while ( E ) S  
for ( E ; E ; E ) S  
null statement  
break  
quit

Function definitions are exemplified by

```
define <letter> ( <letter> , ..., <letter> ) {  
    auto <letter> , ... , <letter>  
    S ; ... S  
    return ( E )  
}
```

All function arguments are passed by value.

The value of a statement that is an expression is printed unless the main operator is an assignment. Either semicolons or newlines may separate statements. Assignment to *scale* influences the number of digits to be retained on arithmetic operations. Assignments to *ibase* or *obase* set the input and output number radix respectively.

**BC(I)**

**BC(I)**

The same letter may be used as an array name, a function name, and a simple variable simultaneously. 'Auto' variables are saved and restored during function calls. All other variables are global to the program. When using arrays as function arguments or defining them as automatic variables empty square brackets must follow the array name.

For example

```
scale = 20
define e(x){
    auto a, b, c, i, s
    a = 1
    b = 1
    s = 1
    for(i=1; 1; i++){
        a = a*x
        b = b*i
        c = a/b
        if(c == 0) return(s)
        s = s+c
    }
}
```

defines a function to compute an approximate value of the exponential function and

```
for(i=1; i<=10; i++) e(i)
```

prints approximate values of the exponential function of the first ten integers.

**FILES**

/usr/lib/lib.b                    mathematical library

**SEE ALSO**

*dc* (I), C Reference Manual, "BC - An Arbitrary Precision Desk-Calculator Language."

**BUGS**

No &&, || yet.

*for* statement must have all three E's

*quit* is interpreted when read, not when executed.