

NAME

basename – strip directory and suffix from filenames

SYNOPSIS

basename *NAME* [*SUFFIX*]

basename *OPTION*

DESCRIPTION

Print *NAME* with any leading directory components removed. If specified, also remove a trailing *SUFFIX*.

—help display this help and exit

—version

output version information and exit

AUTHOR

Written by FIXME unknown.

REPORTING BUGS

Report bugs to <bug-coreutils@gnu.org>.

COPYRIGHT

Copyright © 2004 Free Software Foundation, Inc.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

SEE ALSO

The full documentation for **basename** is maintained as a Texinfo manual. If the **info** and **basename** programs are properly installed at your site, the command

info basename

should give you access to the complete manual.

NAME

cat – concatenate files and print on the standard output

SYNOPSIS

cat [*OPTION*] [*FILE*]...

DESCRIPTION

Concatenate *FILE*(s), or standard input, to standard output.

-A, --show-all

equivalent to **-vET**

-b, --number-nonblank

number nonblank output lines

-e equivalent to **-vE**

-E, --show-ends

display \$ at end of each line

-n, --number

number all output lines

-s, --squeeze-blank

never more than one single blank line

-t equivalent to **-vT**

-T, --show-tabs

display TAB characters as ^I

-u (ignored)

-v, --show-nonprinting

use ^ and M- notation, except for LFD and TAB

--help display this help and exit

--version

output version information and exit

With no *FILE*, or when *FILE* is –, read standard input.

AUTHOR

Written by Torbjorn Granlund and Richard M. Stallman.

REPORTING BUGS

Report bugs to <bug-coreutils@gnu.org>.

COPYRIGHT

Copyright © 2004 Free Software Foundation, Inc.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

SEE ALSO

The full documentation for **cat** is maintained as a Texinfo manual. If the **info** and **cat** programs are properly installed at your site, the command

info cat

should give you access to the complete manual.

NAME

chgrp – change group ownership

SYNOPSIS

chgrp [*OPTION*]... *GROUP FILE*...

chgrp [*OPTION*]... *--reference=RFILE FILE*...

DESCRIPTION

Change the group of each *FILE* to *GROUP*. With **--reference**, change the group of each *FILE* to that of *RFILE*.

-c, --changes

like verbose but report only when a change is made

--dereference

affect the referent of each symbolic link, rather than the symbolic link itself (this is the default)

-h, --no-dereference

affect each symbolic link instead of any referenced file (useful only on systems that can change the ownership of a symlink)

--no-preserve-root do not treat '/' specially (the default)

--preserve-root

fail to operate recursively on '/'

-f, --silent, --quiet

suppress most error messages

--reference=RFILE

use *RFILE*'s group rather than the specifying *GROUP* value

-R, --recursive

operate on files and directories recursively

-v, --verbose

output a diagnostic for every file processed

The following options modify how a hierarchy is traversed when the **-R** option is also specified. If more than one is specified, only the final one takes effect.

-H if a command line argument is a symbolic link to a directory, traverse it

-L traverse every symbolic link to a directory encountered

-P do not traverse any symbolic links (default)

--help display this help and exit

--version

output version information and exit

AUTHOR

Written by David MacKenzie and Jim Meyering.

REPORTING BUGS

Report bugs to <bug-coreutils@gnu.org>.

COPYRIGHT

Copyright © 2004 Free Software Foundation, Inc.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

SEE ALSO

The full documentation for **chgrp** is maintained as a Texinfo manual. If the **info** and **chgrp** programs are properly installed at your site, the command

info chgrp

should give you access to the complete manual.

NAME

chmod – change file access permissions

SYNOPSIS

chmod [*OPTION*]... *MODE*[,*MODE*]... *FILE*...

chmod [*OPTION*]... *OCTAL-MODE* *FILE*...

chmod [*OPTION*]... --*reference*=*RFILE* *FILE*...

DESCRIPTION

This manual page documents the GNU version of **chmod**. **chmod** changes the permissions of each given file according to *mode*, which can be either a symbolic representation of changes to make, or an octal number representing the bit pattern for the new permissions.

The format of a symbolic mode is '[ugoa...][[+--][rwxXstugo...][...][,...]'. Multiple symbolic operations can be given, separated by commas.

A combination of the letters 'ugoa' controls which users' access to the file will be changed: the user who owns it (u), other users in the file's group (g), other users not in the file's group (o), or all users (a). If none of these are given, the effect is as if 'a' were given, but bits that are set in the umask are not affected.

The operator '+' causes the permissions selected to be added to the existing permissions of each file; '-' causes them to be removed; and '=' causes them to be the only permissions that the file has.

The letters 'rwxXstugo' select the new permissions for the affected users: read (r), write (w), execute (or access for directories) (x), execute only if the file is a directory or already has execute permission for some user (X), set user or group ID on execution (s), sticky (t), the permissions granted to the user who owns the file (u), the permissions granted to other users who are members of the file's group (g), and the permissions granted to users that are in neither of the two preceding categories (o).

A numeric mode is from one to four octal digits (0-7), derived by adding up the bits with values 4, 2, and 1. Any omitted digits are assumed to be leading zeros. The first digit selects the set user ID (4) and set group ID (2) and sticky (1) attributes. The second digit selects permissions for the user who owns the file: read (4), write (2), and execute (1); the third selects permissions for other users in the file's group, with the same values; and the fourth for other users not in the file's group, with the same values.

chmod never changes the permissions of symbolic links; the **chmod** system call cannot change their permissions. This is not a problem since the permissions of symbolic links are never used. However, for each symbolic link listed on the command line, **chmod** changes the permissions of the pointed-to file. In contrast, **chmod** ignores symbolic links encountered during recursive directory traversals.

STICKY FILES

On older Unix systems, the sticky bit caused executable files to be hoarded in swap space. This feature is not useful on modern VM systems, and the Linux kernel ignores the sticky bit on files. Other kernels may use the sticky bit on files for system-defined purposes. On some systems, only the superuser can set the sticky bit on files.

STICKY DIRECTORIES

When the sticky bit is set on a directory, files in that directory may be unlinked or renamed only by root or their owner. Without the sticky bit, anyone able to write to the directory can delete or rename files. The sticky bit is commonly found on directories, such as /tmp, that are world-writable.

OPTIONS

Change the mode of each FILE to MODE.

-c, --changes

like verbose but report only when a change is made

--no-preserve-root

do not treat '/' specially (the default)

--preserve-root

fail to operate recursively on '/'

- f, --silent, --quiet**
suppress most error messages
- v, --verbose**
output a diagnostic for every file processed
- reference=RFILE**
use RFILE's mode instead of MODE values
- R, --recursive**
change files and directories recursively
- help** display this help and exit
- version**
output version information and exit

Each MODE is one or more of the letters ugoa, one of the symbols += and one or more of the letters rwxXstugo.

AUTHOR

Written by David MacKenzie and Jim Meyering.

REPORTING BUGS

Report bugs to <bug-coreutils@gnu.org>.

COPYRIGHT

Copyright © 2004 Free Software Foundation, Inc.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

SEE ALSO

The full documentation for **chmod** is maintained as a Texinfo manual. If the **info** and **chmod** programs are properly installed at your site, the command

info chmod

should give you access to the complete manual.

NAME

chown – change file owner and group

SYNOPSIS

chown [*OPTION*]... [*OWNER*][:*[GROUP]*] *FILE*...

chown [*OPTION*]... *--reference=RFILE* *FILE*...

DESCRIPTION

This manual page documents the GNU version of **chown**. **chown** changes the user and/or group ownership of each given file, according to its first non-option argument, which is interpreted as follows. If only a user name (or numeric user ID) is given, that user is made the owner of each given file, and the files' group is not changed. If the user name is followed by a colon or dot and a group name (or numeric group ID), with no spaces between them, the group ownership of the files is changed as well. If a colon or dot but no group name follows the user name, that user is made the owner of the files and the group of the files is changed to that user's login group. If the colon or dot and group are given, but the user name is omitted, only the group of the files is changed; in this case, **chown** performs the same function as **chgrp**.

OPTIONS

Change the owner and/or group of each *FILE* to *OWNER* and/or *GROUP*. With **--reference**, change the owner and group of each *FILE* to those of *RFILE*.

-c, --changes

like verbose but report only when a change is made

--dereference

affect the referent of each symbolic link, rather than the symbolic link itself (this is the default)

-h, --no-dereference

affect each symbolic link instead of any referenced file (useful only on systems that can change the ownership of a symlink)

--from=CURRENT_OWNER:CURRENT_GROUP

change the owner and/or group of each file only if its current owner and/or group match those specified here. Either may be omitted, in which case a match is not required for the omitted attribute.

--no-preserve-root do not treat '/' specially (the default)

--preserve-root

fail to operate recursively on '/'

-f, --silent, --quiet

suppress most error messages

--reference=RFILE

use *RFILE*'s owner and group rather than the specifying *OWNER:GROUP* values

-R, --recursive

operate on files and directories recursively

-v, --verbose

output a diagnostic for every file processed

The following options modify how a hierarchy is traversed when the **-R** option is also specified. If more than one is specified, only the final one takes effect.

-H if a command line argument is a symbolic link to a directory, traverse it

-L traverse every symbolic link to a directory encountered

-P do not traverse any symbolic links (default)

--help display this help and exit

--version

output version information and exit

Owner is unchanged if missing. Group is unchanged if missing, but changed to login group if implied by a ':' following a symbolic OWNER. OWNER and GROUP may be numeric as well as symbolic.

AUTHOR

Written by David MacKenzie and Jim Meyering.

REPORTING BUGS

Report bugs to <bug-coreutils@gnu.org>.

COPYRIGHT

Copyright © 2004 Free Software Foundation, Inc.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

SEE ALSO

The full documentation for **chown** is maintained as a Texinfo manual. If the **info** and **chown** programs are properly installed at your site, the command

info chown

should give you access to the complete manual.

NAME

chroot – run command or interactive shell with special root directory

SYNOPSIS

chroot *NEWROOT* [*COMMAND*...]

chroot *OPTION*

DESCRIPTION

Run *COMMAND* with root directory set to *NEWROOT*.

—help display this help and exit

—version

output version information and exit

If no command is given, run “\${SHELL} –i” (default: /bin/sh).

AUTHOR

Written by Roland McGrath.

REPORTING BUGS

Report bugs to <bug-coreutils@gnu.org>.

COPYRIGHT

Copyright © 2004 Free Software Foundation, Inc.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

SEE ALSO

The full documentation for **chroot** is maintained as a Texinfo manual. If the **info** and **chroot** programs are properly installed at your site, the command

info chroot

should give you access to the complete manual.

NAME

cksum – checksum and count the bytes in a file

SYNOPSIS

cksum [*FILE*]...

cksum [*OPTION*]

DESCRIPTION

Print CRC checksum and byte counts of each FILE.

--help display this help and exit

--version

output version information and exit

AUTHOR

Written by Q. Frank Xia.

REPORTING BUGS

Report bugs to <bug-coreutils@gnu.org>.

COPYRIGHT

Copyright © 2004 Free Software Foundation, Inc.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

SEE ALSO

The full documentation for **cksum** is maintained as a Texinfo manual. If the **info** and **cksum** programs are properly installed at your site, the command

info cksum

should give you access to the complete manual.

NAME

`comm` – compare two sorted files line by line

SYNOPSIS

comm [*OPTION*]... *FILE1 FILE2*

DESCRIPTION

Compare sorted files *FILE1* and *FILE2* line by line.

With no options, produce three-column output. Column one contains lines unique to *FILE1*, column two contains lines unique to *FILE2*, and column three contains lines common to both files.

- 1** suppress lines unique to *FILE1*
- 2** suppress lines unique to *FILE2*
- 3** suppress lines that appear in both files
- help** display this help and exit
- version**
 output version information and exit

AUTHOR

Written by Richard Stallman and David MacKenzie.

REPORTING BUGS

Report bugs to <bug-coreutils@gnu.org>.

COPYRIGHT

Copyright © 2004 Free Software Foundation, Inc.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

SEE ALSO

The full documentation for **comm** is maintained as a Texinfo manual. If the **info** and **comm** programs are properly installed at your site, the command

info comm

should give you access to the complete manual.

NAME

cp – copy files and directories

SYNOPSIS

cp [*OPTION*]... [-*T*] *SOURCE DEST*
cp [*OPTION*]... *SOURCE... DIRECTORY*
cp [*OPTION*]... -*t* *DIRECTORY SOURCE...*

DESCRIPTION

Copy *SOURCE* to *DEST*, or multiple *SOURCE*(s) to *DIRECTORY*.

Mandatory arguments to long options are mandatory for short options too.

-a, --archive
 same as **-dpR**

--backup[=*CONTROL*]
 make a backup of each existing destination file

-b like **--backup** but does not accept an argument

--copy-contents
 copy contents of special files when recursive

-d same as **--no-dereference --preserve=link**

--no-dereference
 never follow symbolic links

-f, --force
 if an existing destination file cannot be opened, remove it and try again

-i, --interactive
 prompt before overwrite

-H follow command-line symbolic links

-l, --link
 link files instead of copying

-L, --dereference
 always follow symbolic links

-p same as **--preserve=mode,ownership,timestamps**

--preserve[=*ATTR_LIST*]
 preserve the specified attributes (default: mode,ownership,timestamps), if possible additional attributes: links, all

--no-preserve=*ATTR_LIST*
 don't preserve the specified attributes

--parents
 append source path to *DIRECTORY*

-P same as '**--no-dereference**'

-R, -r, --recursive
 copy directories recursively

--remove-destination
 remove each existing destination file before attempting to open it (contrast with **--force**)

--reply={yes,no,query}
 specify how to handle the prompt about an existing destination file

--sparse=*WHEN*
 control creation of sparse files

--strip-trailing-slashes remove any trailing slashes from each *SOURCE* argument

- s, --symbolic-link**
make symbolic links instead of copying
- S, --suffix=*SUFFIX***
override the usual backup suffix
- t, --target-directory=*DIRECTORY***
copy all SOURCE arguments into DIRECTORY
- T, --no-target-directory**
treat DEST as a normal file
- u, --update**
copy only when the SOURCE file is newer than the destination file or when the destination file is missing
- v, --verbose**
explain what is being done
- x, --one-file-system**
stay on this file system
- help** display this help and exit
- version**
output version information and exit

By default, sparse SOURCE files are detected by a crude heuristic and the corresponding DEST file is made sparse as well. That is the behavior selected by **--sparse=auto**. Specify **--sparse=always** to create a sparse DEST file whenever the SOURCE file contains a long enough sequence of zero bytes. Use **--sparse=never** to inhibit creation of sparse files.

The backup suffix is '~', unless set with **--suffix** or SIMPLE_BACKUP_SUFFIX. The version control method may be selected via the **--backup** option or through the VERSION_CONTROL environment variable. Here are the values:

- none, off
never make backups (even if **--backup** is given)
- numbered, t
make numbered backups
- existing, nil
numbered if numbered backups exist, simple otherwise
- simple, never
always make simple backups

As a special case, cp makes a backup of SOURCE when the force and backup options are given and SOURCE and DEST are the same name for an existing, regular file.

AUTHOR

Written by Torbjorn Granlund, David MacKenzie, and Jim Meyering.

REPORTING BUGS

Report bugs to <bug-coreutils@gnu.org>.

COPYRIGHT

Copyright © 2004 Free Software Foundation, Inc.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

SEE ALSO

The full documentation for **cp** is maintained as a Texinfo manual. If the **info** and **cp** programs are properly installed at your site, the command

info cp

should give you access to the complete manual.

NAME

`csplit` – split a file into sections determined by context lines

SYNOPSIS

csplit [*OPTION*]... *FILE PATTERN*...

DESCRIPTION

Output pieces of *FILE* separated by *PATTERN*(s) to files ‘xx01’, ‘xx02’, ..., and output byte counts of each piece to standard output.

Mandatory arguments to long options are mandatory for short options too.

-b, --suffix-format=FORMAT use sprintf *FORMAT* instead of %02d

-f, --prefix=PREFIX
use *PREFIX* instead of ‘xx’

-k, --keep-files
do not remove output files on errors

-n, --digits=DIGITS
use specified number of digits instead of 2

-s, --quiet, --silent
do not print counts of output file sizes

-z, --elide-empty-files
remove empty output files

--help display this help and exit

--version
output version information and exit

Read standard input if *FILE* is -. Each *PATTERN* may be:

INTEGER
copy up to but not including specified line number

/REGEXP/[OFFSET]
copy up to but not including a matching line

%REGEXP%[OFFSET]
skip to, but not including a matching line

{INTEGER}
repeat the previous pattern specified number of times

{}* repeat the previous pattern as many times as possible

A line *OFFSET* is a required ‘+’ or ‘-’ followed by a positive integer.

AUTHOR

Written by Stuart Kemp and David MacKenzie.

REPORTING BUGS

Report bugs to <bug-coreutils@gnu.org>.

COPYRIGHT

Copyright © 2004 Free Software Foundation, Inc.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

SEE ALSO

The full documentation for **csplit** is maintained as a Texinfo manual. If the **info** and **csplit** programs are properly installed at your site, the command

info csplit

should give you access to the complete manual.

NAME

cut – remove sections from each line of files

SYNOPSIS

cut [*OPTION*]... [*FILE*]...

DESCRIPTION

Print selected parts of lines from each *FILE* to standard output.

Mandatory arguments to long options are mandatory for short options too.

-b, --bytes=LIST

select only these bytes

-c, --characters=LIST

select only these characters

-d, --delimiter=DELIM

use DELIM instead of TAB for field delimiter

-f, --fields=LIST

select only these fields; also print any line that contains no delimiter character, unless the **-s** option is specified

-n (ignored)

--complement

complement the set of selected bytes, characters or fields.

-s, --only-delimited

do not print lines not containing delimiters

--output-delimiter=STRING

use STRING as the output delimiter the default is to use the input delimiter

--help display this help and exit

--version

output version information and exit

Use one, and only one of **-b**, **-c** or **-f**. Each LIST is made up of one range, or many ranges separated by commas. Selected input is written in the same order that it is read, and is written exactly once. Each range is one of:

N N'th byte, character or field, counted from 1

N- from N'th byte, character or field, to end of line

N-M from N'th to M'th (included) byte, character or field

-M from first to M'th (included) byte, character or field

With no *FILE*, or when *FILE* is **-**, read standard input.

AUTHOR

Written by David Ihnat, David MacKenzie, and Jim Meyering.

REPORTING BUGS

Report bugs to <bug-coreutils@gnu.org>.

COPYRIGHT

Copyright © 2004 Free Software Foundation, Inc.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

SEE ALSO

The full documentation for **cut** is maintained as a Texinfo manual. If the **info** and **cut** programs are properly installed at your site, the command

info cut

should give you access to the complete manual.

NAME

date – print or set the system date and time

SYNOPSIS

date [*OPTION*]... [+*FORMAT*]

date [-u/--utc/--universal] [*MMDDhhmm*[[*CC*]*YY*][*.ss*]]

DESCRIPTION

Display the current time in the given *FORMAT*, or set the system date.

-d, --date=STRING

display time described by *STRING*, not ‘now’

-f, --file=DATEFILE

like **--date** once for each line of *DATEFILE*

--iso-8601[=TIMESPEC] output date/time in ISO 8601 format.

TIMESPEC=‘date’ for date only (the default), ‘hours’, ‘minutes’, ‘seconds’, or ‘ns’ for date and time to the indicated precision.

