Command: **m4 – macro processor**
Syntax: **m4** [**–D** *name* = *value*] [**–U** *name*]
Flags: **–D** Define a symbol
–U Undefine a symbol
Example: m4 <m4test # Run M4

M4 is a macro processor intended as a front end for Ratfor, Pascal, and other languages that do not have a built-in macro processing capability. M4 reads standard input, the processed text is written on the standard output.

The options and their effects are as follows:

–D name[=val]	Defines name to val, or to null in val's absence.
–U name	Undefines name.

Macro calls have the form: name(arg1,arg2, ..., argn)

The ‘‘(’’ must immediately follow the name of the macro. If the name of a defined macro is not followed by a (it is taken to be a call of that macro with no arguments, i.e. name(). Potential macro names consist of alphabetic letters and digits.

Leading unquoted blanks, tabs and newlines are ignored while collecting arguments. Left and right single quotes are used to quote strings. The value of a quoted string is the string stripped of the quotes.

When a macro name is recognized, its arguments are collected by searching for a matching). If fewer arguments are supplied than are in the macro definition, the trailing arguments are taken to be null. Macro evaluation proceeds normally during the collection of the arguments, and any commas or right parentheses which happen to turn up within the value of a nested call are as effective as those in the original input text. (This is typically referred as inside-out macro expansion.) After argument collection, the value of the macro is pushed back onto the input stream and rescanned.

M4 makes available the following built-in macros. They may be redefined, but once this is done the original meaning is lost. Their values are null unless otherwise stated.

define "(name [, val])" the second argument is installed as the value of the macro whose name is the first argument. If there is no second argument, the value is null. Each occurrence of $ n in the replacement text, where n is a digit, is replaced by the n -th argument. Argument 0 is the name of the macro; missing arguments are replaced by the null string.

defn "(name [, name ...])" returns the quoted definition of its argument(s). Useful in renaming macros.

undefine "(name [, name ...])" removes the definition of the macro(s) named. If there is more than one definition for the named macro, (due to previous use of pushdef) all definitions are removed.

pushdef "(name [, val])" like define, but saves any previous definition by stacking the current definition.

popdef "(name [, name ...])" removes current definition of its argument(s), exposing the previous one if any.

ifdef "(name, if-def [, ifnot-def])" if the first argument is defined, the value is the second argument, otherwise the third. If there is no third argument, the value is null. A word indicating the current operating system is predefined. (e.g. unix or vms).

shift "(arg, arg, arg, ...)" returns all but its first argument. The other arguments are quoted and pushed back with commas in between. The quoting nullifies the effect of the extra scan that will subsequently be performed.

changequote "(lqchar, rqchar)" change quote symbols to the first and second arguments. With no arguments, the quotes are reset back to the default characters. (i.e., ‘’).

changecom "(lcchar, rcchar)" change left and right comment markers from the default # and newline. With no arguments, the comment mechanism is reset back to the default characters. With one argument, the left marker becomes the argument and the right marker becomes newline. With two arguments, both markers are affected.

divert "(divnum)" maintains 10 output streams, numbered 0-9. Initially stream 0 is the current stream. The divert macro changes the current output stream to its (digit-string) argument. Output diverted to a stream other than 0 through 9 is lost.

undivert "([divnum [, divnum ...]])" causes immediate output of text from diversions named as argument(s), or all diversions if no argument. Text may be undiverted into another diversion. Undiverting discards the diverted text. At the end of input processing, M4 forces an automatic undivert unless is defined.

divnum "()" returns the value of the current output stream.

dnl "()" reads and discards characters up to and including the next newline.

ifelse "(arg, arg, if-same [, ifnot-same | arg, arg ...])" has three or more arguments. If the first argument is the same string as the second, then the value is the third argument. If not, and if there are more than four arguments, the process is repeated with arguments 4, 5, 6 and 7. Otherwise, the value is either the fourth string, or, if it is not present, null.

incr "(num)" returns the value of its argument incremented by 1. The value of the argument is calculated by interpreting an initial digit-string as a decimal number.

decr "(num)" returns the value of its argument decremented by 1.

eval "(expression)" evaluates its argument as a constant expression, using integer arithmetic. The evaluation mechanism is very similar to that of cpp (#if expression). The expression can involve only integer constants and character constants, possibly connected by the binary operators

* / % + - >> << < > <= >= == != & ˆ | && ||

or the unary operators - ! or tilde or by the ternary operator ? : . Parentheses may be used for grouping. Octal numbers may be specified as in C.

len "(string)" returns the number of characters in its argument.

index "(search-string, string)" returns the position in its first argument where the second argument begins (zero origin), or 1 if the second argument does not occur.

substr "(string, index [, length])" returns a substring of its first argument. The second argument is a zero origin number selecting the first character (internally treated as an expression); the third argument indicates the length of the substring. A missing third argument is taken to be large enough to extend to the end of the first string.

translit "(source, from [, to])" transliterates the characters in its first argument from the set given by the second argument to the set given by the third. If the third argument is shorter than the second, all extra characters in the second argument are deleted from the first argument. If the third argument is missing altogether, all characters in the second argument are deleted from the first argument.

include "(filename)" returns the contents of the file that is named in the argument.

sinclude "(filename)" is identical to include, except that it says nothing if the file is inaccessable.

paste "(filename)" returns the contents of the file named in the argument without any processing, unlike include.

spaste "(filename)" is identical to paste, except that it says nothing if the file is inaccessibl[De.

syscmd "(command)" executes the UNIX command given in the first argument. No value is returned.

sysval "()" is the return code from the last call to syscmd.
.PP **maketemp "(string)"** fills in a string of XXXXXX in its argument with the current process ID.

m4exit "([exitcode])" causes immediate exit from M4. Argument 1, if given, is the exit code; the default is 0.

m4wrap "(m4-macro-or-built-n)" argument 1 will be pushed back at final EOF; example: m4wrap(‘dumptable()’).

errprint "(str [, str, str, ...])" prints its argument(s) on stderr. If there is more than one argument, each argument is separated by a space during the output. An arbitrary number of arguments may be supplied.

dumpdef "([name, name, ...])" prints current names and definitions, for the named items, or for all if no arguments are given.

Author

M4 was written by Ozan S. Yigif.