NAME

grep, egrep, fgrep - print lines matching a pattern

SYNOPSIS

grep [options] PATTERN [FILE...]
grep [options] [-e PATTERN | -f FILE] [FILE...]

DESCRIPTION

Grep searches the named input FILEs (or standard input if no files are named, or the file name - is given) for lines containing a match to the given PATTERN. By default, grep prints the matching lines.

In addition, two variant programs egrep and fgrep are available. Egrep is the same as grep -E. Fgrep is the same as grep -F.

OPTIONS

-A NUM, --after-context=NUM
Print NUM lines of trailing context after matching lines.

-a, --text
Process a binary file as if it were text; this is equivalent to the --binary-files=text option.

-B NUM, --before-context=NUM
Print NUM lines of leading context before matching lines.

-C [NUM], -NUM, --context[=NUM]
Print NUM lines (default 2) of output context.

-b, --byte-offset
Print the byte offset within the input file before each line of output.

--binary-files=TYPE
If the first few bytes of a file indicate that the file contains binary data, assume that the file is of type TYPE. By default, TYPE is binary, and grep normally outputs either a one-line message saying that a binary file matches, or no message if there is no match. If TYPE is without-match, grep assumes that a binary file does not match; this is equivalent to the -I option. If TYPE is text, grep processes a binary file as if it were text; this is equivalent to the -a option. Warning: grep --binary-files=text might output binary garbage, which can have nasty side effects if the output is a terminal and if the terminal driver interprets some of it as commands.

-c, --count
Suppress normal output; instead print a count of matching lines for each input file. With the -v, --invert-match option (see below), count non-matching lines.

-d ACTION, --directories=ACTION
If an input file is a directory, use ACTION to process it. By default, ACTION is read, which means that directories are read just as if they were ordinary files. If ACTION is skip, directories are silently skipped. If ACTION is recurse, grep reads all files under each directory, recursively; this is equivalent to the -r option.

-E, --extended-regexp
Interpret PATTERN as an extended regular expression (see below).

-e PATTERN, --regexp=PATTERN
Use PATTERN as the pattern; useful to protect patterns beginning with -.

-F, --fixed-strings
Interpret PATTERN as a list of fixed strings, separated by newlines, any of which is to be matched.

-f FILE, --file=FILE
Obtain patterns from FILE, one per line. The empty file contains zero patterns, and therefore matches nothing.

-G, --basic-regexp
Interpret PATTERN as a basic regular expression (see below). This is the default.

-H, --with-filename
Print the filename for each match.

-h, --no-filename
Suppress the prefixing of filenames on output when multiple files are searched.

--help
Output a brief help message.

-I Process a binary file as if it did not contain matching data; this is equivalent to the - - binary-files=without-match option.

-i, --ignore-case
Ignore case distinctions in both the PATTERN and the input files.

-L, --files-without-match
Suppress normal output; instead print the name of each input file from which no output would normally have been printed. The scanning will stop on the first match.

-l, --files-with-matches
Suppress normal output; instead print the name of each input file from which output would normally have been printed. The scanning will stop on the first match.

--mmap
If possible, use the mmap(2) system call to read input, instead of the default read(2) system call. In some situations, --mmap yields better performance. However, - - mmap can cause undefined behavior (including core dumps) if an input file shrinks while grep is operating, or if an I/O error occurs.

-n, --line-number
Prefix each line of output with the line number within its input file.

-q, --quiet, --silent
Quiet; suppress normal output. The scanning will stop on the first match. Also see the -s or --no-messages option below.

-r, --recursive
Read all files under each directory, recursively; this is equivalent to the -d recurse option.

-s, --no-messages
Suppress error messages about nonexistent or unreadable files. Portability note: unlike GNU grep, traditional grep did not conform to POSIX.2, because traditional grep lacked a -q option and its -s option behaved like GNU grep's -q option. Shell scripts intended to be portable to traditional grep should avoid both -q and -s and should redirect output to /dev/null instead.