-r, --reference=FILE

display the last modification time of *FILE*

-R, --rfc-2822

output RFC-2822 compliant date string

-s, --set=STRING

set time described by *STRING*

-u, --utc, --universal

print or set Coordinated Universal Time

--help display this help and exit

--version

output version information and exit

FORMAT controls the output. The only valid option for the second form specifies Coordinated Universal Time. Interpreted sequences are:

%%	a literal %
%a	locale’s abbreviated weekday name (Sun..Sat)
%A	locale’s full weekday name, variable length (Sunday..Saturday)
%b	locale’s abbreviated month name (Jan..Dec)
%B	locale’s full month name, variable length (January..December)
%c	locale’s date and time (Sat Nov 04 12:02:33 EST 1989)
%C	century (year divided by 100 and truncated to an integer) [00–99]
%d	day of month (01..31)
%D	date (mm/dd/yy)
%e	day of month, blank padded (1..31)
%F	same as %Y-%m-%d
%g	the 2–digit year corresponding to the %V week number
%G	the 4–digit year corresponding to the %V week number
%h	same as %b
%H	hour (00..23)
%I	hour (01..12)
%j	day of year (001..366)

%k	hour (0..23)
%l	hour (1..12)
%m	month (01..12)
%M	minute (00..59)
%n	a newline
%N	nanoseconds (000000000..999999999)
%p	locale's upper case AM or PM indicator (blank in many locales)
%P	locale's lower case am or pm indicator (blank in many locales)
%r	time, 12-hour (hh:mm:ss [AP]M)
%R	time, 24-hour (hh:mm)
%s	seconds since '00:00:00 1970-01-01 UTC' (a GNU extension)
%S	second (00..60); the 60 is necessary to accommodate a leap second
%t	a horizontal tab
%T	time, 24-hour (hh:mm:ss)
%u	day of week (1..7); 1 represents Monday
%U	week number of year with Sunday as first day of week (00..53)
%V	week number of year with Monday as first day of week (01..53)
%w	day of week (0..6); 0 represents Sunday
%W	week number of year with Monday as first day of week (00..53)
%x	locale's date representation (mm/dd/yy)
%X	locale's time representation (%H:%M:%S)
%y	last two digits of year (00..99)
%Y	year (1970...)
%z	RFC-2822 style numeric timezone (-0500) (a nonstandard extension)
%Z	time zone (e.g., EDT), or nothing if no time zone is determinable

By default, date pads numeric fields with zeroes. GNU date recognizes the following modifiers between '%' and a numeric directive.

'-' (hyphen) do not pad the field '_' (underscore) pad the field with spaces

AUTHOR

Written by David MacKenzie.

REPORTING BUGS

Report bugs to <bug-coreutils@gnu.org>.

COPYRIGHT

Copyright © 2004 Free Software Foundation, Inc.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

SEE ALSO

The full documentation for **date** is maintained as a Texinfo manual. If the **info** and **date** programs are properly installed at your site, the command

info date

should give you access to the complete manual.

NAME

dd – convert and copy a file

SYNOPSIS

dd [*OPERAND*]...

dd *OPTION*

DESCRIPTION

Copy a file, converting and formatting according to the operands.

bs=BYTES

force ibs=BYTES and obs=BYTES

cbs=BYTES

convert BYTES bytes at a time

conv=CONVS

convert the file as per the comma separated symbol list

count=BLOCKS

copy only BLOCKS input blocks

ibs=BYTES

read BYTES bytes at a time

if=FILE

read from FILE instead of stdin

iflag=FLAGS

read as per the comma separated symbol list

obs=BYTES

write BYTES bytes at a time

of=FILE

write to FILE instead of stdout

oflag=FLAGS

write as per the comma separated symbol list

seek=BLOCKS

skip BLOCKS obs-sized blocks at start of output

skip=BLOCKS

skip BLOCKS ibs-sized blocks at start of input

status=noxfer

suppress transfer statistics

BLOCKS and BYTES may be followed by the following multiplicative suffixes: xM M, c 1, w 2, b 512, kB 1000, K 1024, MB 1000*1000, M 1024*1024, GB 1000*1000*1000, G 1024*1024*1024, and so on for T, P, E, Z, Y.

Each CONV symbol may be:

ascii from EBCDIC to ASCII

ebedic from ASCII to EBCDIC

ibm from ASCII to alternate EBCDIC

block pad newline-terminated records with spaces to cbs-size

unblock

replace trailing spaces in cbs-size records with newline

lcase change upper case to lower case

nocreat do not create the output file

excl fail if the output file already exists

notrunc do not truncate the output file
 ucase change lower case to upper case
 swab swap every pair of input bytes
 noerror continue after read errors
 sync pad every input block with NULs to `ibs=size`; when used
 with block or unblock, pad with spaces rather than NULs
 fdatsync physically write output file data before finishing fsync likewise, but also write
 metadata

Each FLAG symbol may be:

append append mode (makes sense only for output)
 direct use direct I/O for data
 dsync use synchronized I/O for data
 sync likewise, but also for metadata
 nonblock
 use non-blocking I/O
 nofollow
 do not follow symlinks
 noctty do not assign controlling terminal from file

Sending a SIGUSR1 signal to a running ‘dd’ process makes it print I/O statistics to standard error, then to resume copying.

```

$ dd if=/dev/zero of=/dev/null& pid=$!
$ kill -USR1 $pid; sleep 1; kill $pid

18335302+0 records in 18335302+0 records out 9387674624 bytes (9.4 GB) copied, 34.6279
seconds, 271 MB/s
  
```

Options are:

—help display this help and exit
—version
 output version information and exit

AUTHOR

Written by Paul Rubin, David MacKenzie, and Stuart Kemp.

REPORTING BUGS

Report bugs to <bug-coreutils@gnu.org>.

COPYRIGHT

Copyright © 2004 Free Software Foundation, Inc.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

SEE ALSO

The full documentation for **dd** is maintained as a Texinfo manual. If the **info** and **dd** programs are properly installed at your site, the command

info dd

should give you access to the complete manual.

NAME

df – report file system disk space usage

SYNOPSIS

df [*OPTION*]... [*FILE*]...

DESCRIPTION

This manual page documents the GNU version of **df**. **df** displays the amount of disk space available on the file system containing each file name argument. If no file name is given, the space available on all currently mounted file systems is shown. Disk space is shown in 1K blocks by default, unless the environment variable POSIXLY_CORRECT is set, in which case 512-byte blocks are used.

If an argument is the absolute file name of a disk device node containing a mounted file system, **df** shows the space available on that file system rather than on the file system containing the device node (which is always the root file system). This version of **df** cannot show the space available on unmounted file systems, because on most kinds of systems doing so requires very nonportable intimate knowledge of file system structures.

OPTIONS

Show information about the file system on which each *FILE* resides, or all file systems by default.

Mandatory arguments to long options are mandatory for short options too.

-a, --all

include file systems having 0 blocks

-B, --block-size=SIZE use SIZE-byte blocks

-h, --human-readable

print sizes in human readable format (e.g., 1K 234M 2G)

-H, --si

likewise, but use powers of 1000 not 1024

-i, --inodes

list inode information instead of block usage

-k like **--block-size=1K**

-l, --local

limit listing to local file systems

--no-sync

do not invoke sync before getting usage info (default)

-P, --portability

use the POSIX output format

--sync invoke sync before getting usage info

-t, --type=TYPE

limit listing to file systems of type TYPE

-T, --print-type

print file system type

-x, --exclude-type=TYPE

limit listing to file systems not of type TYPE

-v (ignored)

--help display this help and exit

--version

output version information and exit

SIZE may be (or may be an integer optionally followed by) one of following: kB 1000, K 1024, MB 1000*1000, M 1024*1024, and so on for G, T, P, E, Z, Y.

AUTHOR

Written by Torbjorn Granlund, David MacKenzie, and Paul Eggert.

REPORTING BUGS

Report bugs to <bug-coreutils@gnu.org>.

COPYRIGHT

Copyright © 2004 Free Software Foundation, Inc.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

SEE ALSO

The full documentation for **df** is maintained as a Texinfo manual. If the **info** and **df** programs are properly installed at your site, the command

info df

should give you access to the complete manual.

NAME

`dir` – list directory contents

SYNOPSIS

dir [*OPTION*]... [*FILE*]...

DESCRIPTION

List information about the *FILES* (the current directory by default). Sort entries alphabetically if none of **-cftuSUX** nor **--sort**.

Mandatory arguments to long options are mandatory for short options too.

-a, --all

do not ignore entries starting with `.`

-A, --almost-all

do not list implied `.` and `..`

--author

with **-l**, print the author of each file

-b, --escape

print octal escapes for nongraphic characters

--block-size=SIZE

use *SIZE*-byte blocks

-B, --ignore-backups

do not list implied entries ending with `~`

-c

with **-lt**: sort by, and show, ctime (time of last modification of file status information) with **-l**: show ctime and sort by name otherwise: sort by ctime

-C

list entries by columns

--color[=WHEN]

control whether color is used to distinguish file types. *WHEN* may be ‘never’, ‘always’, or ‘auto’

-d, --directory

list directory entries instead of contents, and do not dereference symbolic links

-D, --dired

generate output designed for Emacs’ dired mode

-f

do not sort, enable **-aU**, disable **-lst**

-F, --classify

append indicator (one of `*/=@|`) to entries

--format=WORD

across **-x**, commas **-m**, horizontal **-x**, long **-l**, single-column **-l**, verbose **-l**, vertical **-C**

--full-time

like **-l** **--time-style=full-iso**

-g

like **-l**, but do not list owner

-G, --no-group

like **-l**, but do not list group

-h, --human-readable

with **-l**, print sizes in human readable format (e.g., 1K 234M 2G)

--si

likewise, but use powers of 1000 not 1024

-H, --dereference-command-line

follow symbolic links listed on the command line

--dereference-command-line-symlink-to-dir

follow each command line symbolic link that points to a directory

--hide=PATTERN
do not list implied entries matching shell PATTERN (overridden by **-a** or **-A**)

--indicator-style=WORD append indicator with style WORD to entry names:
none (default), classify (**-F**), file-type (**-p**)

-i, --inode
with **-l**, print the index number of each file

-I, --ignore=PATTERN
do not list implied entries matching shell PATTERN

-k like **--block-size=1K**

-l use a long listing format

-L, --dereference
when showing file information for a symbolic link, show information for the file the link references rather than for the link itself

-m fill width with a comma separated list of entries

-n, --numeric-uid-gid
like **-l**, but list numeric UIDs and GIDs

-N, --literal
print raw entry names (don't treat e.g. control characters specially)

-o like **-l**, but do not list group information

-p, --file-type
append indicator (one of **/=@|**) to entries

-q, --hide-control-chars
print **?** instead of non graphic characters

--show-control-chars
show non graphic characters as-is (default unless program is 'ls' and output is a terminal)

-Q, --quote-name
enclose entry names in double quotes

--quoting-style=WORD
use quoting style WORD for entry names: literal, locale, shell, shell-always, c, escape

-r, --reverse
reverse order while sorting

-R, --recursive
list subdirectories recursively

-s, --size
with **-l**, print size of each file, in blocks

-S sort by file size

--sort=WORD
extension **-X**, none **-U**, size **-S**, time **-t**, version **-v**, status **-c**, time **-t**, atime **-u**, access **-u**, use **-u**

--time=WORD
with **-l**, show time as WORD instead of modification time: atime, access, use, ctime or status;
use specified time as sort key if **--sort=time**

--time-style=STYLE
with **-l**, show times using style STYLE: full-iso, long-iso, iso, locale, +FORMAT. FORMAT is interpreted like 'date'; if FORMAT is FORMAT1<newline>FORMAT2, FORMAT1 applies to non-recent files and FORMAT2 to recent files; if STYLE is prefixed with 'posix-', STYLE takes effect only outside the POSIX locale

-t sort by modification time

- T, --tabsize=COLS**
assume tab stops at each COLS instead of 8
- u** with **-lt**: sort by, and show, access time with **-l**: show access time and sort by name otherwise: sort by access time
- U** do not sort; list entries in directory order
- v** sort by version
- w, --width=COLS**
assume screen width instead of current value
- x** list entries by lines instead of by columns
- X** sort alphabetically by entry extension
- 1** list one file per line
- help** display this help and exit
- version**
output version information and exit

SIZE may be (or may be an integer optionally followed by) one of following: kB 1000, K 1024, MB 1000*1000, M 1024*1024, and so on for G, T, P, E, Z, Y.

By default, color is not used to distinguish types of files. That is equivalent to using **--color=none**. Using the **--color** option without the optional WHEN argument is equivalent to using **--color=always**. With **--color=auto**, color codes are output only if standard output is connected to a terminal (tty).

Exit status is 0 if OK, 1 if minor problems, 2 if serious trouble.

AUTHOR

Written by Richard Stallman and David MacKenzie.

REPORTING BUGS

Report bugs to <bug-coreutils@gnu.org>.

COPYRIGHT

Copyright © 2004 Free Software Foundation, Inc.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

SEE ALSO

The full documentation for **dir** is maintained as a Texinfo manual. If the **info** and **dir** programs are properly installed at your site, the command

info dir

should give you access to the complete manual.

NAME

dircolors – color setup for ls

SYNOPSIS

dircolors [*OPTION*]... [*FILE*]

DESCRIPTION

Output commands to set the LS_COLORS environment variable.

Determine format of output:

-b, --sh, --bourne-shell

output Bourne shell code to set LS_COLORS

-c, --csh, --c-shell

output C shell code to set LS_COLORS

-p, --print-database

output defaults

--help display this help and exit

--version

output version information and exit

If *FILE* is specified, read it to determine which colors to use for which file types and extensions. Otherwise, a precompiled database is used. For details on the format of these files, run '**dircolors --print-database**'.

AUTHOR

Written by H. Peter Anvin.

REPORTING BUGS

Report bugs to <bug-coreutils@gnu.org>.

COPYRIGHT

Copyright © 2004 Free Software Foundation, Inc.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

SEE ALSO

The full documentation for **dircolors** is maintained as a Texinfo manual. If the **info** and **dircolors** programs are properly installed at your site, the command

info dircolors

should give you access to the complete manual.

NAME

`dirname` – strip non-directory suffix from file name

SYNOPSIS

dirname *NAME*

dirname *OPTION*

DESCRIPTION

Print *NAME* with its trailing /component removed; if *NAME* contains no /'s, output '.' (meaning the current directory).

—help display this help and exit

—version

output version information and exit

AUTHOR

Written by David MacKenzie and Jim Meyering.

REPORTING BUGS

Report bugs to <bug-coreutils@gnu.org>.

COPYRIGHT

Copyright © 2004 Free Software Foundation, Inc.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

SEE ALSO

The full documentation for **dirname** is maintained as a Texinfo manual. If the **info** and **dirname** programs are properly installed at your site, the command

info dirname

should give you access to the complete manual.

NAME

du – estimate file space usage

SYNOPSIS

du [*OPTION*]... [*FILE*]...

du [*OPTION*]... --files0-from=*F*

DESCRIPTION

Summarize disk usage of each *FILE*, recursively for directories.

Mandatory arguments to long options are mandatory for short options too.

-a, --all

write counts for all files, not just directories

--apparent-size

print apparent sizes, rather than disk usage; although the apparent size is usually smaller, it may be larger due to holes in ('sparse') files, internal fragmentation, indirect blocks, and the like

-B, --block-size=SIZE use *SIZE*-byte blocks

-b, --bytes

equivalent to '**--apparent-size --block-size=1**'

-c, --total

produce a grand total

-D, --dereference-args

dereference *FILE*s that are symbolic links

--files0-from=F

summarize disk usage of the NUL-terminated file names specified in file *F*

-H like **--si**, but also evokes a warning; will soon change to be equivalent to **--dereference-args (-D)**

-h, --human-readable

print sizes in human readable format (e.g., 1K 234M 2G)

--si like **-h**, but use powers of 1000 not 1024

-k like **--block-size=1K**

-l, --count-links

count sizes many times if hard linked

-L, --dereference

dereference all symbolic links

-P, --no-dereference

don't follow any symbolic links (this is the default)

-0, --null

end each output line with 0 byte rather than newline

-S, --separate-dirs

do not include size of subdirectories

-s, --summarize

display only a total for each argument

-x, --one-file-system

skip directories on different file systems

-X FILE, --exclude-from=FILE

Exclude files that match any pattern in *FILE*.

--exclude=PATTERN Exclude files that match *PATTERN*.

--max-depth=N

print the total for a directory (or file, with **--all**) only if it is N or fewer levels below the command line argument; **--max-depth=0** is the same as **--summarize**

--help display this help and exit

--version

output version information and exit

SIZE may be (or may be an integer optionally followed by) one of following: kB 1000, K 1024, MB 1000*1000, M 1024*1024, and so on for G, T, P, E, Z, Y.

PATTERNS

PATTERN is a shell pattern (not a regular expression). The pattern **?** matches any one character, whereas ***** matches any string (composed of zero, one or multiple characters). For example, ***.o** will match any files whose names end in **.o**. Therefore, the command

du --exclude='*.o'

will skip all files and subdirectories ending in **.o** (including the file **.o** itself).

AUTHOR

Written by Torbjorn Granlund, David MacKenzie, Paul Eggert, and Jim Meyering.

REPORTING BUGS

Report bugs to <bug-coreutils@gnu.org>.

COPYRIGHT

Copyright © 2005 Free Software Foundation, Inc.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

SEE ALSO

The full documentation for **du** is maintained as a Texinfo manual. If the **info** and **du** programs are properly installed at your site, the command

info du

should give you access to the complete manual.

NAME

echo – display a line of text

SYNOPSIS

echo [*OPTION*]... [*STRING*]...

DESCRIPTION

NOTE: your shell may have its own version of echo which will supersede the version described here. Please refer to your shell's documentation for details about the options it supports.

Echo the *STRING*(s) to standard output.

- n** do not output the trailing newline
- e** enable interpretation of backslash escapes
- E** disable interpretation of backslash escapes (default)
- help** display this help and exit
- version** output version information and exit

If **-e** is in effect, the following sequences are recognized:

- \0NNN** the character whose ASCII code is NNN (octal)
- ** backslash
- \a** alert (BEL)
- \b** backspace
- \c** suppress trailing newline
- \f** form feed
- \n** new line
- \r** carriage return
- \t** horizontal tab
- \v** vertical tab

AUTHOR

Written by FIXME unknown.

REPORTING BUGS

Report bugs to <bug-coreutils@gnu.org>.

COPYRIGHT

Copyright © 2004 Free Software Foundation, Inc.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

SEE ALSO

The full documentation for **echo** is maintained as a Texinfo manual. If the **info** and **echo** programs are properly installed at your site, the command

info echo

should give you access to the complete manual.

NAME

env – run a program in a modified environment

SYNOPSIS

env [*OPTION*]... [-] [*NAME=VALUE*]... [*COMMAND* [*ARG*]...]

DESCRIPTION

Set each *NAME* to *VALUE* in the environment and run *COMMAND*.

-i, --ignore-environment

start with an empty environment

-u, --unset=NAME

remove variable from the environment

--help display this help and exit

--version

output version information and exit

A mere **-** implies **-i**. If no *COMMAND*, print the resulting environment.

AUTHOR

Written by Richard Mlynarik and David MacKenzie.

REPORTING BUGS

Report bugs to <bug-coreutils@gnu.org>.

COPYRIGHT

Copyright © 2004 Free Software Foundation, Inc.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

SEE ALSO

The full documentation for **env** is maintained as a Texinfo manual. If the **info** and **env** programs are properly installed at your site, the command

info env

should give you access to the complete manual.

NAME

expand – convert tabs to spaces

SYNOPSIS

expand [*OPTION*]... [*FILE*]...

DESCRIPTION

Convert tabs in each *FILE* to spaces, writing to standard output. With no *FILE*, or when *FILE* is –, read standard input.

Mandatory arguments to long options are mandatory for short options too.

-i, --initial

do not convert tabs after non blanks

-t, --tabs=NUMBER

have tabs *NUMBER* characters apart, not 8

-t, --tabs=LIST

use comma separated list of explicit tab positions

--help display this help and exit

--version

output version information and exit

AUTHOR

Written by David MacKenzie.

