

NAME

dc – desk calculator

SYNOPSIS

dc [file]

DESCRIPTION

Dc is an arbitrary precision integer arithmetic package. The overall structure of *dc* is a stacking (reverse Polish) calculator. The following constructions are recognized by the calculator:

- number** The value of the number is pushed on the stack. A number is an unbroken string of the digits 0-9. It may be preceded by an underscore `_` to input a negative number.
- + - * / % ^** The top two values on the stack are added (+), subtracted (-), multiplied (*), divided (/), remaindered (%), or exponentiated (^). The two entries are popped off the stack; the result is pushed on the stack in their place.
- sx** The top of the stack is popped and stored into a register named *x*, where *x* may be any character.
- lx** The value in register *x* is pushed on the stack. The register *x* is not altered. All registers start with zero value.
- d** The top value on the stack is pushed on the stack. Thus the top value is duplicated.
- p** The top value on the stack is printed. The top value remains unchanged.
- f** All values on the stack and in registers are printed.
- q** exits the program. If executing a string, the nesting level is popped by two.
- x** treats the top element of the stack as a character string and executes it as a string of dc commands.
- [...]** puts the bracketed ascii string onto the top of the stack.
- <x =x >x** The top two elements of the stack are popped and compared. Register *x* is executed if they obey the stated relation.
- v** replaces the top element on the stack by its square root.
- !** interprets the rest of the line as a UNIX command.
- c** All values on the stack are popped.
- i** The top value on the stack is popped and used as the number radix for further input.
- o** The top value on the stack is popped and used as the number radix for further output.
- z** The stack level is pushed onto the stack.
- ?** A line of input is taken from the input source (usually the console) and executed.
- new-line** ignored except as the name of a register or to end the response to a **?**.
- space** ignored except as the name of a register or to terminate a number.

If a file name is given, input is taken from that file until end-of-file, then input is taken from the console. An example which prints the first ten values of *n!* is

```
[!a1+dsa*pla10>x]sx
0sa1
lxx
```

FILES

/etc/msh to implement **!**

DIAGNOSTICS

- (x) ? for unrecognized character x.
- (x) ? for not enough elements on the stack to do what was asked by command x.
- 'Out of space' when the free list is exhausted (too many digits).
- 'Out of headers' for too many numbers being kept around.
- 'Out of pushdown' for too many items on the stack.
- 'Nesting Depth' for too many levels of nested execution.

BUGS