-U, --binary
Treat the file(s) as binary. By default, under MS-DOS and MS-Windows, grep guesses the file type by looking at the contents of the first 32KB read from the file. If grep decides the file is a text file, it strips the CR characters from the original file contents (to make regular expressions with ^ and $ work correctly). Specifying -U overrules this guesswork, causing all files to be read and passed to the matching mechanism verbatim; if the file is a text file with CR/LF pairs at the end of each line, this will cause some regular

expressions to fail. This option has no effect on platforms other than MS-DOS and MS-Windows.

-u, --unix-byte-offsets
Report Unix-style byte offsets. This switch causes grep to report byte offsets as if the file were Unix-style text file, i.e. with CR characters stripped off. This will produce results identical to running grep on a Unix machine. This option has no effect unless - b option is also used; it has no effect on platforms other than MS-DOS and MS-Windows.

-V, --version
Print the version number of grep to standard error. This version number should be included in all bug reports (see below).

-v, --invert-match
Invert the sense of matching, to select non-matching lines.

-w, --word-regexp
Select only those lines containing matches that form whole words. The test is that the matching substring must either be at the beginning of the line, or preceded by a non-word constituent character. Similarly, it must be either at the end of the line or followed by a non-word constituent character. Word-constituent characters are letters, digits, and the underscore.

-x, --line-regexp
Select only those matches that exactly match the whole line.

-y Obsolete synonym for -i.

-Z, --null
Output a zero byte (the ASCII NUL character) instead of the character that normally follows a file name. For example, grep -lZ outputs a zero byte after each file name instead of the usual newline. This option makes the output unambiguous, even in the presence of file names containing unusual characters like newlines. This option can be used with commands like find -print0, perl -0, sort -z, and xargs -0 to process arbitrary file names, even those that contain newline characters.

REGULAR EXPRESSIONS
A regular expression is a pattern that describes a set of strings. Regular expressions are constructed analogously to arithmetic expressions, by using various operators to

combine smaller expressions.

Grep understands two different versions of regular expression syntax: "basic" and "extended." In GNU grep, there is no difference in available functionality using either syntax. In other implementations, basic regular expressions are less powerful. The following description applies to extended regular expressions; differences for basic regular expressions are summarized afterwards.

The fundamental building blocks are the regular expressions that match a single character. Most characters, including all letters and digits, are regular expressions that match themselves. Any metacharacter with special meaning may be quoted by preceding it with a backslash.

A list of characters enclosed by [and] matches any single character in that list; if the first character of the list is the caret ^ then it matches any character not in the list. For example, the regular expression [0123456789] matches any single digit. A range of characters may be specified by giving the first and last characters, separated by a hyphen. Finally, certain named classes of characters are predefined. Their names are self explanatory, and they are [:alnum:], [:alpha:], [:cntrl:], [:digit:], [:graph:], [:lower:], [:print:], [:punct:], [:space:], [:upper:], and [:xdigit:]. For example, [[:alnum:]] means [0-9A-Za-z], except the latter form depends upon the POSIX locale and the ASCII character encoding, whereas the former is independent of locale and character set. (Note that the brackets in these class names are part of the symbolic names, and must be included in addition to the brackets delimiting the bracket list.) Most metacharacters lose their special meaning inside lists. To include a literal] place it first in the list. Similarly, to include a literal ^ place it anywhere but first. Finally, to include a literal - place it last.

The period . matches any single character. The symbol 67 a synonym for [[:alnum:]] and W is a synonym for
[^[:alnum]].

The caret ^ and the dollar sign $ are metacharacters that respectively match the empty string at the beginning and end of a line. The symbols < and > respectively match the empty string at the beginning and end of a word. The symbol cthe empty string at the edge of a word, and B
hmatches the empty string provided it's not at the edge of a
eword.
s

A regular expression may be followed by one of several repetition operators:

? The preceding item is optional and matched at most once.
* The preceding item will be matched zero or more times.
+ The preceding item will be matched one or more times.
{n} The preceding item is matched exactly n times.
{n,} The preceding item is matched n or more times.
{n,m}
The preceding item is matched at least n times, but not more than m times.