REPORTING BUGS

Report bugs to <bug-coreutils@gnu.org>.

COPYRIGHT

Copyright © 2004 Free Software Foundation, Inc.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

SEE ALSO

unexpand(1)

The full documentation for **expand** is maintained as a Texinfo manual. If the **info** and **expand** programs are properly installed at your site, the command

info expand

should give you access to the complete manual.

NAME

expr – evaluate expressions

SYNOPSIS

expr *EXPRESSION*

expr *OPTION*

DESCRIPTION

--help display this help and exit

--version

output version information and exit

Print the value of *EXPRESSION* to standard output. A blank line below separates increasing precedence groups. *EXPRESSION* may be:

ARG1 | *ARG2*

ARG1 if it is neither null nor 0, otherwise *ARG2*

ARG1 & *ARG2*

ARG1 if neither argument is null or 0, otherwise 0

ARG1 < *ARG2*

ARG1 is less than *ARG2*

ARG1 <= *ARG2*

ARG1 is less than or equal to *ARG2*

ARG1 = *ARG2*

ARG1 is equal to *ARG2*

ARG1 != *ARG2*

ARG1 is unequal to *ARG2*

ARG1 >= *ARG2*

ARG1 is greater than or equal to *ARG2*

ARG1 > *ARG2*

ARG1 is greater than *ARG2*

ARG1 + *ARG2*

arithmetic sum of *ARG1* and *ARG2*

ARG1 – *ARG2*

arithmetic difference of *ARG1* and *ARG2*

ARG1 * *ARG2*

arithmetic product of *ARG1* and *ARG2*

ARG1 / *ARG2*

arithmetic quotient of *ARG1* divided by *ARG2*

ARG1 % *ARG2*

arithmetic remainder of *ARG1* divided by *ARG2*

STRING : REGEXP

anchored pattern match of REGEXP in STRING

match **STRING** REGEXP

same as **STRING** : REGEXP

substr **STRING** POS LENGTH

substring of STRING, POS counted from 1

index **STRING** CHARS

index in STRING where any CHARS is found, or 0

length **STRING**

length of STRING

+ TOKEN

interpret TOKEN as a string, even if it is a

keyword like 'match' or an operator like '/'

(EXPRESSION)

value of EXPRESSION

Beware that many operators need to be escaped or quoted for shells. Comparisons are arithmetic if both ARGs are numbers, else lexicographical. Pattern matches return the string matched between \(\ and \) or null; if \(\ and \) are not used, they return the number of characters matched or 0.

Exit status is 0 if EXPRESSION is neither null nor 0, 1 if EXPRESSION is null or 0, 2 if EXPRESSION is syntactically invalid, and 3 if an error occurred.

AUTHOR

Written by Mike Parker.

REPORTING BUGS

Report bugs to <bug-coreutils@gnu.org>.

COPYRIGHT

Copyright © 2004 Free Software Foundation, Inc.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

SEE ALSO

The full documentation for **expr** is maintained as a Texinfo manual. If the **info** and **expr** programs are properly installed at your site, the command

info expr

should give you access to the complete manual.

NAME

factor – factor numbers

SYNOPSIS

factor [*NUMBER*]...

factor *OPTION*

DESCRIPTION

Print the prime factors of each *NUMBER*.

—help display this help and exit

—version

output version information and exit

Print the prime factors of all specified integer *NUMBERS*.

If no arguments

are specified on the command line, they are read from standard input.

AUTHOR

Written by Paul Rubin.

REPORTING BUGS

Report bugs to <bug-coreutils@gnu.org>.

COPYRIGHT

Copyright © 2004 Free Software Foundation, Inc.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

SEE ALSO

The full documentation for **factor** is maintained as a Texinfo manual. If the **info** and **factor** programs are properly installed at your site, the command

info factor

should give you access to the complete manual.

NAME

false – do nothing, unsuccessfully

SYNOPSIS

false [*ignored command line arguments*]

false *OPTION*

DESCRIPTION

Exit with a status code indicating failure.

These option names may not be abbreviated.

--help display this help and exit

--version

output version information and exit

AUTHOR

Written by Jim Meyering.

REPORTING BUGS

Report bugs to <bug-coreutils@gnu.org>.

COPYRIGHT

Copyright © 2004 Free Software Foundation, Inc.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

SEE ALSO

The full documentation for **false** is maintained as a Texinfo manual. If the **info** and **false** programs are properly installed at your site, the command

info false

should give you access to the complete manual.

NAME

fmt – simple optimal text formatter

SYNOPSIS

fmt [-*DIGITS*] [*OPTION*]... [*FILE*]...

DESCRIPTION

Reformat each paragraph in the *FILE*(s), writing to standard output. If no *FILE* or if *FILE* is ‘–’, read standard input.

Mandatory arguments to long options are mandatory for short options too.

–c, --crown-margin

preserve indentation of first two lines

–p, --prefix=*STRING*

reformat only lines beginning with *STRING*, reattaching the prefix to reformatted lines

–s, --split-only

split long lines, but do not refill

–t, --tagged-paragraph

indentation of first line different from second

–u, --uniform-spacing

one space between words, two after sentences

–w, --width=*WIDTH*

maximum line width (default of 75 columns)

--help display this help and exit

--version

output version information and exit

With no *FILE*, or when *FILE* is –, read standard input.

AUTHOR

Written by Ross Paterson.

REPORTING BUGS

Report bugs to <bug-coreutils@gnu.org>.

COPYRIGHT

Copyright © 2004 Free Software Foundation, Inc.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

SEE ALSO

The full documentation for **fmt** is maintained as a Texinfo manual. If the **info** and **fmt** programs are properly installed at your site, the command

info fmt

should give you access to the complete manual.

NAME

fold – wrap each input line to fit in specified width

SYNOPSIS

fold [*OPTION*]... [*FILE*]...

DESCRIPTION

Wrap input lines in each *FILE* (standard input by default), writing to standard output.

Mandatory arguments to long options are mandatory for short options too.

-b, --bytes

count bytes rather than columns

-s, --spaces

break at spaces

-w, --width=WIDTH

use *WIDTH* columns instead of 80

--help display this help and exit

--version

output version information and exit

AUTHOR

Written by David MacKenzie.

REPORTING BUGS

Report bugs to <bug-coreutils@gnu.org>.

COPYRIGHT

Copyright © 2004 Free Software Foundation, Inc.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

SEE ALSO

The full documentation for **fold** is maintained as a Texinfo manual. If the **info** and **fold** programs are properly installed at your site, the command

info fold

should give you access to the complete manual.

NAME

groups – print the groups a user is in

SYNOPSIS

groups [*OPTION*]... [*USERNAME*]...

DESCRIPTION

—help display this help and exit

—version

output version information and exit

Same as **id -Gn**. If no **USERNAME**, use current process.

AUTHOR

Written by David MacKenzie.

REPORTING BUGS

Report bugs to <bug-coreutils@gnu.org>.

COPYRIGHT

Copyright © 2004 Free Software Foundation, Inc.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

SEE ALSO

The full documentation for **groups** is maintained as a Texinfo manual. If the **info** and **groups** programs are properly installed at your site, the command

info groups

should give you access to the complete manual.

NAME

head – output the first part of files

SYNOPSIS

head [*OPTION*]... [*FILE*]...

DESCRIPTION

Print the first 10 lines of each *FILE* to standard output. With more than one *FILE*, precede each with a header giving the file name. With no *FILE*, or when *FILE* is –, read standard input.

Mandatory arguments to long options are mandatory for short options too.

–c, --bytes=[–]N

print the first N bytes of each file; with the leading ‘–’, print all but the last N bytes of each file

–n, --lines=[–]N

print the first N lines instead of the first 10; with the leading ‘–’, print all but the last N lines of each file

–q, --quiet, --silent

never print headers giving file names

–v, --verbose

always print headers giving file names

--help display this help and exit

--version

output version information and exit

N may have a multiplier suffix: b 512, k 1024, m 1024*1024.

AUTHOR

Written by David MacKenzie and Jim Meyering.

REPORTING BUGS

Report bugs to <bug-coreutils@gnu.org>.

COPYRIGHT

Copyright © 2004 Free Software Foundation, Inc.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

SEE ALSO

The full documentation for **head** is maintained as a Texinfo manual. If the **info** and **head** programs are properly installed at your site, the command

info head

should give you access to the complete manual.

NAME

hostid – print the numeric identifier for the current host

SYNOPSIS

hostid

hostid *OPTION*

DESCRIPTION

Print the numeric identifier (in hexadecimal) for the current host.

--help display this help and exit

--version
output version information and exit

AUTHOR

Written by Jim Meyering.

REPORTING BUGS

Report bugs to <bug-coreutils@gnu.org>.

COPYRIGHT

Copyright © 2004 Free Software Foundation, Inc.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

SEE ALSO

The full documentation for **hostid** is maintained as a Texinfo manual. If the **info** and **hostid** programs are properly installed at your site, the command

info hostid

should give you access to the complete manual.

NAME

hostname – set or print the name of the current host system

SYNOPSIS

hostname [*NAME*]

hostname *OPTION*

DESCRIPTION

Print or set the hostname of the current system.

--help display this help and exit

--version

output version information and exit

AUTHOR

Written by Jim Meyering.

REPORTING BUGS

Report bugs to <bug-coreutils@gnu.org>.

COPYRIGHT

Copyright © 2004 Free Software Foundation, Inc.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

SEE ALSO

The full documentation for **hostname** is maintained as a Texinfo manual. If the **info** and **hostname** programs are properly installed at your site, the command

info hostname

should give you access to the complete manual.

NAME

id – print real and effective UIDs and GIDs

SYNOPSIS

id [*OPTION*]... [*USERNAME*]

DESCRIPTION

Print information for *USERNAME*, or the current user.

-a ignore, for compatibility with other versions

-g, --group
print only the effective group ID

-G, --groups
print all group IDs

-n, --name
print a name instead of a number, for **-ugG**

-r, --real
print the real ID instead of the effective ID, with **-ugG**

-u, --user
print only the effective user ID

--help display this help and exit

--version
output version information and exit

Without any *OPTION*, print some useful set of identified information.

AUTHOR

Written by Arnold Robbins and David MacKenzie.

REPORTING BUGS

Report bugs to <bug-coreutils@gnu.org>.

COPYRIGHT

Copyright © 2004 Free Software Foundation, Inc.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

SEE ALSO

The full documentation for **id** is maintained as a Texinfo manual. If the **info** and **id** programs are properly installed at your site, the command

info id

should give you access to the complete manual.

NAME

install – copy files and set attributes

SYNOPSIS

install [*OPTION*]... [-*T*] *SOURCE DEST*

install [*OPTION*]... *SOURCE... DIRECTORY*

install [*OPTION*]... -*t* *DIRECTORY SOURCE...*

install [*OPTION*]... -*d* *DIRECTORY...*

DESCRIPTION

In the first three forms, copy *SOURCE* to *DEST* or multiple *SOURCE*(s) to the existing *DIRECTORY*, while setting permission modes and owner/group. In the 4th form, create all components of the given *DIRECTORY*(ies).

Mandatory arguments to long options are mandatory for short options too.

--backup[=*CONTROL*] make a backup of each existing destination file

-b like **--backup** but does not accept an argument

-c (ignored)

-d, --directory

treat all arguments as directory names; create all components of the specified directories

-D create all leading components of *DEST* except the last, then copy *SOURCE* to *DEST*

-g, --group=*GROUP*

set group ownership, instead of process' current group

-m, --mode=*MODE*

set permission mode (as in *chmod*), instead of *rw-r--r--*

-o, --owner=*OWNER*

set ownership (super-user only)

-p, --preserve-timestamps

apply access/modification times of *SOURCE* files to corresponding destination files

-s, --strip

strip symbol tables

-S, --suffix=*SUFFIX* override the usual backup suffix

-t, --target-directory=*DIRECTORY*

copy all *SOURCE* arguments into *DIRECTORY*

-T, --no-target-directory

treat *DEST* as a normal file

-v, --verbose

print the name of each directory as it is created

--help display this help and exit

--version

output version information and exit

The backup suffix is ‘~’, unless set with **--suffix** or *SIMPLE_BACKUP_SUFFIX*. The version control method may be selected via the **--backup** option or through the *VERSION_CONTROL* environment variable. Here are the values:

none, off

never make backups (even if **--backup** is given)

numbered, t

make numbered backups

existing, nil

numbered if numbered backups exist, simple otherwise

simple, never
always make simple backups

AUTHOR

Written by David MacKenzie.

REPORTING BUGS

Report bugs to <bug-coreutils@gnu.org>.

COPYRIGHT

Copyright © 2004 Free Software Foundation, Inc.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

SEE ALSO

The full documentation for **install** is maintained as a Texinfo manual. If the **info** and **install** programs are properly installed at your site, the command

info install

should give you access to the complete manual.

NAME

join – join lines of two files on a common field

SYNOPSIS

join [*OPTION*]... *FILE1 FILE2*

DESCRIPTION

For each pair of input lines with identical join fields, write a line to standard output. The default join field is the first, delimited by whitespace. When *FILE1* or *FILE2* (not both) is –, read standard input.

–a FILENUM

print unpairable lines coming from file *FILENUM*, where *FILENUM* is 1 or 2, corresponding to *FILE1* or *FILE2*

–e EMPTY

replace missing input fields with *EMPTY*

–i, --ignore-case ignore differences in case when comparing fields

–j FIELD

equivalent to ‘–1 *FIELD* –2 *FIELD*’

–o FORMAT

obey *FORMAT* while constructing output line

–t CHAR

use *CHAR* as input and output field separator

–v FILENUM

like **–a FILENUM**, but suppress joined output lines

–1 FIELD

join on this *FIELD* of file 1

–2 FIELD

join on this *FIELD* of file 2

--help display this help and exit

--version

output version information and exit

Unless **–t CHAR** is given, leading blanks separate fields and are ignored, else fields are separated by *CHAR*. Any *FIELD* is a field number counted from 1. *FORMAT* is one or more comma or blank separated specifications, each being ‘*FILENUM.FIELD*’ or ‘0’. Default *FORMAT* outputs the join field, the remaining fields from *FILE1*, the remaining fields from *FILE2*, all separated by *CHAR*.

Important: *FILE1* and *FILE2* must be sorted on the join fields.

AUTHOR

Written by Mike Haertel.

REPORTING BUGS

Report bugs to <bug-coreutils@gnu.org>.

COPYRIGHT

Copyright © 2004 Free Software Foundation, Inc.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

SEE ALSO

The full documentation for **join** is maintained as a Texinfo manual. If the **info** and **join** programs are properly installed at your site, the command

info join

should give you access to the complete manual.

NAME

kill – send signals to processes, or list signals

SYNOPSIS

kill [-s *SIGNAL* / -*SIGNAL*] *PID*...

kill -l [*SIGNAL*]...

kill -t [*SIGNAL*]...

DESCRIPTION

Send signals to processes, or list signals.

Mandatory arguments to long options are mandatory for short options too.

-s, --signal=*SIGNAL*, **-***SIGNAL*

specify the name or number of the signal to be sent

-l, --list

list signal names, or convert signal names to/from numbers

-t, --table

print a table of signal information

--help display this help and exit

--version

output version information and exit

SIGNAL may be a signal name like ‘HUP’, or a signal number like ‘1’, or an exit status of a process terminated by a signal. *PID* is an integer; if negative it identifies a process group.

AUTHOR

Written by Paul Eggert.

REPORTING BUGS

Report bugs to <bug-coreutils@gnu.org>.

COPYRIGHT

Copyright © 2004 Free Software Foundation, Inc.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

SEE ALSO

The full documentation for **kill** is maintained as a Texinfo manual. If the **info** and **kill** programs are properly installed at your site, the command

info kill

should give you access to the complete manual.

NAME

link – call the link function to create a link to a file

SYNOPSIS

link *FILE1 FILE2*

link *OPTION*

DESCRIPTION

Call the link function to create a link named FILE2 to an existing FILE1.

--help display this help and exit

--version

output version information and exit

AUTHOR

Written by Michael Stone.

REPORTING BUGS

Report bugs to <bug-coreutils@gnu.org>.

COPYRIGHT

Copyright © 2004 Free Software Foundation, Inc.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

SEE ALSO

The full documentation for **link** is maintained as a Texinfo manual. If the **info** and **link** programs are properly installed at your site, the command

info link

should give you access to the complete manual.

NAME

ln – make links between files

SYNOPSIS

ln [*OPTION*]... [-*T*] *TARGET LINK_NAME* (1st form)

ln [*OPTION*]... *TARGET* (2nd form)

ln [*OPTION*]... *TARGET... DIRECTORY* (3rd form)

ln [*OPTION*]... -*t* *DIRECTORY TARGET...* (4th form)

DESCRIPTION

In the 1st form, create a link to *TARGET* with the name *LINK_NAME*. In the 2nd form, create a link to *TARGET* in the current directory. In the 3rd and 4th forms, create links to each *TARGET* in *DIRECTORY*. Create hard links by default, symbolic links with **--symbolic**. When creating hard links, each *TARGET* must exist.

Mandatory arguments to long options are mandatory for short options too.

--backup[=*CONTROL*]

make a backup of each existing destination file

-b like **--backup** but does not accept an argument

-d, -F, --directory

allow the superuser to attempt to hard link directories (note: will probably fail due to system restrictions, even for the superuser)

-f, --force

remove existing destination files

-n, --no-dereference

treat destination that is a symlink to a directory as if it were a normal file

-i, --interactive

prompt whether to remove destinations

-s, --symbolic

make symbolic links instead of hard links

-S, --suffix=*SUFFIX*

override the usual backup suffix

-t, --target-directory=*DIRECTORY*

specify the *DIRECTORY* in which to create the links

-T, --no-target-directory

treat *LINK_NAME* as a normal file

-v, --verbose

print name of each file before linking

--help display this help and exit

--version

output version information and exit

The backup suffix is '~', unless set with **--suffix** or *SIMPLE_BACKUP_SUFFIX*. The version control method may be selected via the **--backup** option or through the *VERSION_CONTROL* environment variable. Here are the values:

none, off

never make backups (even if **--backup** is given)

numbered, t

make numbered backups

existing, nil

numbered if numbered backups exist, simple otherwise

simple, never
always make simple backups

AUTHOR

Written by Mike Parker and David MacKenzie.

REPORTING BUGS

Report bugs to <bug-coreutils@gnu.org>.

COPYRIGHT

Copyright © 2004 Free Software Foundation, Inc.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

SEE ALSO

The full documentation for **ln** is maintained as a Texinfo manual. If the **info** and **ln** programs are properly installed at your site, the command

info ln

should give you access to the complete manual.

NAME

logname – print user's login name

SYNOPSIS

logname [*OPTION*]

DESCRIPTION

Print the name of the current user.

—help display this help and exit

—version
output version information and exit

AUTHOR

Written by FIXME: unknown.

REPORTING BUGS

Report bugs to <bug-coreutils@gnu.org>.

COPYRIGHT

Copyright © 2004 Free Software Foundation, Inc.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

SEE ALSO

The full documentation for **logname** is maintained as a Texinfo manual. If the **info** and **logname** programs are properly installed at your site, the command

info logname

should give you access to the complete manual.

NAME

ls – list directory contents

SYNOPSIS

ls [*OPTION*]... [*FILE*]...

DESCRIPTION

List information about the FILES (the current directory by default). Sort entries alphabetically if none of **-cftuSUX** nor **--sort**.

Mandatory arguments to long options are mandatory for short options too.

-a, --all

do not ignore entries starting with .

-A, --almost-all

do not list implied . and ..

