NAME

match − searches for patterns in files

SYNOPSIS

match [−*option*] *pattern* [*file* ...]

DESCRIPTION

Match searches the named files or standard input (if no filenames are given) for the occurrences of the given pattern on each line. The program accepts literal characters or special pattern matching characters. All lines that match the pattern are output on standard output. You can only specify one pattern string for each match, however, you can construct an arbitrarily complex string. When you do not specify a file, match can be used as a filter to display desired lines. Standard in is used if no files are specified.

OPTIONS

−not, **−v**
Prints all lines that do not match.

−i Ignore the case of letters

−m Force not to use the magic mode

−w Search for *pattern* as a word

−x Display only those lines which match exactly

−c Display matching count for each file

−l Display name of each file which matches

−s Be silent indicate match in exit code

−h Do not display filenames

−n Precede matching lines with line number (with respect to the input file)

−b Precede matching lines with block number

REGULAR EXPRESSIONS

The following is a table of all the pattern matching characters:

c An ordinary character *(not* one of the special characters discussed below) is a one character regular expression that matches that character.

*****c* A backslash (\\) followed by any special character is a one character regular expression that matches the special character itself. The special characters are:

! # % * { } [ ] \ ? ˆ $

! Logical **OR** as in match *this!that!the_other*. You may have to use ‘{}’ for precedence grouping.

A hash mark followed by any regular expression matches any number (including zero) occurrences of the regular expression.

? Matches exactly any one character. **W?** matches *Wa, Wb, Wc, W1, W2, W3* ...

***** Matches any number of any character.

% Matches exactly nothing. It can be used in groups of ored patterns to specify that an empty alternative is possible.

{} Curly brackets may be used to enclose patterns to specify a precedence grouping, and may be nested. **{%!{test}}version** matches the strings *testversion* and *version.*

[*string***]** A non empty string of characters enclosed in square brackets is a one character regular expression that matches *any one* character in that string. If however the first character of the string is a circumflex (ˆ), the one character expression matches any character which is *not* in the string. The ˆ has this special meaning *only* if it occurs first in the string. The minus (−) may be used to indicate a range of consecutive ASCII characters; for example, [0-9] is equivalent to *any one of the digits.* The − loses it’s special meaning if it occurs first (after an initial ˆ, if any) or last in the string. The right square bracket (]) and the backslash (\) must be quoted with a backslash if you want to use it within the *string*.

ˆ Matches the beginning of a line.

**$** Matches the end of a line. (ˆ*$ matches any entire line)

EXAMPLES

FILES

None.

SEE ALSO

grep(1), fgrep(1), egrep(1)

DIAGNOSTICS

NOTES

Even if a match occurs more than once per line, the line is output only once.

Quote special pattern matching characters to prevent them from being expanded by the Command Interpreter.

BUGS

The length of the pattern is currently limited to 100 characters.

This limit is reduced by 38 if the **−w** option is used.