Two regular expressions may be concatenated; the resulting regular expression matches any string formed by concatenating two substrings that respectively match the concatenated subexpressions.

Two regular expressions may be joined by the infix operator |; the resulting regular expression matches any string matching either subexpression.

Repetition takes precedence over concatenation, which in turn takes precedence over alternation. A whole subexpression may be enclosed in parentheses to override these precedence rules.

The backreference n, where n is a single digit, matches the substring previously matched by the nth parenthesized subexpression of the regular expression.

In basic regular expressions the metacharacters ?, +, {, |, (, and) lose their special meaning; instead use the backslashed versions ?, +, , , and).

Traditional egrep did not support the { metacharacter, and some egrep implementations support instead, so portable scripts should avoid { in egrep patterns and should use [{] to match a literal {.

GNU egrep attempts to support traditional usage by assuming that { is not special if it would be the start of an invalid interval specification. For example, the shell command egrep '{1' searches for the two-character string {1 instead of reporting a syntax error in the regular expression. POSIX.2 allows this behavior as an extension, but portable scripts should avoid it.

ENVIRONMENT VARIABLES

GREP_OPTIONS

This variable specifies default options to be placed in front of any explicit options. For example, if GREP_OPTIONS is ' - - binary-files=without-match - - directories=skip', grep behaves as if the two options - -binary-files=without-match and --directories=skip had

been specified before any explicit options. Option specifications are separated by whitespace. A backslash escapes the next character, so it can be used to specify an option containing whitespace or a backslash.

LC_ALL, LC_MESSAGES, LANG
These variables specify the LC_MESSAGES locale, which determines the language that grep uses for messages. The locale is determined by the first of these variables that is set. American English is used if none of these environment variables are set, or if the message catalog is not installed, or if grep was not compiled with national language support (NLS).

LC_ALL, LC_CTYPE, LANG
These variables specify the LC_CTYPE locale, which determines the type of characters, e.g., which characters are whitespace. The locale is determined by the first of these variables that is set. The POSIX locale is used if none of these environment variables are set, or if the locale catalog is not installed, or if grep was not compiled with national language support (NLS).

POSIXLY_CORRECT
If set, grep behaves as POSIX.2 requires; otherwise, grep behaves more like other GNU programs. POSIX.2 requires that options that follow file names must be treated as file names; by default, such options are permuted to the front of the operand list and are treated as options. Also, POSIX.2 requires that unrecognized options be diagnosed as "illegal", but since they are not really against the law the default is to diagnose them as "invalid". POSIXLY_CORRECT also disables _N_GNU_nonoption_argv_flags_, described below.

_N_GNU_nonoption_argv_flags_
(Here N is grep's numeric process ID.) If the ith character of this environment variable's value is 1, do not consider the ith operand of grep to be an option, even if it appears to be one. A shell can put this variable in the environment for each command it runs, specifying which operands are the results of file name wildcard expansion and therefore should not be treated as options. This behavior is available only with the GNU C library, and only when POSIXLY_CORRECT is not set.

DIAGNOSTICS
Normally, exit status is 0 if matches were found, and 1 if no matches were found. (The -v option inverts the sense of the exit status.) Exit status is 2 if there were syntax

errors in the pattern, inaccessible input files, or other system errors.

BUGS

Email bug reports to bug-gnu-utils@gnu.org. Be sure to include the word "grep" somewhere in the "Subject:" field.

Large repetition counts in the {m,n} construct may cause grep to use lots of memory. In addition, certain other obscure regular expressions require exponential time and space, and may cause grep to run out of memory.

Backreferences are very slow, and may require exponential time.