--author

with **-l**, print the author of each file

-b, --escape

print octal escapes for nongraphic characters

--block-size=SIZE

use SIZE-byte blocks

-B, --ignore-backups

do not list implied entries ending with ~

-c

with **-lt**: sort by, and show, ctime (time of last modification of file status information) with **-l**: show ctime and sort by name otherwise: sort by ctime

-C

list entries by columns

--color[=WHEN]

control whether color is used to distinguish file types. WHEN may be ‘never’, ‘always’, or ‘auto’

-d, --directory

list directory entries instead of contents, and do not dereference symbolic links

-D, --dired

generate output designed for Emacs’ dired mode

-f

do not sort, enable **-aU**, disable **-lst**

-F, --classify

append indicator (one of */=@|) to entries

--format=WORD

across **-x**, commas **-m**, horizontal **-x**, long **-l**, single-column **-l**, verbose **-l**, vertical **-C**

--full-time

like **-l** **--time-style=full-iso**

-g

like **-l**, but do not list owner

-G, --no-group

like **-l**, but do not list group

-h, --human-readable

with **-l**, print sizes in human readable format (e.g., 1K 234M 2G)

--si

likewise, but use powers of 1000 not 1024

-H, --dereference-command-line

follow symbolic links listed on the command line

--dereference-command-line-symlink-to-dir

follow each command line symbolic link that points to a directory

--hide=PATTERN
do not list implied entries matching shell PATTERN (overridden by **-a** or **-A**)

--indicator-style=WORD append indicator with style WORD to entry names:
none (default), classify (**-F**), file-type (**-p**)

-i, --inode
with **-l**, print the index number of each file

-I, --ignore=PATTERN
do not list implied entries matching shell PATTERN

-k like **--block-size=1K**

-l use a long listing format

-L, --dereference
when showing file information for a symbolic link, show information for the file the link references rather than for the link itself

-m fill width with a comma separated list of entries

-n, --numeric-uid-gid
like **-l**, but list numeric UIDs and GIDs

-N, --literal
print raw entry names (don't treat e.g. control characters specially)

-o like **-l**, but do not list group information

-p, --file-type
append indicator (one of **/= @|**) to entries

-q, --hide-control-chars
print **?** instead of non graphic characters

--show-control-chars
show non graphic characters as-is (default unless program is 'ls' and output is a terminal)

-Q, --quote-name
enclose entry names in double quotes

--quoting-style=WORD
use quoting style WORD for entry names: literal, locale, shell, shell-always, c, escape

-r, --reverse
reverse order while sorting

-R, --recursive
list subdirectories recursively

-s, --size
with **-l**, print size of each file, in blocks

-S sort by file size

--sort=WORD
extension **-X**, none **-U**, size **-S**, time **-t**, version **-v**, status **-c**, time **-t**, atime **-u**, access **-u**, use **-u**

--time=WORD
with **-l**, show time as WORD instead of modification time: atime, access, use, ctime or status;
use specified time as sort key if **--sort=time**

--time-style=STYLE
with **-l**, show times using style STYLE: full-iso, long-iso, iso, locale, +FORMAT. FORMAT is interpreted like 'date'; if FORMAT is FORMAT1<newline>FORMAT2, FORMAT1 applies to non-recent files and FORMAT2 to recent files; if STYLE is prefixed with 'posix-', STYLE takes effect only outside the POSIX locale

-t sort by modification time

- T, --tabsize=COLS**
assume tab stops at each COLS instead of 8
- u** with **-lt**: sort by, and show, access time with **-l**: show access time and sort by name otherwise: sort by access time
- U** do not sort; list entries in directory order
- v** sort by version
- w, --width=COLS**
assume screen width instead of current value
- x** list entries by lines instead of by columns
- X** sort alphabetically by entry extension
- 1** list one file per line
- help** display this help and exit
- version**
output version information and exit

SIZE may be (or may be an integer optionally followed by) one of following: kB 1000, K 1024, MB 1000*1000, M 1024*1024, and so on for G, T, P, E, Z, Y.

By default, color is not used to distinguish types of files. That is equivalent to using **--color=none**. Using the **--color** option without the optional WHEN argument is equivalent to using **--color=always**. With **--color=auto**, color codes are output only if standard output is connected to a terminal (tty).

Exit status is 0 if OK, 1 if minor problems, 2 if serious trouble.

AUTHOR

Written by Richard Stallman and David MacKenzie.

REPORTING BUGS

Report bugs to <bug-coreutils@gnu.org>.

COPYRIGHT

Copyright © 2004 Free Software Foundation, Inc.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

SEE ALSO

The full documentation for **ls** is maintained as a Texinfo manual. If the **info** and **ls** programs are properly installed at your site, the command

info ls

should give you access to the complete manual.

NAME

md5sum – compute and check MD5 message digest

SYNOPSIS

md5sum [*OPTION*] [*FILE*]...

md5sum [*OPTION*] *--check* [*FILE*]

DESCRIPTION

Print or check MD5 (128-bit) checksums. With no *FILE*, or when *FILE* is –, read standard input.

–b, --binary

read files in binary mode (default on DOS/Windows)

–c, --check

check MD5 sums against given list

–t, --text

read files in text mode (default)

The following two options are useful only when verifying checksums:

--status

don't output anything, status code shows success

–w, --warn

warn about improperly formatted checksum lines

--help display this help and exit

--version

output version information and exit

The sums are computed as described in RFC 1321. When checking, the input should be a former output of this program. The default mode is to print a line with checksum, a character indicating type (* for binary, ' ' for text), and name for each *FILE*.

AUTHOR

Written by Ulrich Drepper and Scott Miller.

REPORTING BUGS

Report bugs to <bug-coreutils@gnu.org>.

COPYRIGHT

Copyright © 2004 Free Software Foundation, Inc.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

SEE ALSO

The full documentation for **md5sum** is maintained as a Texinfo manual. If the **info** and **md5sum** programs are properly installed at your site, the command

info md5sum

should give you access to the complete manual.

NAME

mkdir – make directories

SYNOPSIS

mkdir [*OPTION*] *DIRECTORY*...

DESCRIPTION

Create the *DIRECTORY*(ies), if they do not already exist.

Mandatory arguments to long options are mandatory for short options too.

-m, --mode=MODE

set permission mode (as in chmod), not rwxrwxrwx – umask

-p, --parents

no error if existing, make parent directories as needed

-v, --verbose

print a message for each created directory

--help display this help and exit

--version

output version information and exit

AUTHOR

Written by David MacKenzie.

REPORTING BUGS

Report bugs to <bug-coreutils@gnu.org>.

COPYRIGHT

Copyright © 2004 Free Software Foundation, Inc.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

SEE ALSO

The full documentation for **mkdir** is maintained as a Texinfo manual. If the **info** and **mkdir** programs are properly installed at your site, the command

info mkdir

should give you access to the complete manual.

NAME

mkfifo – make FIFOs (named pipes)

SYNOPSIS

mkfifo [*OPTION*] *NAME*...

DESCRIPTION

Create named pipes (FIFOs) with the given NAMES.

Mandatory arguments to long options are mandatory for short options too.

-m, --mode=MODE

set permission mode (as in chmod), not a=rw – umask

--help display this help and exit

--version

output version information and exit

AUTHOR

Written by David MacKenzie.

REPORTING BUGS

Report bugs to <bug-coreutils@gnu.org>.

COPYRIGHT

Copyright © 2004 Free Software Foundation, Inc.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

SEE ALSO

The full documentation for **mkfifo** is maintained as a Texinfo manual. If the **info** and **mkfifo** programs are properly installed at your site, the command

info mkfifo

should give you access to the complete manual.

NAME

mknod – make block or character special files

SYNOPSIS

mknod [*OPTION*]... *NAME TYPE* [*MAJOR MINOR*]

DESCRIPTION

Create the special file *NAME* of the given *TYPE*.

Mandatory arguments to long options are mandatory for short options too.

-m, --mode=MODE

set permission mode (as in *chmod*), not *a=rw - umask*

--help display this help and exit

--version

output version information and exit

Both *MAJOR* and *MINOR* must be specified when *TYPE* is *b*, *c*, or *u*, and they must be omitted when *TYPE* is *p*. If *MAJOR* or *MINOR* begins with *0x* or *0X*, it is interpreted as hexadecimal; otherwise, if it begins with *0*, as octal; otherwise, as decimal. *TYPE* may be:

b create a block (buffered) special file

c, *u* create a character (unbuffered) special file

p create a FIFO

AUTHOR

Written by David MacKenzie.

REPORTING BUGS

Report bugs to <bug-coreutils@gnu.org>.

COPYRIGHT

Copyright © 2004 Free Software Foundation, Inc.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

SEE ALSO

The full documentation for **mknod** is maintained as a Texinfo manual. If the **info** and **mknod** programs are properly installed at your site, the command

info mknod

should give you access to the complete manual.

NAME

mv – move (rename) files

SYNOPSIS

mv [*OPTION*]... [-*T*] *SOURCE DEST*

mv [*OPTION*]... *SOURCE... DIRECTORY*

mv [*OPTION*]... -*t* *DIRECTORY SOURCE...*

DESCRIPTION

Rename *SOURCE* to *DEST*, or move *SOURCE*(s) to *DIRECTORY*.

Mandatory arguments to long options are mandatory for short options too.

--backup[=*CONTROL*]

make a backup of each existing destination file

-b like **--backup** but does not accept an argument

-f, --force

do not prompt before overwriting (equivalent to **--reply=yes**)

-i, --interactive

prompt before overwrite (equivalent to **--reply=query**)

--reply={yes,no,query}

specify how to handle the prompt about an existing destination file

--strip-trailing-slashes remove any trailing slashes from each *SOURCE* argument

-S, --suffix=*SUFFIX*

override the usual backup suffix

-t, --target-directory=*DIRECTORY*

move all *SOURCE* arguments into *DIRECTORY*

-T, --no-target-directory

treat *DEST* as a normal file

-u, --update

move only when the *SOURCE* file is newer than the destination file or when the destination file is missing

-v, --verbose

explain what is being done

--help display this help and exit

--version

output version information and exit

The backup suffix is '~', unless set with **--suffix** or `SIMPLE_BACKUP_SUFFIX`. The version control method may be selected via the **--backup** option or through the `VERSION_CONTROL` environment variable. Here are the values:

none, off

never make backups (even if **--backup** is given)

numbered, t

make numbered backups

existing, nil

numbered if numbered backups exist, simple otherwise

simple, never

always make simple backups

AUTHOR

Written by Mike Parker, David MacKenzie, and Jim Meyering.

REPORTING BUGS

Report bugs to <bug-coreutils@gnu.org>.

COPYRIGHT

Copyright © 2004 Free Software Foundation, Inc.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

SEE ALSO

The full documentation for **mv** is maintained as a Texinfo manual. If the **info** and **mv** programs are properly installed at your site, the command

info mv

should give you access to the complete manual.

NAME

`nice` – run a program with modified scheduling priority

SYNOPSIS

nice [*OPTION*] [*COMMAND* [*ARG*]...]

DESCRIPTION

Run *COMMAND* with an adjusted nice value, which affects the scheduling priority. With no *COMMAND*, print the current nice value. Nice values range from **-20** (most favorable scheduling) to 19 (least favorable).

-n, --adjustment=*N*

add integer *N* to the nice value (default 10)

--help display this help and exit

--version

output version information and exit

AUTHOR

Written by David MacKenzie.

REPORTING BUGS

Report bugs to <bug-coreutils@gnu.org>.

COPYRIGHT

Copyright © 2004 Free Software Foundation, Inc.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

SEE ALSO

The full documentation for **nice** is maintained as a Texinfo manual. If the **info** and **nice** programs are properly installed at your site, the command

info nice

should give you access to the complete manual.

NAME

nl – number lines of files

SYNOPSIS

nl [*OPTION*]... [*FILE*]...

DESCRIPTION

Write each *FILE* to standard output, with line numbers added. With no *FILE*, or when *FILE* is –, read standard input.

Mandatory arguments to long options are mandatory for short options too.

- b, --body-numbering=STYLE**
use *STYLE* for numbering body lines
- d, --section-delimiter=CC**
use *CC* for separating logical pages
- f, --footer-numbering=STYLE**
use *STYLE* for numbering footer lines
- h, --header-numbering=STYLE**
use *STYLE* for numbering header lines
- i, --page-increment=NUMBER**
line number increment at each line
- l, --join-blank-lines=NUMBER**
group of *NUMBER* empty lines counted as one
- n, --number-format=FORMAT**
insert line numbers according to *FORMAT*
- p, --no-renumber**
do not reset line numbers at logical pages
- s, --number-separator=STRING**
add *STRING* after (possible) line number
- v, --first-page=NUMBER**
first line number on each logical page
- w, --number-width=NUMBER**
use *NUMBER* columns for line numbers
- help** display this help and exit
- version**
output version information and exit

By default, selects **-v1 -i1 -l1 -sTAB -w6 -nrn -hn -bt -fn**. *CC* are two delimiter characters for separating logical pages, a missing second character implies `..`. Type `\\` for `\`. *STYLE* is one of:

- a** number all lines
- t** number only nonempty lines
- n** number no lines
- pBRE** number only lines that contain a match for the basic regular expression, *BRE*

FORMAT is one of:

- ln** left justified, no leading zeros
- rn** right justified, no leading zeros
- rz** right justified, leading zeros

AUTHOR

Written by Scott Bartram and David MacKenzie.

REPORTING BUGS

Report bugs to <bug-coreutils@gnu.org>.

COPYRIGHT

Copyright © 2004 Free Software Foundation, Inc.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

SEE ALSO

The full documentation for **nl** is maintained as a Texinfo manual. If the **info** and **nl** programs are properly installed at your site, the command

info nl

should give you access to the complete manual.

NAME

nohup – run a command immune to hangups, with output to a non-tty

SYNOPSIS

nohup *COMMAND* [*ARG*]...

nohup *OPTION*

DESCRIPTION

Run *COMMAND*, ignoring hangup signals.

--help display this help and exit

--version

output version information and exit

AUTHOR

Written by Jim Meyering.

REPORTING BUGS

Report bugs to <bug-coreutils@gnu.org>.

COPYRIGHT

Copyright © 2004 Free Software Foundation, Inc.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

SEE ALSO

The full documentation for **nohup** is maintained as a Texinfo manual. If the **info** and **nohup** programs are properly installed at your site, the command

info nohup

should give you access to the complete manual.

NAME

od – dump files in octal and other formats

SYNOPSIS

od [*OPTION*]... [*FILE*]...

od [-*abcdfilosx*]... [*FILE*] [[+] *OFFSET*][.][*b*]

od --traditional [*OPTION*]... [*FILE*] [[+] *OFFSET*][.][*b*] [+] [*LABEL*][.][*b*]

DESCRIPTION

Write an unambiguous representation, octal bytes by default, of *FILE* to standard output. With more than one *FILE* argument, concatenate them in the listed order to form the input. With no *FILE*, or when *FILE* is –, read standard input.

All arguments to long options are mandatory for short options.

-A, --address-radix=*RADIX*

decide how file offsets are printed

-j, --skip-bytes=*BYTES*

skip *BYTES* input bytes first

-N, --read-bytes=*BYTES*

limit dump to *BYTES* input bytes

-S, --strings[=*BYTES***]**

output strings of at least *BYTES* graphic chars

-t, --format=*TYPE*

select output format or formats

-v, --output-duplicates

do not use * to mark line suppression

-w, --width[=*BYTES***]**

output *BYTES* bytes per output line

--traditional

accept arguments in traditional form

--help display this help and exit

--version

output version information and exit

Traditional format specifications may be intermixed; they accumulate:

-a same as **-t a**, select named characters

-b same as **-t o1**, select octal bytes

-c same as **-t c**, select ASCII characters or backslash escapes

-d same as **-t u2**, select unsigned decimal 2-byte units

-f same as **-t fF**, select floats

-i same as **-t dI**, select decimal ints

-l same as **-t dL**, select decimal longs

-o same as **-t o2**, select octal 2-byte units

-s same as **-t d2**, select decimal 2-byte units

-x same as **-t x2**, select hexadecimal 2-byte units

If first and second call formats both apply, the second format is assumed if the last operand begins with + or (if there are 2 operands) a digit. An *OFFSET* operand means **-j** *OFFSET*. *LABEL* is the pseudo-address at first byte printed, incremented when dump is progressing. For *OFFSET* and *LABEL*, a 0x or 0X prefix indicates hexadecimal; suffixes may be . for octal and b for multiply by 512.

TYPE is made up of one or more of these specifications:

a named character
c ASCII character or backslash escape
d[SIZE] signed decimal, SIZE bytes per integer
f[SIZE] floating point, SIZE bytes per integer
o[SIZE] octal, SIZE bytes per integer
u[SIZE] unsigned decimal, SIZE bytes per integer
x[SIZE] hexadecimal, SIZE bytes per integer

SIZE is a number. For TYPE in `doux`, SIZE may also be C for `sizeof(char)`, S for `sizeof(short)`, I for `sizeof(int)` or L for `sizeof(long)`. If TYPE is f, SIZE may also be F for `sizeof(float)`, D for `sizeof(double)` or L for `sizeof(long double)`.

RADIX is d for decimal, o for octal, x for hexadecimal or n for none. BYTES is hexadecimal with 0x or 0X prefix, it is multiplied by 512 with b suffix, by 1024 with k and by 1048576 with m. Adding a z suffix to any type adds a display of printable characters to the end of each line of output. **—string** without a number implies 3. **—width** without a number implies 32. By default, od uses **—A o —t d2 —w 16**.

AUTHOR

Written by Jim Meyering.

REPORTING BUGS

Report bugs to <bug-coreutils@gnu.org>.

COPYRIGHT

Copyright © 2004 Free Software Foundation, Inc.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

SEE ALSO

The full documentation for **od** is maintained as a Texinfo manual. If the **info** and **od** programs are properly installed at your site, the command

info od

should give you access to the complete manual.

NAME

paste – merge lines of files

SYNOPSIS

paste [*OPTION*]... [*FILE*]...

DESCRIPTION

Write lines consisting of the sequentially corresponding lines from each *FILE*, separated by TABs, to standard output. With no *FILE*, or when *FILE* is –, read standard input.

Mandatory arguments to long options are mandatory for short options too.

-d, --delimiters=LIST

reuse characters from *LIST* instead of TABs

-s, --serial

paste one file at a time instead of in parallel

--help display this help and exit

--version

output version information and exit

AUTHOR

Written by David M. Ihnat and David MacKenzie.

REPORTING BUGS

Report bugs to <bug-coreutils@gnu.org>.

COPYRIGHT

Copyright © 2004 Free Software Foundation, Inc.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

SEE ALSO

The full documentation for **paste** is maintained as a Texinfo manual. If the **info** and **paste** programs are properly installed at your site, the command

info paste

should give you access to the complete manual.

NAME

pathchk – check whether file names are valid or portable

SYNOPSIS

pathchk [*OPTION*]... *NAME*...

DESCRIPTION

Diagnose unportable constructs in *NAME*.

-p, --portability

check for all POSIX systems, not only this one

--help display this help and exit

--version

output version information and exit

AUTHOR

Written by Paul Eggert, David MacKenzie, and Jim Meyering.

REPORTING BUGS

Report bugs to <bug-coreutils@gnu.org>.

COPYRIGHT

Copyright © 2004 Free Software Foundation, Inc.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

SEE ALSO

The full documentation for **pathchk** is maintained as a Texinfo manual. If the **info** and **pathchk** programs are properly installed at your site, the command

info pathchk

should give you access to the complete manual.

NAME

pinky – lightweight finger

SYNOPSIS

pinky [*OPTION*]... [*USER*]...

DESCRIPTION

- l** produce long format output for the specified USERS
- b** omit the user's home directory and shell in long format
- h** omit the user's project file in long format
- p** omit the user's plan file in long format
- s** do short format output, this is the default
- f** omit the line of column headings in short format
- w** omit the user's full name in short format
- i** omit the user's full name and remote host in short format
- q** omit the user's full name, remote host and idle time in short format
- help** display this help and exit
- version**
output version information and exit

A lightweight 'finger' program; print user information. The utmp file will be /var/run/utmp.

AUTHOR

Written by Joseph Arceneaux, David MacKenzie, and Kaveh Ghazi.

REPORTING BUGS

Report bugs to <bug-coreutils@gnu.org>.

COPYRIGHT

Copyright © 2004 Free Software Foundation, Inc.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

SEE ALSO

The full documentation for **pinky** is maintained as a Texinfo manual. If the **info** and **pinky** programs are properly installed at your site, the command

info pinky

should give you access to the complete manual.

NAME

pr – convert text files for printing

SYNOPSIS

pr [*OPTION*]... [*FILE*]...

DESCRIPTION

Paginate or columnate *FILE*(s) for printing.

Mandatory arguments to long options are mandatory for short options too.

+FIRST_PAGE[:LAST_PAGE], **--pages=FIRST_PAGE[:LAST_PAGE]**
begin [stop] printing with page FIRST_[LAST_]PAGE

-COLUMN, **--columns=COLUMN**
output *COLUMN* columns and print columns down, unless **-a** is used. Balance number of lines in the columns on each page.

-a, **--across**
print columns across rather than down, used together with **-COLUMN**

-c, **--show-control-chars**
use hat notation (^G) and octal backslash notation

-d, **--double-space**
double space the output

-D, **--date-format=FORMAT**
use *FORMAT* for the header date

-e[CHAR[WIDTH]], **--expand-tabs[=CHAR[WIDTH]]**
expand input CHARs (TABs) to tab WIDTH (8)

-F, **-f**, **--form-feed**
use form feeds instead of newlines to separate pages (by a 3-line page header with **-F** or a 5-line header and trailer without **-F**)

-h *HEADER*, **--header=HEADER**
use a centered *HEADER* instead of filename in page header, **-h ""** prints a blank line, don't use **-h ""**

-i[CHAR[WIDTH]], **--output-tabs[=CHAR[WIDTH]]**
replace spaces with CHARs (TABs) to tab WIDTH (8)

-J, **--join-lines**
merge full lines, turns off **-W** line truncation, no column alignment, **--sep-string[=STRING]** sets separators

-I *PAGE_LENGTH*, **--length=PAGE_LENGTH**
set the page length to *PAGE_LENGTH* (66) lines (default number of lines of text 56, and with **-F** 63)

-m, **--merge**
print all files in parallel, one in each column, truncate lines, but join lines of full length with **-J**

-n[SEP[DIGITS]], **--number-lines[=SEP[DIGITS]]**
number lines, use *DIGITS* (5) digits, then *SEP* (TAB), default counting starts with 1st line of input file

-N *NUMBER*, **--first-line-number=NUMBER**
start counting with *NUMBER* at 1st line of first page printed (see **+FIRST_PAGE**)

-o *MARGIN*, **--indent=MARGIN**
offset each line with *MARGIN* (zero) spaces, do not affect **-w** or **-W**, *MARGIN* will be added to *PAGE_WIDTH*

- r, --no-file-warnings**
omit warning when a file cannot be opened
- s[CHAR], --separator[=CHAR]**
separate columns by a single character, default for CHAR is the <TAB> character without **-w** and 'no char' with **-w -s[CHAR]** turns off line truncation of all 3 column options (**-COLUMN|-a -COLUMN|-m**) except **-w** is set
- SSTRING, --sep-string[=STRING]**
separate columns by STRING, without **-S**: Default separator <TAB> with **-J** and <space> otherwise (same as **-S" "**), no effect on column options
- t, --omit-header** omit page headers and trailers
- T, --omit-pagination**
omit page headers and trailers, eliminate any pagination by form feeds set in input files
- v, --show-nonprinting**
use octal backslash notation
- w PAGE_WIDTH, --width=PAGE_WIDTH**
set page width to PAGE_WIDTH (72) characters for multiple text-column output only, **-s[char]** turns off (72)
- W PAGE_WIDTH, --page-width=PAGE_WIDTH**
set page width to PAGE_WIDTH (72) characters always, truncate lines, except **-J** option is set, no interference with **-S** or **-s**
- help** display this help and exit
- version**
output version information and exit
- T** implied by **-l nn** when $nn \leq 10$ or ≤ 3 with **-F**. With no FILE, or when FILE is **-**, read standard input.

AUTHOR

Written by Pete TerMaat and Roland Huebner.

REPORTING BUGS

Report bugs to <bug-coreutils@gnu.org>.

COPYRIGHT

Copyright © 2005 Free Software Foundation, Inc.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

SEE ALSO

The full documentation for **pr** is maintained as a Texinfo manual. If the **info** and **pr** programs are properly installed at your site, the command

info pr

should give you access to the complete manual.

NAME

printenv – print all or part of environment

SYNOPSIS

printenv [*VARIABLE*]...

printenv *OPTION*

DESCRIPTION

If no environment *VARIABLE* specified, print them all.

--help display this help and exit

--version

output version information and exit

AUTHOR

Written by David MacKenzie and Richard Mlynarik.

REPORTING BUGS

Report bugs to <bug-coreutils@gnu.org>.

COPYRIGHT

Copyright © 2004 Free Software Foundation, Inc.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

SEE ALSO

The full documentation for **printenv** is maintained as a Texinfo manual. If the **info** and **printenv** programs are properly installed at your site, the command

info printenv

should give you access to the complete manual.

NAME

printf – format and print data

SYNOPSIS

printf *FORMAT* [*ARGUMENT*]...

printf *OPTION*

DESCRIPTION

NOTE: your shell may have its own version of printf which will supersede the version described here. Please refer to your shell's documentation for details about the options it supports.

Print ARGUMENT(s) according to FORMAT.

--help display this help and exit

--version

output version information and exit

FORMAT controls the output as in C printf. Interpreted sequences are:

\ " double quote

\NNN character with octal value NNN (1 to 3 digits)

\\ backslash

\a alert (BEL)

\b backspace

\c produce no further output

\f form feed

\n new line

\r carriage return

\t horizontal tab

\v vertical tab

\xHH byte with hexadecimal value HH (1 to 2 digits)

\uHHHH

Unicode (ISO/IEC 10646) character with hex value HHHH (4 digits)

\UHHHHHHHH

Unicode character with hex value HHHHHHHH (8 digits)

%% a single %

%b ARGUMENT as a string with ‘\’ escapes interpreted,
except that octal escapes are of the form \0 or \0NNN

and all C format specifications ending with one of diouxXfeEgGcs, with ARGUMENTs converted to proper type first. Variable widths are handled.

AUTHOR

Written by David MacKenzie.

REPORTING BUGS

Report bugs to <bug-coreutils@gnu.org>.

COPYRIGHT

Copyright © 2004 Free Software Foundation, Inc.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

SEE ALSO

The full documentation for **printf** is maintained as a Texinfo manual. If the **info** and **printf** programs are properly installed at your site, the command

info printf

should give you access to the complete manual.

NAME

ptx – produce a permuted index of file contents

SYNOPSIS

ptx [*OPTION*]... [*INPUT*]... (*without -G*)

ptx -G [*OPTION*]... [*INPUT*] [*OUTPUT*]

DESCRIPTION

Output a permuted index, including context, of the words in the input files.

Mandatory arguments to long options are mandatory for short options too.

- A, --auto-reference**
output automatically generated references
- C, --copyright**
display Copyright and copying conditions
- G, --traditional**
behave more like System V ‘ptx’
- F, --flag-truncation=STRING**
use STRING for flagging line truncations
- M, --macro-name=STRING**
macro name to use instead of ‘xx’
- O, --format=roff**
generate output as roff directives
- R, --right-side-refs**
put references at right, not counted in **-w**
- S, --sentence-regexp=REGEXP**
for end of lines or end of sentences
- T, --format=tex**
generate output as TeX directives
- W, --word-regexp=REGEXP**
use REGEXP to match each keyword
- b, --break-file=FILE**
word break characters in this FILE
- f, --ignore-case**
fold lower case to upper case for sorting
- g, --gap-size=NUMBER**
gap size in columns between output fields
- i, --ignore-file=FILE**
read ignore word list from FILE
- o, --only-file=FILE**
read only word list from this FILE
- r, --references**
first field of each line is a reference
- t, --typeset-mode** – not implemented –
- w, --width=NUMBER**
output width in columns, reference excluded
- help** display this help and exit
- version**
output version information and exit

With no FILE or if FILE is –, read Standard Input. ‘-F /’ by default.

AUTHOR

Written by F. Pinard.

REPORTING BUGS

Report bugs to <bug-coreutils@gnu.org>.

COPYRIGHT

Copyright © 2004 Free Software Foundation, Inc.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

SEE ALSO

The full documentation for **ptx** is maintained as a Texinfo manual. If the **info** and **ptx** programs are properly installed at your site, the command

info ptx

should give you access to the complete manual.

NAME

`pwd` – print name of current/working directory

SYNOPSIS

`pwd` [*OPTION*]

DESCRIPTION

NOTE: your shell may have its own version of `pwd` which will supersede the version described here. Please refer to your shell's documentation for details about the options it supports.

Print the full filename of the current working directory.

--help display this help and exit

--version

output version information and exit

AUTHOR

Written by Jim Meyering.

REPORTING BUGS

Report bugs to <bug-coreutils@gnu.org>.

COPYRIGHT

Copyright © 2004 Free Software Foundation, Inc.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

SEE ALSO

The full documentation for **pwd** is maintained as a Texinfo manual. If the **info** and **pwd** programs are properly installed at your site, the command

info pwd

should give you access to the complete manual.

NAME

readlink – display value of a symbolic link

SYNOPSIS

readlink [*OPTION*]... *FILE*

DESCRIPTION

Display value of a symbolic link on standard output.

-f, --canonicalize

canonicalize by following every symlink in every component of the given path recursively; all but the last path component must exist

-e, --canonicalize-existing

canonicalize by following every symlink in every component of the given path recursively, all path components must exist

-m, --canonicalize-missing

canonicalize by following every symlink in every component of the given path recursively, without requirements on components existence

-n, --no-newline

do not output the trailing newline

-q, --quiet,

-s, --silent

suppress most error messages

-v, --verbose

report error messages

--help display this help and exit

--version

output version information and exit

AUTHOR

Written by Dmitry V. Levin.

REPORTING BUGS

Report bugs to <bug-coreutils@gnu.org>.

COPYRIGHT

Copyright © 2004 Free Software Foundation, Inc.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

SEE ALSO

The full documentation for **readlink** is maintained as a Texinfo manual. If the **info** and **readlink** programs are properly installed at your site, the command

info readlink

should give you access to the complete manual.

NAME

rm – remove files or directories

SYNOPSIS

rm [*OPTION*]... *FILE*...

DESCRIPTION

This manual page documents the GNU version of **rm**. **rm** removes each specified file. By default, it does not remove directories.

If a file is unwritable, the standard input is a tty, and the *-f* or *--force* option is not given, **rm** prompts the user for whether to remove the file. If the response is not affirmative, the file is skipped.

OPTIONS

Remove (unlink) the *FILE*(s).

-d, --directory

unlink *FILE*, even if it is a non-empty directory (super-user only; this works only if your system

supports ‘unlink’ for nonempty directories)

-f, --force

ignore nonexistent files, never prompt

-i, --interactive

prompt before any removal

--no-preserve-root do not treat ‘/’ specially (the default)

--preserve-root

fail to operate recursively on ‘/’

-r, -R, --recursive

remove the contents of directories recursively

-v, --verbose

explain what is being done

--help display this help and exit

--version

output version information and exit

To remove a file whose name starts with a ‘-’, for example ‘-foo’, use one of these commands:

rm -- -foo

rm ./-foo

Note that if you use **rm** to remove a file, it is usually possible to recover the contents of that file. If you want more assurance that the contents are truly unrecoverable, consider using **shred**.

AUTHOR

Written by Paul Rubin, David MacKenzie, Richard Stallman, and Jim Meyering.

REPORTING BUGS

Report bugs to <bug-coreutils@gnu.org>.

COPYRIGHT

Copyright © 2004 Free Software Foundation, Inc.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

SEE ALSO

chattr(1), shred(1)

The full documentation for **rm** is maintained as a Texinfo manual. If the **info** and **rm** programs are properly installed at your site, the command

info rm

should give you access to the complete manual.

NAME

rmdir – remove empty directories

SYNOPSIS

rmdir [*OPTION*]... *DIRECTORY*...

DESCRIPTION

Remove the *DIRECTORY*(ies), if they are empty.

--ignore-fail-on-non-empty

ignore each failure that is solely because a directory is non-empty

-p, --parents

remove *DIRECTORY*, then try to remove each directory component of that path name. E.g., ‘**rmdir -p a/b/c**’ is similar to ‘**rmdir a/b/c a/b a**’.

-v, --verbose

output a diagnostic for every directory processed

--help display this help and exit

--version

output version information and exit

AUTHOR

Written by David MacKenzie.

REPORTING BUGS

Report bugs to <bug-coreutils@gnu.org>.

COPYRIGHT

Copyright © 2004 Free Software Foundation, Inc.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

SEE ALSO

The full documentation for **rmdir** is maintained as a Texinfo manual. If the **info** and **rmdir** programs are properly installed at your site, the command

info rmdir

should give you access to the complete manual.

NAME

seq – print a sequence of numbers

SYNOPSIS

seq [*OPTION*]... *LAST*

seq [*OPTION*]... *FIRST LAST*

seq [*OPTION*]... *FIRST INCREMENT LAST*

DESCRIPTION

Print numbers from *FIRST* to *LAST*, in steps of *INCREMENT*.

-f, --format=FORMAT

use printf style floating-point *FORMAT* (default: %g)

-s, --separator=STRING

use *STRING* to separate numbers (default: \n)

-w, --equal-width

equalize width by padding with leading zeroes

--help display this help and exit

--version

output version information and exit

If *FIRST* or *INCREMENT* is omitted, it defaults to 1. That is, an omitted *INCREMENT* defaults to 1 even when *LAST* is smaller than *FIRST*. *FIRST*, *INCREMENT*, and *LAST* are interpreted as floating point values. *INCREMENT* is usually positive if *FIRST* is smaller than *LAST*, and *INCREMENT* is usually negative if *FIRST* is greater than *LAST*. When given, the *FORMAT* argument must contain exactly one of the printf-style, floating point output formats %e, %f, %g

AUTHOR

Written by Ulrich Drepper.

REPORTING BUGS

Report bugs to <bug-coreutils@gnu.org>.

COPYRIGHT

Copyright © 2004 Free Software Foundation, Inc.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

SEE ALSO

The full documentation for **seq** is maintained as a Texinfo manual. If the **info** and **seq** programs are properly installed at your site, the command

info seq

should give you access to the complete manual.

NAME

sha1sum – compute and check SHA1 message digest

SYNOPSIS

sha1sum [*OPTION*] [*FILE*]...

sha1sum [*OPTION*] *--check* [*FILE*]

DESCRIPTION

Print or check SHA1 (160-bit) checksums. With no *FILE*, or when *FILE* is –, read standard input.

-b, --binary

read files in binary mode (default on DOS/Windows)

-c, --check

check SHA1 sums against given list

-t, --text

read files in text mode (default)

The following two options are useful only when verifying checksums:

--status

don't output anything, status code shows success

-w, --warn

warn about improperly formatted checksum lines

--help display this help and exit

--version

output version information and exit

The sums are computed as described in FIPS-180-1. When checking, the input should be a former output of this program. The default mode is to print a line with checksum, a character indicating type (* for binary, ' for text), and name for each *FILE*.

AUTHOR

Written by Ulrich Drepper and Scott Miller.

REPORTING BUGS

Report bugs to <bug-coreutils@gnu.org>.

COPYRIGHT

Copyright © 2004 Free Software Foundation, Inc.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

SEE ALSO

The full documentation for **sha1sum** is maintained as a Texinfo manual. If the **info** and **sha1sum** programs are properly installed at your site, the command

info sha1sum

should give you access to the complete manual.

NAME

shred – overwrite a file to hide its contents, and optionally delete it

SYNOPSIS

shred [*OPTIONS*] *FILE* [...]

DESCRIPTION

Overwrite the specified FILE(s) repeatedly, in order to make it harder for even very expensive hardware probing to recover the data.

Mandatory arguments to long options are mandatory for short options too.

-f, --force

change permissions to allow writing if necessary

-n, --iterations=N

Overwrite N times instead of the default (25)

-s, --size=N

shred this many bytes (suffixes like K, M, G accepted)

-u, --remove

truncate and remove file after overwriting

-v, --verbose

show progress

-x, --exact

do not round file sizes up to the next full block;

this is the default for non-regular files

-z, --zero

add a final overwrite with zeros to hide shredding

--help display this help and exit

--version

output version information and exit

If FILE is –, shred standard output.

Delete FILE(s) if **--remove** (**-u**) is specified. The default is not to remove the files because it is common to operate on device files like /dev/hda, and those files usually should not be removed. When operating on regular files, most people use the **--remove** option.

CAUTION: Note that shred relies on a very important assumption: that the file system overwrites data in place. This is the traditional way to do things, but many modern file system designs do not satisfy this assumption. The following are examples of file systems on which shred is not effective:

- * log-structured or journaled file systems, such as those supplied with

 - AIX and Solaris (and JFS, ReiserFS, XFS, Ext3, etc.)

- * file systems that write redundant data and carry on even if some writes

 - fail, such as RAID-based file systems

- * file systems that make snapshots, such as Network Appliance's NFS server

- * file systems that cache in temporary locations, such as NFS

 - version 3 clients

- * compressed file systems

In addition, file system backups and remote mirrors may contain copies of the file that cannot be removed, and that will allow a shredded file to be recovered later.

AUTHOR

Written by Colin Plumb.

REPORTING BUGS

Report bugs to <bug-coreutils@gnu.org>.

COPYRIGHT

Copyright © 2005 Free Software Foundation, Inc.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

SEE ALSO

The full documentation for **shred** is maintained as a Texinfo manual. If the **info** and **shred** programs are properly installed at your site, the command

info shred

should give you access to the complete manual.

NAME

sleep – delay for a specified amount of time

SYNOPSIS

sleep *NUMBER*[*SUFFIX*]...

sleep *OPTION*

DESCRIPTION

Pause for *NUMBER* seconds. *SUFFIX* may be ‘s’ for seconds (the default), ‘m’ for minutes, ‘h’ for hours or ‘d’ for days. Unlike most implementations that require *NUMBER* be an integer, here *NUMBER* may be an arbitrary floating point number.

--help display this help and exit

--version

output version information and exit

AUTHOR

Written by Jim Meyering and Paul Eggert.

REPORTING BUGS

Report bugs to <bug-coreutils@gnu.org>.

COPYRIGHT

Copyright © 2004 Free Software Foundation, Inc.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

SEE ALSO

The full documentation for **sleep** is maintained as a Texinfo manual. If the **info** and **sleep** programs are properly installed at your site, the command

info sleep

should give you access to the complete manual.

NAME

sort – sort lines of text files

SYNOPSIS

sort [*OPTION*]... [*FILE*]...

DESCRIPTION

Write sorted concatenation of all *FILE*(s) to standard output.

Ordering options:

Mandatory arguments to long options are mandatory for short options too.

-b, --ignore-leading-blanks ignore leading blanks

-d, --dictionary-order
consider only blanks and alphanumeric characters

-f, --ignore-case
fold lower case to upper case characters

-g, --general-numeric-sort
compare according to general numerical value

-i, --ignore-nonprinting
consider only printable characters

-M, --month-sort
compare (unknown) < ‘JAN’ < ... < ‘DEC’

-n, --numeric-sort
compare according to string numerical value

-r, --reverse
reverse the result of comparisons

Other options:

-c, --check
check whether input is sorted; do not sort

-k, --key=POS1[,POS2]
start a key at POS1, end it at POS 2 (origin 1)

-m, --merge
merge already sorted files; do not sort

-o, --output=FILE
write result to FILE instead of standard output

-s, --stable
stabilize sort by disabling last-resort comparison

-S, --buffer-size=SIZE
use SIZE for main memory buffer

-t, --field-separator=SEP use SEP instead of non-blank to blank transition

-T, --temporary-directory=DIR
use DIR for temporaries, not \$TMPDIR or /tmp; multiple options specify multiple directories

-u, --unique
with **-c**, check for strict ordering; without **-c**, output only the first of an equal run

-z, --zero-terminated
end lines with 0 byte, not newline

--help display this help and exit

--version
output version information and exit

POS is F[C][OPTS], where F is the field number and C the character position in the field. OPTS is one

or more single-letter ordering options, which override global ordering options for that key. If no key is given, use the entire line as the key.

SIZE may be followed by the following multiplicative suffixes: % 1% of memory, b 1, K 1024 (default), and so on for M, G, T, P, E, Z, Y.

With no FILE, or when FILE is `–`, read standard input.

*** WARNING *** The locale specified by the environment affects sort order. Set `LC_ALL=C` to get the traditional sort order that uses native byte values.

AUTHOR

Written by Mike Haertel and Paul Eggert.

REPORTING BUGS

Report bugs to [<bug-coreutils@gnu.org>](mailto:bug-coreutils@gnu.org).

COPYRIGHT

Copyright © 2004 Free Software Foundation, Inc.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

SEE ALSO

The full documentation for **sort** is maintained as a Texinfo manual. If the **info** and **sort** programs are properly installed at your site, the command

info sort

should give you access to the complete manual.

NAME

split – split a file into pieces

SYNOPSIS

split [*OPTION*] [*INPUT* [*PREFIX*]]

DESCRIPTION

Output fixed-size pieces of *INPUT* to *PREFIX*aa, *PREFIX*ab, ...; default *PREFIX* is 'x'. With no *INPUT*, or when *INPUT* is –, read standard input.

Mandatory arguments to long options are mandatory for short options too.

-a, --suffix-length=N

use suffixes of length *N* (default 2)

-b, --bytes=SIZE

put *SIZE* bytes per output file

-C, --line-bytes=SIZE

put at most *SIZE* bytes of lines per output file

-d, --numeric-suffixes

use numeric suffixes instead of alphabetic

-l, --lines=NUMBER

put *NUMBER* lines per output file

--verbose

print a diagnostic to standard error just before each output file is opened

--help display this help and exit

--version

output version information and exit

SIZE may have a multiplier suffix: b for 512, k for 1K, m for 1 Meg.

AUTHOR

Written by Torbjorn Granlund and Richard M. Stallman.

REPORTING BUGS

Report bugs to <bug-coreutils@gnu.org>.

COPYRIGHT

Copyright © 2004 Free Software Foundation, Inc.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

SEE ALSO

The full documentation for **split** is maintained as a Texinfo manual. If the **info** and **split** programs are properly installed at your site, the command

info split

should give you access to the complete manual.

NAME

stat – display file or file system status

SYNOPSIS

stat [*OPTION*] *FILE*...

DESCRIPTION

Display file or file system status.

-f, --file-system

display file system status instead of file status

-c --format=FORMAT

use the specified FORMAT instead of the default

-L, --dereference

follow links

-t, --terse

print the information in terse form

--help display this help and exit

--version

output version information and exit

The valid format sequences for files (without **--file-system**):

%A	Access rights in human readable form
%a	Access rights in octal
%B	The size in bytes of each block reported by ‘%b’
%b	Number of blocks allocated (see %B)
%D	Device number in hex
%d	Device number in decimal
%F	File type
%f	Raw mode in hex
%G	Group name of owner
%g	Group ID of owner
%h	Number of hard links
%i	Inode number
%N	Quoted File name with dereference if symbolic link
%n	File name
%o	IO block size
%s	Total size, in bytes
%T	Minor device type in hex
%t	Major device type in hex
%U	User name of owner
%u	User ID of owner
%X	Time of last access as seconds since Epoch
%x	Time of last access
%Y	Time of last modification as seconds since Epoch
%y	Time of last modification
%Z	Time of last change as seconds since Epoch

%z Time of last change

Valid format sequences for file systems:

%a Free blocks available to non-superuser

%b Total data blocks in file system

%c Total file nodes in file system

%d Free file nodes in file system

%f Free blocks in file system

%i File System id in hex

%l Maximum length of filenames

%n File name

%s Optimal transfer block size

%T Type in human readable form

%t Type in hex

AUTHOR

Written by Michael Meskes.

REPORTING BUGS

Report bugs to <bug-coreutils@gnu.org>.

COPYRIGHT

Copyright © 2004 Free Software Foundation, Inc.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

SEE ALSO

The full documentation for **stat** is maintained as a Texinfo manual. If the **info** and **stat** programs are properly installed at your site, the command

info stat

should give you access to the complete manual.

NAME

stty – change and print terminal line settings

SYNOPSIS

stty [-F *DEVICE*] [--file=*DEVICE*] [*SETTING*]...

stty [-F *DEVICE*] [--file=*DEVICE*] [-a/--all]

stty [-F *DEVICE*] [--file=*DEVICE*] [-g/--save]

DESCRIPTION

Print or change terminal characteristics.

-a, --all

print all current settings in human-readable form

-g, --save

print all current settings in a stty-readable form

-F, --file=DEVICE

open and use the specified *DEVICE* instead of stdin

--help display this help and exit

--version

output version information and exit

Optional – before *SETTING* indicates negation. An * marks non-POSIX settings. The underlying system defines which settings are available.

Special characters:

* dsusp *CHAR*

CHAR will send a terminal stop signal once input flushed

eof *CHAR*

CHAR will send an end of file (terminate the input)

eol *CHAR*

CHAR will end the line

* eol2 *CHAR*

alternate *CHAR* for ending the line

erase *CHAR*

CHAR will erase the last character typed

intr *CHAR*

CHAR will send an interrupt signal

kill *CHAR*

CHAR will erase the current line

* lnext *CHAR*

CHAR will enter the next character quoted

quit *CHAR*

CHAR will send a quit signal

* rprnt *CHAR*

CHAR will redraw the current line

start *CHAR*

CHAR will restart the output after stopping it

stop *CHAR*

CHAR will stop the output

susp *CHAR*

CHAR will send a terminal stop signal

* swch CHAR
CHAR will switch to a different shell layer

* werase CHAR
CHAR will erase the last word typed

Special settings:

N set the input and output speeds to N bauds

* cols N
tell the kernel that the terminal has N columns

* columns N
same as cols N

ispeed N
set the input speed to N

* line N
use line discipline N

min N with **-icanon**, set N characters minimum for a completed read

ospeed N
set the output speed to N

* rows N
tell the kernel that the terminal has N rows

* size print the number of rows and columns according to the kernel

speed print the terminal speed

time N with **-icanon**, set read timeout of N tenths of a second

Control settings:

[−]clocal
disable modem control signals

[−]cread
allow input to be received

* [−]crtcts
enable RTS/CTS handshaking

csN set character size to N bits, N in [5..8]

[−]cstopb
use two stop bits per character (one with ‘−’)

[−]hup send a hangup signal when the last process closes the tty

[−]hupcl
same as [−]hup

[−]parenb
generate parity bit in output and expect parity bit in input

[−]parodd
set odd parity (even with ‘−’)

Input settings:

[−]brkint
breaks cause an interrupt signal

[−]icrnl translate carriage return to newline

[−]ignbrk
ignore break characters

[−]igncr
ignore carriage return

[-]ignpar
 ignore characters with parity errors
 * [-]imaxbel
 beep and do not flush a full input buffer on a character
 [-]inlcr translate newline to carriage return
 [-]inpck
 enable input parity checking
 [-]istrip
 clear high (8th) bit of input characters
 * [-]iutf8
 assume input characters are UTF-8 encoded
 * [-]iuclc
 translate uppercase characters to lowercase
 * [-]ixany
 let any character restart output, not only start character
 [-]ixoff
 enable sending of start/stop characters
 [-]ixon enable XON/XOFF flow control
 [-]parmrk
 mark parity errors (with a 255-0-character sequence)
 [-]tandem
 same as [-]ixoff

Output settings:

* bsN backspace delay style, N in [0..1]
 * crN carriage return delay style, N in [0..3]
 * ffN form feed delay style, N in [0..1]
 * nlN newline delay style, N in [0..1]
 * [-]ocrnl
 translate carriage return to newline
 * [-]ofdel
 use delete characters for fill instead of null characters
 * [-]ofill
 use fill (padding) characters instead of timing for delays
 * [-]olcuc
 translate lowercase characters to uppercase
 * [-]onlcr
 translate newline to carriage return-newline
 * [-]onlret
 newline performs a carriage return
 * [-]onocr
 do not print carriage returns in the first column
 [-]opost
 postprocess output
 * tabN horizontal tab delay style, N in [0..3]
 * tabs same as tab0
 * -**tabs** same as tab3
 * vtN vertical tab delay style, N in [0..1]

Local settings:

- `[-]crterase`
echo erase characters as backspace–space–backspace
- * `crtkill`
kill all line by obeying the `echoprt` and `echoe` settings
- * **`-crtkill`**
kill all line by obeying the `echoctl` and `echok` settings
- * `[-]ctlecho`
echo control characters in hat notation (`^c`)
- `[-]echo`
echo input characters
- * `[-]echoctl`
same as `[-]ctlecho`
- `[-]echoe`
same as `[-]crterase`
- `[-]echok`
echo a newline after a kill character
- * `[-]echoke`
same as `[-]crtkill`
- `[-]echonl`
echo newline even if not echoing other characters
- * `[-]echoprt`
echo erased characters backward, between ``` and `'`
- `[-]icanon`
enable erase, kill, werase, and `rprnt` special characters
- `[-]ixten`
enable non-POSIX special characters
- `[-]isig` enable interrupt, quit, and suspend special characters
- `[-]noflsh`
disable flushing after interrupt and quit special characters
- * `[-]prterase`
same as `[-]echoprt`
- * `[-]tostop`
stop background jobs that try to write to the terminal
- * `[-]xcase`
with `icanon`, escape with ``` for uppercase characters

Combination settings:

- * `[-]LCASE`
same as `[-]lcase`
- `cbreak` same as **`-icanon`**
- `-cbreak`**
same as `icanon`
- `cooked` same as `brkint ignpar istrip icrnl ixon opost isig icanon`, eof and eol characters to their default values
- `-cooked`**
same as `raw`
- `crt` same as `echoe echoctl echoke`
- `dec` same as `echoe echoctl echoke -ixany intr ^c erase 0177 kill ^u`

* **[−]decctlq**
 same as **[−]ixany**

ek erase and kill characters to their default values

evenp same as **parenb −parodd cs7**

−evenp same as **−parenb cs8**

* **[−]lcase**
 same as **xcase iuclc olcuc**

litout same as **−parenb −istrip −opost cs8**

−litout same as **parenb istrip opost cs7**

nl same as **−icrnl −onlcr**

−nl same as **icrnl −inlcr −igncr onlcr −ocrnl −onlret**

oddp same as **parenb parodd cs7**

−oddp same as **−parenb cs8**

[−]parity
 same as **[−]evenp**

pass8 same as **−parenb −istrip cs8**

−pass8 same as **parenb istrip cs7**

raw same as **−ignbrk −brkint −ignpar −parmrk −inpck −istrip −inlcr −igncr −icrnl −ixon −ixoff −iuclc −ixany −imaxbel −opost −isig −icanon −xcase min 1 time 0**

−raw same as **cooked**

sane same as **cread −ignbrk brkint −inlcr −igncr icrnl −iutf8 −ixoff −iuclc −ixany imaxbel opost −olcuc −ocrnl onlcr −onocr −onlret −ofill −ofdel nl0 cr0 tab0 bs0 vt0 ff0 isig icanon iexten echo echoe echok −echonl −nofish −xcase −tostop −echoprt echotl echoke, all special characters to their default values.**

Handle the tty line connected to standard input. Without arguments, prints baud rate, line discipline, and deviations from stty sane. In settings, CHAR is taken literally, or coded as in ^c, 0x37, 0177 or 127; special values ^− or undef used to disable special characters.

AUTHOR

Written by David MacKenzie.

REPORTING BUGS

Report bugs to <bug-coreutils@gnu.org>.

COPYRIGHT

Copyright © 2004 Free Software Foundation, Inc.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

SEE ALSO

The full documentation for **stty** is maintained as a Texinfo manual. If the **info** and **stty** programs are properly installed at your site, the command

info stty

should give you access to the complete manual.

NAME

su – run a shell with substitute user and group IDs

SYNOPSIS

su [*OPTION*]... [-] [*USER* [*ARG*]...]

DESCRIPTION

Change the effective user id and group id to that of *USER*.

–, –l, –login

make the shell a login shell

–c, –command=COMMAND

pass a single *COMMAND* to the shell with **–c**

–f, –fast

pass **–f** to the shell (for *csh* or *tcsh*)

–m, –preserve-environment

do not reset environment variables

–p same as **–m**

–s, –shell=SHELL

run *SHELL* if */etc/shells* allows it

–help display this help and exit

–version

output version information and exit

A mere **–** implies **–l**. If *USER* not given, assume root.

AUTHOR

Written by David MacKenzie.

REPORTING BUGS

Report bugs to <bug-coreutils@gnu.org>.

COPYRIGHT

Copyright © 2004 Free Software Foundation, Inc.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

SEE ALSO

The full documentation for **su** is maintained as a Texinfo manual. If the **info** and **su** programs are properly installed at your site, the command

info su

should give you access to the complete manual.

NAME

sum – checksum and count the blocks in a file

SYNOPSIS

sum [*OPTION*]... [*FILE*]...

DESCRIPTION

Print checksum and block counts for each FILE.

-r defeat **-s**, use BSD sum algorithm, use 1K blocks

-s, --sysv
 use System V sum algorithm, use 512 bytes blocks

--help display this help and exit

--version
 output version information and exit

With no FILE, or when FILE is –, read standard input.

AUTHOR

Written by Kayvan Aghaiepour and David MacKenzie.

REPORTING BUGS

Report bugs to <bug-coreutils@gnu.org>.

COPYRIGHT

Copyright © 2004 Free Software Foundation, Inc.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

SEE ALSO

The full documentation for **sum** is maintained as a Texinfo manual. If the **info** and **sum** programs are properly installed at your site, the command

info sum

should give you access to the complete manual.

NAME

sync – flush file system buffers

SYNOPSIS

sync [*OPTION*]

DESCRIPTION

Force changed blocks to disk, update the super block.

—help display this help and exit

—version
output version information and exit

AUTHOR

Written by Jim Meyering.

REPORTING BUGS

Report bugs to <bug-coreutils@gnu.org>.

COPYRIGHT

Copyright © 2004 Free Software Foundation, Inc.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

SEE ALSO

The full documentation for **sync** is maintained as a Texinfo manual. If the **info** and **sync** programs are properly installed at your site, the command

info sync

should give you access to the complete manual.

NAME

tac – concatenate and print files in reverse

SYNOPSIS

tac [*OPTION*]... [*FILE*]...

DESCRIPTION

Write each *FILE* to standard output, last line first. With no *FILE*, or when *FILE* is **-**, read standard input.

Mandatory arguments to long options are mandatory for short options too.

-b, --before

attach the separator before instead of after

-r, --regex

interpret the separator as a regular expression

-s, --separator=STRING

use *STRING* as the separator instead of newline

--help display this help and exit

--version

output version information and exit

AUTHOR

Written by Jay Lepreau and David MacKenzie.

REPORTING BUGS

Report bugs to <bug-coreutils@gnu.org>.

COPYRIGHT

Copyright © 2004 Free Software Foundation, Inc.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

SEE ALSO

The full documentation for **tac** is maintained as a Texinfo manual. If the **info** and **tac** programs are properly installed at your site, the command

info tac

should give you access to the complete manual.

NAME

tail – output the last part of files

SYNOPSIS

tail [*OPTION*]... [*FILE*]...

DESCRIPTION

Print the last 10 lines of each *FILE* to standard output. With more than one *FILE*, precede each with a header giving the file name. With no *FILE*, or when *FILE* is –, read standard input.

Mandatory arguments to long options are mandatory for short options too.

--retry

keep trying to open a file even if it is inaccessible when tail starts or if it becomes inaccessible later; useful when following by name, i.e., with **--follow=name**

-c, --bytes=N

output the last N bytes

-f, --follow[={name|descriptor}]

output appended data as the file grows; **-f**, **--follow**, and **--follow=descriptor** are equivalent

-F same as **--follow=name --retry****-n, --lines=N**

output the last N lines, instead of the last 10

--max-unchanged-stats=N

with **--follow=name**, reopen a *FILE* which has not changed size after N (default 5) iterations to see if it has been unlinked or renamed (this is the usual case of rotated log files)

--pid=PID

with **-f**, terminate after process ID, PID dies

-q, --quiet, --silent

never output headers giving file names

-s, --sleep-interval=S

with **-f**, sleep for approximately S seconds (default 1.0) between iterations.

-v, --verbose

always output headers giving file names

--help display this help and exit**--version**

output version information and exit

If the first character of N (the number of bytes or lines) is a '+', print beginning with the Nth item from the start of each file, otherwise, print the last N items in the file. N may have a multiplier suffix: b 512, k 1024, m 1024*1024.

With **--follow** (**-f**), tail defaults to following the file descriptor, which means that even if a tail'ed file is renamed, tail will continue to track its end. This default behavior is not desirable when you really want to track the actual name of the file, not the file descriptor (e.g., log rotation). Use **--follow=name** in that case. That causes tail to track the named file by reopening it periodically to see if it has been removed and recreated by some other program.

AUTHOR

Written by Paul Rubin, David MacKenzie, Ian Lance Taylor, and Jim Meyering.

REPORTING BUGS

Report bugs to <bug-coreutils@gnu.org>.

COPYRIGHT

Copyright © 2004 Free Software Foundation, Inc.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

SEE ALSO

The full documentation for **tail** is maintained as a Texinfo manual. If the **info** and **tail** programs are properly installed at your site, the command

info tail

should give you access to the complete manual.

NAME

tee – read from standard input and write to standard output and files

SYNOPSIS

tee [*OPTION*]... [*FILE*]...

DESCRIPTION

Copy standard input to each *FILE*, and also to standard output.

-a, --append

append to the given *FILE*s, do not overwrite

-i, --ignore-interrupts

ignore interrupt signals

--help display this help and exit

--version

output version information and exit

If a *FILE* is –, copy again to standard output.

AUTHOR

Written by Mike Parker, Richard M. Stallman, and David MacKenzie.

REPORTING BUGS

Report bugs to <bug-coreutils@gnu.org>.

COPYRIGHT

Copyright © 2004 Free Software Foundation, Inc.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

SEE ALSO

The full documentation for **tee** is maintained as a Texinfo manual. If the **info** and **tee** programs are properly installed at your site, the command

info tee

should give you access to the complete manual.

NAME

test – check file types and compare values

SYNOPSIS

test *EXPRESSION*

test

[*EXPRESSION*]

[]

[*OPTION*

DESCRIPTION

Exit with the status determined by *EXPRESSION*.

—help display this help and exit

—version

output version information and exit

An omitted *EXPRESSION* defaults to false. Otherwise, *EXPRESSION* is true or false and sets exit status. It is one of:

(*EXPRESSION*)

EXPRESSION is true

! *EXPRESSION*

EXPRESSION is false

EXPRESSION1 **–a** *EXPRESSION2*

both *EXPRESSION1* and *EXPRESSION2* are true

EXPRESSION1 **–o** *EXPRESSION2*

either *EXPRESSION1* or *EXPRESSION2* is true

–n *STRING*

the length of *STRING* is nonzero

STRING

equivalent to **–n** *STRING*

–z *STRING*

the length of *STRING* is zero

STRING1 = *STRING2*

the strings are equal

STRING1 != *STRING2*

the strings are not equal

INTEGER1 **–eq** *INTEGER2*

INTEGER1 is equal to *INTEGER2*

INTEGER1 **–ge** *INTEGER2*

INTEGER1 is greater than or equal to *INTEGER2*

INTEGER1 **–gt** *INTEGER2*

INTEGER1 is greater than *INTEGER2*

INTEGER1 **–le** *INTEGER2*

INTEGER1 is less than or equal to *INTEGER2*

INTEGER1 **–lt** *INTEGER2*

INTEGER1 is less than *INTEGER2*

INTEGER1 **–ne** *INTEGER2*

INTEGER1 is not equal to *INTEGER2*

FILE1 **–ef** *FILE2*

FILE1 and *FILE2* have the same device and inode numbers

FILE1 -nt FILE2
FILE1 is newer (modification date) than FILE2

FILE1 -ot FILE2
FILE1 is older than FILE2

-b FILE
FILE exists and is block special

-c FILE
FILE exists and is character special

-d FILE
FILE exists and is a directory

-e FILE
FILE exists

-f FILE
FILE exists and is a regular file

-g FILE
FILE exists and is set-group-ID

-G FILE
FILE exists and is owned by the effective group ID

-h FILE
FILE exists and is a symbolic link (same as **-L**)

-k FILE
FILE exists and has its sticky bit set

-L FILE
FILE exists and is a symbolic link (same as **-h**)

-O FILE
FILE exists and is owned by the effective user ID

-p FILE
FILE exists and is a named pipe

-r FILE
FILE exists and read permission is granted

-s FILE
FILE exists and has a size greater than zero

-S FILE
FILE exists and is a socket

-t FD file descriptor FD is opened on a terminal

-u FILE
FILE exists and its set-user-ID bit is set

-w FILE
FILE exists and write permission is granted

-x FILE
FILE exists and execute (or search) permission is granted

Except for **-h** and **-L**, all FILE-related tests dereference symbolic links. Beware that parentheses need to be escaped (e.g., by backslashes) for shells. INTEGER may also be **-I STRING**, which evaluates to the length of STRING.

AUTHOR

Written by Kevin Braunsdorf and Matthew Bradburn.

REPORTING BUGS

Report bugs to <bug-coreutils@gnu.org>.

COPYRIGHT

Copyright © 2004 Free Software Foundation, Inc.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

SEE ALSO

The full documentation for **test** is maintained as a Texinfo manual. If the **info** and **test** programs are properly installed at your site, the command

info test

should give you access to the complete manual.

NAME

touch – change file timestamps

SYNOPSIS

touch [*OPTION*]... *FILE*...

DESCRIPTION

Update the access and modification times of each *FILE* to the current time.

Mandatory arguments to long options are mandatory for short options too.

-a change only the access time

-c, --no-create
do not create any files

-d, --date=STRING
parse *STRING* and use it instead of current time

-f (ignored)

-m change only the modification time

-r, --reference=FILE
use this file's times instead of current time

-t STAMP
use [[CC]YY]MMDDhhmm[.ss] instead of current time

--time=WORD
change the specified time: *WORD* is access, atime, or use: equivalent to **-a** *WORD* is modify
or mtime: equivalent to **-m**

--help display this help and exit

--version
output version information and exit

Note that the **-d** and **-t** options accept different time–date formats.

AUTHOR

Written by Paul Rubin, Arnold Robbins, Jim Kingdon, David MacKenzie, and Randy Smith.

REPORTING BUGS

Report bugs to <bug-coreutils@gnu.org>.

COPYRIGHT

Copyright © 2004 Free Software Foundation, Inc.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

SEE ALSO

The full documentation for **touch** is maintained as a Texinfo manual. If the **info** and **touch** programs are properly installed at your site, the command

info touch

should give you access to the complete manual.

NAME

tr – translate or delete characters

SYNOPSIS

tr [*OPTION*]... *SET1* [*SET2*]

DESCRIPTION

Translate, squeeze, and/or delete characters from standard input, writing to standard output.

-c, -C, --complement

first complement SET1

-d, --delete

delete characters in SET1, do not translate

-s, --squeeze-repeats

replace each input sequence of a repeated character that is listed in SET1 with a single occurrence of that character

-t, --truncate-set1

first truncate SET1 to length of SET2

--help display this help and exit

--version

output version information and exit

SETs are specified as strings of characters. Most represent themselves. Interpreted sequences are:

\NNN character with octal value NNN (1 to 3 octal digits)

**** backslash

\a audible BEL

\b backspace

\f form feed

\n new line

\r return

\t horizontal tab

\v vertical tab

CHAR1–CHAR2

all characters from CHAR1 to CHAR2 in ascending order

[CHAR*]

in SET2, copies of CHAR until length of SET1

[CHAR*REPEAT]

REPEAT copies of CHAR, REPEAT octal if starting with 0

[:alnum:]

all letters and digits

[:alpha:]

all letters

[:blank:]

all horizontal whitespace

[:cntrl:] all control characters

[:digit:] all digits

[:graph:]

all printable characters, not including space

`[:lower:]`
all lower case letters

`[:print:]`
all printable characters, including space

`[:punct:]`
all punctuation characters

`[:space:]`
all horizontal or vertical whitespace

`[:upper:]`
all upper case letters

`[:xdigit:]`
all hexadecimal digits

`[=CHAR=]`
all characters which are equivalent to CHAR

Translation occurs if `-d` is not given and both SET1 and SET2 appear. `-t` may be used only when translating. SET2 is extended to length of SET1 by repeating its last character as necessary. Excess characters of SET2 are ignored. Only `[:lower:]` and `[:upper:]` are guaranteed to expand in ascending order; used in SET2 while translating, they may only be used in pairs to specify case conversion. `-s` uses SET1 if not translating nor deleting; else squeezing uses SET2 and occurs after translation or deletion.

AUTHOR

Written by Jim Meyering.

REPORTING BUGS

Report bugs to <bug-coreutils@gnu.org>.

COPYRIGHT

Copyright © 2004 Free Software Foundation, Inc.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

SEE ALSO

The full documentation for **tr** is maintained as a Texinfo manual. If the **info** and **tr** programs are properly installed at your site, the command

info tr

should give you access to the complete manual.

NAME

true – do nothing, successfully

SYNOPSIS

true [*ignored command line arguments*]

true *OPTION*

DESCRIPTION

Exit with a status code indicating success.

These option names may not be abbreviated.

—help display this help and exit

—version

output version information and exit

AUTHOR

Written by Jim Meyering.

REPORTING BUGS

Report bugs to <bug-coreutils@gnu.org>.

COPYRIGHT

Copyright © 2004 Free Software Foundation, Inc.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

SEE ALSO

The full documentation for **true** is maintained as a Texinfo manual. If the **info** and **true** programs are properly installed at your site, the command

info true

should give you access to the complete manual.

NAME

`tsort` – perform topological sort

SYNOPSIS

tsort [*OPTION*] [*FILE*]

DESCRIPTION

Write totally ordered list consistent with the partial ordering in *FILE*. With no *FILE*, or when *FILE* is `–`, read standard input.

—help display this help and exit

—version
output version information and exit

AUTHOR

Written by Mark Kettenis.

REPORTING BUGS

Report bugs to <bug-coreutils@gnu.org>.

COPYRIGHT

Copyright © 2004 Free Software Foundation, Inc.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

SEE ALSO

The full documentation for **tsort** is maintained as a Texinfo manual. If the **info** and **tsort** programs are properly installed at your site, the command

info tsort

should give you access to the complete manual.

NAME

tty – print the file name of the terminal connected to standard input

SYNOPSIS

tty [*OPTION*]...

DESCRIPTION

Print the file name of the terminal connected to standard input.

-s, --silent, --quiet

print nothing, only return an exit status

--help display this help and exit

--version

output version information and exit

AUTHOR

Written by David MacKenzie.

REPORTING BUGS

Report bugs to <bug-coreutils@gnu.org>.

COPYRIGHT

Copyright © 2004 Free Software Foundation, Inc.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

SEE ALSO

The full documentation for **tty** is maintained as a Texinfo manual. If the **info** and **tty** programs are properly installed at your site, the command

info tty

should give you access to the complete manual.

NAME

uname – print system information

SYNOPSIS

uname [*OPTION*]...

DESCRIPTION

Print certain system information. With no *OPTION*, same as **–s**.

–a, --all

print all information, in the following order:

–s, --kernel–name

print the kernel name

–n, --nodename

print the network node hostname

–r, --kernel–release

print the kernel release

–v, --kernel–version

print the kernel version

–m, --machine

print the machine hardware name

–p, --processor

print the processor type

–i, --hardware–platform

print the hardware platform

–o, --operating–system

print the operating system

--help display this help and exit

--version

output version information and exit

AUTHOR

Written by David MacKenzie.

REPORTING BUGS

Report bugs to <bug-coreutils@gnu.org>.

COPYRIGHT

Copyright © 2004 Free Software Foundation, Inc.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

SEE ALSO

The full documentation for **uname** is maintained as a Texinfo manual. If the **info** and **uname** programs are properly installed at your site, the command

info uname

should give you access to the complete manual.

NAME

unexpand – convert spaces to tabs

SYNOPSIS

unexpand [*OPTION*]... [*FILE*]...

DESCRIPTION

Convert blanks in each *FILE* to tabs, writing to standard output. With no *FILE*, or when *FILE* is –, read standard input.

Mandatory arguments to long options are mandatory for short options too.

-a, --all

convert all blanks, instead of just initial blanks

--first-only convert only leading sequences of blanks (overrides **-a**)

-t, --tabs=*N*

have tabs *N* characters apart instead of 8 (enables **-a**)

-t, --tabs=*LIST*

use comma separated *LIST* of tab positions (enables **-a**)

--help display this help and exit

--version

output version information and exit

AUTHOR

Written by David MacKenzie.

REPORTING BUGS

Report bugs to <bug-coreutils@gnu.org>.

COPYRIGHT

Copyright © 2004 Free Software Foundation, Inc.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

SEE ALSO

expand(1)

The full documentation for **unexpand** is maintained as a Texinfo manual. If the **info** and **unexpand** programs are properly installed at your site, the command

info unexpand

should give you access to the complete manual.

NAME

uniq – report or omit repeated lines

SYNOPSIS

uniq [*OPTION*]... [*INPUT* [*OUTPUT*]]

DESCRIPTION

Discard all but one of successive identical lines from INPUT (or standard input), writing to OUTPUT (or standard output).

Mandatory arguments to long options are mandatory for short options too.

-c, --count

prefix lines by the number of occurrences

-d, --repeated

only print duplicate lines

-D, --all-repeated[=*delimit-method*] print all duplicate lines

delimit-method={ none(default),prepend,separate } Delimiting is done with blank lines.

-f, --skip-fields=*N*

avoid comparing the first N fields

-i, --ignore-case

ignore differences in case when comparing

-s, --skip-chars=*N*

avoid comparing the first N characters

-u, --unique

only print unique lines

-w, --check-chars=*N*

compare no more than N characters in lines

--help display this help and exit

--version

output version information and exit

A field is a run of whitespace, then non-whitespace characters. Fields are skipped before chars.

AUTHOR

Written by Richard Stallman and David MacKenzie.

REPORTING BUGS

Report bugs to <bug-coreutils@gnu.org>.

COPYRIGHT

Copyright © 2004 Free Software Foundation, Inc.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

SEE ALSO

The full documentation for **uniq** is maintained as a Texinfo manual. If the **info** and **uniq** programs are properly installed at your site, the command

info uniq

should give you access to the complete manual.

NAME

unlink – call the unlink function to remove the specified file

SYNOPSIS

unlink *FILE*

unlink *OPTION*

DESCRIPTION

Call the unlink function to remove the specified FILE.

—help display this help and exit

—version

output version information and exit

AUTHOR

Written by Michael Stone.

REPORTING BUGS

Report bugs to <bug-coreutils@gnu.org>.

COPYRIGHT

Copyright © 2004 Free Software Foundation, Inc.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

SEE ALSO

The full documentation for **unlink** is maintained as a Texinfo manual. If the **info** and **unlink** programs are properly installed at your site, the command

info unlink

should give you access to the complete manual.

NAME

uptime – tell how long the system has been running

SYNOPSIS

uptime [*OPTION*]... [*FILE*]

DESCRIPTION

Print the current time, the length of time the system has been up, the number of users on the system, and the average number of jobs in the run queue over the last 1, 5 and 15 minutes. If *FILE* is not specified, use /var/run/utmp. /var/log/wtmp as *FILE* is common.

—help display this help and exit

—version

output version information and exit

AUTHOR

Written by Joseph Arceneaux, David MacKenzie, and Kaveh Ghazi.

REPORTING BUGS

Report bugs to <bug-coreutils@gnu.org>.

COPYRIGHT

Copyright © 2004 Free Software Foundation, Inc.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

SEE ALSO

The full documentation for **uptime** is maintained as a Texinfo manual. If the **info** and **uptime** programs are properly installed at your site, the command

info uptime

should give you access to the complete manual.

NAME

users – print the user names of users currently logged in to the current host

SYNOPSIS

users [*OPTION*]... [*FILE*]

DESCRIPTION

Output who is currently logged in according to *FILE*. If *FILE* is not specified, use */var/run/utmp*. */var/log/wtmp* as *FILE* is common.

—help display this help and exit

—version

output version information and exit

AUTHOR

Written by Joseph Arceneaux and David MacKenzie.

REPORTING BUGS

Report bugs to <bug-coreutils@gnu.org>.

COPYRIGHT

Copyright © 2004 Free Software Foundation, Inc.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

SEE ALSO

The full documentation for **users** is maintained as a Texinfo manual. If the **info** and **users** programs are properly installed at your site, the command

info users

should give you access to the complete manual.

NAME

vdir – list directory contents

SYNOPSIS

vdir [*OPTION*]... [*FILE*]...

DESCRIPTION

List information about the **FILES** (the current directory by default). Sort entries alphabetically if none of **-cftuSUX** nor **--sort**.

Mandatory arguments to long options are mandatory for short options too.

-a, --all

do not ignore entries starting with **.**

-A, --almost-all

do not list implied **.** and **..**

--author

with **-l**, print the author of each file

-b, --escape

print octal escapes for nongraphic characters

--block-size=SIZE

use **SIZE**-byte blocks

-B, --ignore-backups

do not list implied entries ending with **~**

-c

with **-lt**: sort by, and show, ctime (time of last modification of file status information) with **-l**: show ctime and sort by name otherwise: sort by ctime

-C

list entries by columns

--color[=WHEN]

control whether color is used to distinguish file types. **WHEN** may be ‘never’, ‘always’, or ‘auto’

-d, --directory

list directory entries instead of contents, and do not dereference symbolic links

-D, --dired

generate output designed for Emacs’ dired mode

-f

do not sort, enable **-aU**, disable **-lst**

-F, --classify

append indicator (one of ***/=@|**) to entries

--format=WORD

across **-x**, commas **-m**, horizontal **-x**, long **-l**, single-column **-l**, verbose **-l**, vertical **-C**

--full-time

like **-l --time-style=full-iso**

-g

like **-l**, but do not list owner

-G, --no-group

like **-l**, but do not list group

-h, --human-readable

with **-l**, print sizes in human readable format (e.g., 1K 234M 2G)

--si

likewise, but use powers of 1000 not 1024

-H, --dereference-command-line

follow symbolic links listed on the command line

--dereference-command-line-symlink-to-dir

follow each command line symbolic link that points to a directory

--hide=PATTERN
do not list implied entries matching shell PATTERN (overridden by **-a** or **-A**)

--indicator-style=WORD append indicator with style WORD to entry names:
none (default), classify (**-F**), file-type (**-p**)

-i, --inode
with **-l**, print the index number of each file

-I, --ignore=PATTERN
do not list implied entries matching shell PATTERN

-k like **--block-size=1K**

-l use a long listing format

-L, --dereference
when showing file information for a symbolic link, show information for the file the link references rather than for the link itself

-m fill width with a comma separated list of entries

-n, --numeric-uid-gid
like **-l**, but list numeric UIDs and GIDs

-N, --literal
print raw entry names (don't treat e.g. control characters specially)

-o like **-l**, but do not list group information

-p, --file-type
append indicator (one of **/=@|**) to entries

-q, --hide-control-chars
print **?** instead of non graphic characters

--show-control-chars
show non graphic characters as-is (default unless program is **'ls'** and output is a terminal)

-Q, --quote-name
enclose entry names in double quotes

--quoting-style=WORD
use quoting style WORD for entry names: literal, locale, shell, shell-always, c, escape

-r, --reverse
reverse order while sorting

-R, --recursive
list subdirectories recursively

-s, --size
with **-l**, print size of each file, in blocks

-S sort by file size

--sort=WORD
extension **-X**, none **-U**, size **-S**, time **-t**, version **-v**, status **-c**, time **-t**, atime **-u**, access **-u**, use **-u**

--time=WORD
with **-l**, show time as WORD instead of modification time: atime, access, use, ctime or status;
use specified time as sort key if **--sort=time**

--time-style=STYLE
with **-l**, show times using style STYLE: full-iso, long-iso, iso, locale, +FORMAT. FORMAT is interpreted like 'date'; if FORMAT is FORMAT1<newline>FORMAT2, FORMAT1 applies to non-recent files and FORMAT2 to recent files; if STYLE is prefixed with 'posix-', STYLE takes effect only outside the POSIX locale

-t sort by modification time

- T, --tabsize=COLS**
assume tab stops at each COLS instead of 8
- u** with **-lt**: sort by, and show, access time with **-l**: show access time and sort by name otherwise: sort by access time
- U** do not sort; list entries in directory order
- v** sort by version
- w, --width=COLS**
assume screen width instead of current value
- x** list entries by lines instead of by columns
- X** sort alphabetically by entry extension
- 1** list one file per line
- help** display this help and exit
- version**
output version information and exit

SIZE may be (or may be an integer optionally followed by) one of following: kB 1000, K 1024, MB 1000*1000, M 1024*1024, and so on for G, T, P, E, Z, Y.

By default, color is not used to distinguish types of files. That is equivalent to using **--color=none**. Using the **--color** option without the optional WHEN argument is equivalent to using **--color=always**. With **--color=auto**, color codes are output only if standard output is connected to a terminal (tty).

Exit status is 0 if OK, 1 if minor problems, 2 if serious trouble.

AUTHOR

Written by Richard Stallman and David MacKenzie.

REPORTING BUGS

Report bugs to <bug-coreutils@gnu.org>.

COPYRIGHT

Copyright © 2004 Free Software Foundation, Inc.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

SEE ALSO

The full documentation for **vdir** is maintained as a Texinfo manual. If the **info** and **vdir** programs are properly installed at your site, the command

info vdir

should give you access to the complete manual.

NAME

wc – print the number of newlines, words, and bytes in files

SYNOPSIS

wc [*OPTION*]... [*FILE*]...

DESCRIPTION

Print newline, word, and byte counts for each *FILE*, and a total line if more than one *FILE* is specified. With no *FILE*, or when *FILE* is –, read standard input.

-c, --bytes

print the byte counts

-m, --chars

print the character counts

-l, --lines

print the newline counts

-L, --max-line-length

print the length of the longest line

-w, --words

print the word counts

--help display this help and exit

--version

output version information and exit

AUTHOR

Written by Paul Rubin and David MacKenzie.

REPORTING BUGS

Report bugs to <bug-coreutils@gnu.org>.

COPYRIGHT

Copyright © 2004 Free Software Foundation, Inc.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

SEE ALSO

The full documentation for **wc** is maintained as a Texinfo manual. If the **info** and **wc** programs are properly installed at your site, the command

info wc

should give you access to the complete manual.

NAME

who – show who is logged on

SYNOPSIS

who [*OPTION*]... [*FILE* / *ARG1* *ARG2*]

DESCRIPTION

-a, --all

same as **-b -d --login -p -r -t -T -u**

-b, --boot

time of last system boot

-d, --dead

print dead processes

-H, --heading

print line of column headings

-i, --idle

add idle time as HOURS:MINUTES, . or old (deprecated, use **-u**)

-l, --login

print system login processes

--lookup

attempt to canonicalize hostnames via DNS

-m

only hostname and user associated with stdin

-p, --process

print active processes spawned by init

-q, --count

all login names and number of users logged on

-r, --runlevel

print current runlevel

-s, --short

print only name, line, and time (default)

-t, --time

print last system clock change

-T, -w, --mesg

add user's message status as +, - or ?

-u, --users

list users logged in

--message

same as **-T**

--writable

same as **-T**

--help display this help and exit

--version

output version information and exit

If *FILE* is not specified, use */var/run/utmp*. */var/log/wtmp* as *FILE* is common. If *ARG1 ARG2* given, **-m** presumed: 'am i' or 'mom likes' are usual.

AUTHOR

Written by Joseph Arceneaux, David MacKenzie, and Michael Stone.

REPORTING BUGS

Report bugs to <bug-coreutils@gnu.org>.

COPYRIGHT

Copyright © 2004 Free Software Foundation, Inc.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

SEE ALSO

The full documentation for **who** is maintained as a Texinfo manual. If the **info** and **who** programs are properly installed at your site, the command

info who

should give you access to the complete manual.

NAME

whoami – print effective userid

SYNOPSIS

whoami [*OPTION*]...

DESCRIPTION

Print the user name associated with the current effective user id. Same as **id -un**.

—help display this help and exit

—version

output version information and exit

AUTHOR

Written by Richard Mlynarik.

REPORTING BUGS

Report bugs to <bug-coreutils@gnu.org>.

COPYRIGHT

Copyright © 2004 Free Software Foundation, Inc.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

SEE ALSO

The full documentation for **whoami** is maintained as a Texinfo manual. If the **info** and **whoami** programs are properly installed at your site, the command

info whoami

should give you access to the complete manual.

NAME

yes – output a string repeatedly until killed

SYNOPSIS

yes [*STRING*]...

yes *OPTION*

DESCRIPTION

Repeatedly output a line with all specified *STRING*(s), or ‘y’.

—help display this help and exit

—version

output version information and exit

AUTHOR

Written by David MacKenzie.

REPORTING BUGS

Report bugs to <bug-coreutils@gnu.org>.

COPYRIGHT

Copyright © 2004 Free Software Foundation, Inc.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

SEE ALSO

The full documentation for **yes** is maintained as a Texinfo manual. If the **info** and **yes** programs are properly installed at your site, the command

info yes

should give you access to the complete manual.

NAME

basename - return non-directory portion of a pathname

SYNOPSIS

basename *string* [*suffix*]

DESCRIPTION

The *string* operand shall be treated as a pathname, as defined in the Base Definitions volume of IEEE Std 1003.1-2001, Section 3.266, Pathname. The string *string* shall be converted to the filename corresponding to the last pathname component in *string* and then the suffix string *suffix*, if present, shall be removed. This shall be done by performing actions equivalent to the following steps in order:

If *string* is a null string, it is unspecified whether the resulting string is '.' or a null string. In either case, skip steps 2 through 6.

If *string* is "/" , it is implementation-defined whether steps 3 to 6 are skipped or processed.

If *string* consists entirely of slash characters, *string* shall be set to a single slash character. In this case, skip steps 4 to 6.

If there are any trailing slash characters in *string*, they shall be removed.

If there are any slash characters remaining in *string*, the prefix of *string* up to and including the last slash character in *string* shall be removed.

If the *suffix* operand is present, is not identical to the characters remaining in *string*, and is identical to a suffix of the characters remaining in *string*, the suffix *suffix* shall be removed from *string*. Otherwise, *string* is not modified by this step. It shall not be considered an error if *suffix* is not found in *string*.

The resulting string shall be written to standard output.

OPTIONS

None.

OPERANDS

The following operands shall be supported:

string A string.

suffix A string.

STDIN

Not used.

INPUT FILES

None.

ENVIRONMENT VARIABLES

The following environment variables shall affect the execution of *basename*:

LANG Provide a default value for the internationalization variables that are unset or null. (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2, Internationalization Variables for the precedence of internationalization variables used to determine the values of locale categories.)

LC_ALL

If set to a non-empty string value, override the values of all the other internationalization variables.

LC_CTYPE

Determine the locale for the interpretation of sequences of bytes of text data as characters (for example, single-byte as opposed to multi-byte characters in arguments).

LC_MESSAGES

Determine the locale that should be used to affect the format and contents of diagnostic messages written to standard error.

basename(P)

basename(P)

NLSPATH

Determine the location of message catalogs for the processing of *LC_MESSAGES*.

ASYNCHRONOUS EVENTS

Default.

STDOUT

The *basename* utility shall write a line to the standard output in the following format:

`"%s\n", <resulting string>`

STDERR

The standard error shall be used only for diagnostic messages.

OUTPUT FILES

None.

EXTENDED DESCRIPTION

None.

EXIT STATUS

The following exit values shall be returned:

- 0 Successful completion.
- >0 An error occurred.

CONSEQUENCES OF ERRORS

Default.

The following sections are informative.

APPLICATION USAGE

The definition of *pathname* specifies implementation-defined behavior for pathnames starting with two slash characters. Therefore, applications shall not arbitrarily add slashes to the beginning of a pathname unless they can ensure that there are more or less than two or are prepared to deal with the implementation-defined consequences.

EXAMPLES

If the string *string* is a valid pathname:

`$(basename "string")`

produces a filename that could be used to open the file named by *string* in the directory returned by:

`$(dirname "string")`

If the string *string* is not a valid pathname, the same algorithm is used, but the result need not be a valid filename. The *basename* utility is not expected to make any judgements about the validity of *string* as a pathname; it just follows the specified algorithm to produce a result string.

The following shell script compiles `/usr/src/cmd/cat.c` and moves the output to a file named `cat` in the current directory when invoked with the argument `/usr/src/cmd/cat` or with the argument `/usr/src/cmd/cat.c`:

```
c99 $(dirname "$1")/$(basename "$1" .c).c
mv a.out $(basename "$1" .c)
```

RATIONALE

The behaviors of *basename* and *dirname* have been coordinated so that when *string* is a valid pathname:

\$(basename "string")

would be a valid filename for the file in the directory:

\$(dirname "string")

This would not work for the early proposal versions of these utilities due to the way it specified handling of trailing slashes.

Since the definition of *pathname* specifies implementation-defined behavior for pathnames starting with two slash characters, this volume of IEEE Std 1003.1-2001 specifies similar implementation-defined behavior for the *basename* and *dirname* utilities.

FUTURE DIRECTIONS

None.

SEE ALSO

Parameters and Variables , *dirname()*

COPYRIGHT

Portions of this text are reprinted and reproduced in electronic form from IEEE Std 1003.1, 2003 Edition, Standard for Information Technology -- Portable Operating System Interface (POSIX), The Open Group Base Specifications Issue 6, Copyright (C) 2001-2003 by the Institute of Electrical and Electronics Engineers, Inc and The Open Group. In the event of any discrepancy between this version and the original IEEE and The Open Group Standard, the original IEEE and The Open Group Standard is the referee document. The original Standard can be obtained online at <http://www.open-group.org/unix/online.html> .

NAME

cat - concatenate and print files

SYNOPSIS

cat [-u][*file* ...]

DESCRIPTION

The *cat* utility shall read files in sequence and shall write their contents to the standard output in the same sequence.

OPTIONS

The *cat* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section 12.2, Utility Syntax Guidelines.

The following option shall be supported:

-u Write bytes from the input file to the standard output without delay as each is read.

OPERANDS

The following operand shall be supported:

file A pathname of an input file. If no *file* operands are specified, the standard input shall be used. If a *file* is '-', the *cat* utility shall read from the standard input at that point in the sequence. The *cat* utility shall not close and reopen standard input when it is referenced in this way, but shall accept multiple occurrences of '-' as a *file* operand.

STDIN

The standard input shall be used only if no *file* operands are specified, or if a *file* operand is '-'. See the INPUT FILES section.

INPUT FILES

The input files can be any file type.

ENVIRONMENT VARIABLES

The following environment variables shall affect the execution of *cat*:

LANG Provide a default value for the internationalization variables that are unset or null. (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2, Internationalization Variables for the precedence of internationalization variables used to determine the values of locale categories.)

LC_ALL

If set to a non-empty string value, override the values of all the other internationalization variables.

LC_CTYPE

Determine the locale for the interpretation of sequences of bytes of text data as characters (for example, single-byte as opposed to multi-byte characters in arguments).

LC_MESSAGES

Determine the locale that should be used to affect the format and contents of diagnostic messages written to standard error.

NLSPATH

Determine the location of message catalogs for the processing of *LC_MESSAGES*.

ASYNCHRONOUS EVENTS

Default.

STDOUT

The standard output shall contain the sequence of bytes read from the input files. Nothing else shall be written to the standard output.

STDERR

The standard error shall be used only for diagnostic messages.

OUTPUT FILES

None.

EXTENDED DESCRIPTION

None.

EXIT STATUS

The following exit values shall be returned:

- 0 All input files were output successfully.
- >0 An error occurred.

CONSEQUENCES OF ERRORS

Default.

The following sections are informative.

APPLICATION USAGE

The **-u** option has value in prototyping non-blocking reads from FIFOs. The intent is to support the following sequence:

```
mkfifo foo
cat -u foo > /dev/tty13 &
cat -u > foo
```

It is unspecified whether standard output is or is not buffered in the default case. This is sometimes of interest when standard output is associated with a terminal, since buffering may delay the output. The presence of the **-u** option guarantees that unbuffered I/O is available. It is implementation-defined whether the *cat* utility buffers output if the **-u** option is not specified. Traditionally, the **-u** option is implemented using the equivalent of the *setvbuf()* function defined in the System Interfaces volume of IEEE Std 1003.1-2001.

EXAMPLES

The following command:

```
cat myfile
```

writes the contents of the file **myfile** to standard output.

The following command:

```
cat doc1 doc2 > doc.all
```

concatenates the files **doc1** and **doc2** and writes the result to **doc.all**.

Because of the shell language mechanism used to perform output redirection, a command such as this:

```
cat doc doc.end > doc
```

causes the original data in **doc** to be lost.

The command:

```
cat start - middle - end > file
```

when standard input is a terminal, gets two arbitrary pieces of input from the terminal with a single invocation of *cat*. Note, however, that if standard input is a regular file, this would be equivalent to the command:

cat start - middle /dev/null end > file

because the entire contents of the file would be consumed by *cat* the first time '-' was used as a *file* operand and an end-of-file condition would be detected immediately when '-' was referenced the second time.

RATIONALE

Historical versions of the *cat* utility include the options **-e**, **-t**, and **-v**, which permit the ends of lines, <tab>s, and invisible characters, respectively, to be rendered visible in the output. The standard developers omitted these options because they provide too fine a degree of control over what is made visible, and similar output can be obtained using a command such as:

sed -n -e 's/\$/\$/ ' -e l pathname

The **-s** option was omitted because it corresponds to different functions in BSD and System V-based systems. The BSD **-s** option to squeeze blank lines can be accomplished by the shell script shown in the following example:

```
sed -n '
# Write non-empty lines.
./ {
    p
    d
}
# Write a single empty line, then look for more empty lines.
/^$/ p
# Get next line, discard the held <newline> (empty line),
# and look for more empty lines.
:Empty
/^$/ {
    N
    s/./
    b Empty
}
# Write the non-empty line before going back to search
# for the first in a set of empty lines.
p

```

The System V **-s** option to silence error messages can be accomplished by redirecting the standard error. Note that the BSD documentation for *cat* uses the term "blank line" to mean the same as the POSIX "empty line": a line consisting only of a <newline>.

The BSD **-n** option was omitted because similar functionality can be obtained from the **-n** option of the *pr* utility.

FUTURE DIRECTIONS

None.

SEE ALSO

more, the System Interfaces volume of IEEE Std 1003.1-2001, *setvbuf()*

COPYRIGHT

Portions of this text are reprinted and reproduced in electronic form from IEEE Std 1003.1, 2003 Edition, Standard for Information Technology -- Portable Operating System Interface (POSIX), The Open Group Base Specifications Issue 6, Copyright (C) 2001-2003 by the Institute of Electrical and Electronics Engineers, Inc and The Open Group. In the event of any discrepancy between this version and the original IEEE and The Open Group Standard, the original IEEE and The Open Group Standard is the referee document. The original Standard can be obtained online at <http://www.open-group.org/unix/online.html>.

NAME

chgrp - change the file group ownership

SYNOPSIS

chgrp [-hR] *group file ...*

chgrp -R [-H | -L | -P] *group file ...*

DESCRIPTION

The *chgrp* utility shall set the group ID of the file named by each *file* operand to the group ID specified by the *group* operand.

For each *file* operand, or, if the **-R** option is used, each file encountered while walking the directory trees specified by the *file* operands, the *chgrp* utility shall perform actions equivalent to the *chown()* function defined in the System Interfaces volume of IEEE Std 1003.1-2001, called with the following arguments:

The *file* operand shall be used as the *path* argument.

The user ID of the file shall be used as the *owner* argument.

The specified group ID shall be used as the *group* argument.

Unless *chgrp* is invoked by a process with appropriate privileges, the set-user-ID and set-group-ID bits of a regular file shall be cleared upon successful completion; the set-user-ID and set-group-ID bits of other file types may be cleared.

OPTIONS

The *chgrp* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section 12.2, Utility Syntax Guidelines.

The following options shall be supported by the implementation:

- h** If the system supports group IDs for symbolic links, for each *file* operand that names a file of type symbolic link, *chgrp* shall attempt to set the group ID of the symbolic link instead of the file referenced by the symbolic link. If the system does not support group IDs for symbolic links, for each *file* operand that names a file of type symbolic link, *chgrp* shall do nothing more with the current file and shall go on to any remaining files.
- H** If the **-R** option is specified and a symbolic link referencing a file of type directory is specified on the command line, *chgrp* shall change the group of the directory referenced by the symbolic link and all files in the file hierarchy below it.
- L** If the **-R** option is specified and a symbolic link referencing a file of type directory is specified on the command line or encountered during the traversal of a file hierarchy, *chgrp* shall change the group of the directory referenced by the symbolic link and all files in the file hierarchy below it.
- P** If the **-R** option is specified and a symbolic link is specified on the command line or encountered during the traversal of a file hierarchy, *chgrp* shall change the group ID of the symbolic link if the system supports this operation. The *chgrp* utility shall not follow the symbolic link to any other part of the file hierarchy.
- R** Recursively change file group IDs. For each *file* operand that names a directory, *chgrp* shall change the group of the directory and all files in the file hierarchy below it. Unless a **-H**, **-L**, or **-P** option is specified, it is unspecified which of these options will be used as the default.

Specifying more than one of the mutually-exclusive options **-H**, **-L**, and **-P** shall not be considered an error. The last option specified shall determine the behavior of the utility.

OPERANDS

The following operands shall be supported:

group A group name from the group database or a numeric group ID. Either specifies a group ID to be given to each file named by one of the *file* operands. If a numeric *group* operand exists in

the group database as a group name, the group ID number associated with that group name is used as the group ID.

file A pathname of a file whose group ID is to be modified.

STDIN

Not used.

INPUT FILES

None.

ENVIRONMENT VARIABLES

The following environment variables shall affect the execution of *chgrp*:

LANG Provide a default value for the internationalization variables that are unset or null. (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2, Internationalization Variables for the precedence of internationalization variables used to determine the values of locale categories.)

LC_ALL

If set to a non-empty string value, override the values of all the other internationalization variables.

LC_CTYPE

Determine the locale for the interpretation of sequences of bytes of text data as characters (for example, single-byte as opposed to multi-byte characters in arguments).

LC_MESSAGES

Determine the locale that should be used to affect the format and contents of diagnostic messages written to standard error.

NLSPATH

Determine the location of message catalogs for the processing of *LC_MESSAGES*.

ASYNCHRONOUS EVENTS

Default.

STDOUT

Not used.

STDERR

The standard error shall be used only for diagnostic messages.

OUTPUT FILES

None.

EXTENDED DESCRIPTION

None.

EXIT STATUS

The following exit values shall be returned:

- 0 The utility executed successfully and all requested changes were made.
- >0 An error occurred.

CONSEQUENCES OF ERRORS

Default.

The following sections are informative.

APPLICATION USAGE

Only the owner of a file or the user with appropriate privileges may change the owner or group of a file.

Some implementations restrict the use of *chgrp* to a user with appropriate privileges when the *group* specified is not the effective group ID or one of the supplementary group IDs of the calling process.

EXAMPLES

None.

RATIONALE

The System V and BSD versions use different exit status codes. Some implementations used the exit status as a count of the number of errors that occurred; this practice is unworkable since it can overflow the range of valid exit status values. The standard developers chose to mask these by specifying only 0 and >0 as exit values.

The functionality of *chgrp* is described substantially through references to *chown()*. In this way, there is no duplication of effort required for describing the interactions of permissions, multiple groups, and so on.

FUTURE DIRECTIONS

None.

SEE ALSO

chmod() , *chown()* , the System Interfaces volume of IEEE Std 1003.1-2001, *chown()*

COPYRIGHT

Portions of this text are reprinted and reproduced in electronic form from IEEE Std 1003.1, 2003 Edition, Standard for Information Technology -- Portable Operating System Interface (POSIX), The Open Group Base Specifications Issue 6, Copyright (C) 2001-2003 by the Institute of Electrical and Electronics Engineers, Inc and The Open Group. In the event of any discrepancy between this version and the original IEEE and The Open Group Standard, the original IEEE and The Open Group Standard is the referee document. The original Standard can be obtained online at <http://www.open-group.org/unix/online.html> .

NAME

chmod - change the file modes

SYNOPSIS

chmod [-R] *mode file ...*

DESCRIPTION

The *chmod* utility shall change any or all of the file mode bits of the file named by each *file* operand in the way specified by the *mode* operand.

It is implementation-defined whether and how the *chmod* utility affects any alternate or additional file access control mechanism (see the Base Definitions volume of IEEE Std 1003.1-2001, Section 4.4, File Access Permissions) being used for the specified file.

Only a process whose effective user ID matches the user ID of the file, or a process with the appropriate privileges, shall be permitted to change the file mode bits of a file.

OPTIONS

The *chmod* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section 12.2, Utility Syntax Guidelines.

The following option shall be supported:

- R** Recursively change file mode bits. For each *file* operand that names a directory, *chmod* shall change the file mode bits of the directory and all files in the file hierarchy below it.

OPERANDS

The following operands shall be supported:

mode Represents the change to be made to the file mode bits of each file named by one of the *file* operands; see the EXTENDED DESCRIPTION section.

file A pathname of a file whose file mode bits shall be modified.

STDIN

Not used.

INPUT FILES

None.

ENVIRONMENT VARIABLES

The following environment variables shall affect the execution of *chmod*:

LANG Provide a default value for the internationalization variables that are unset or null. (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2, Internationalization Variables for the precedence of internationalization variables used to determine the values of locale categories.)

LC_ALL

If set to a non-empty string value, override the values of all the other internationalization variables.

LC_CTYPE

Determine the locale for the interpretation of sequences of bytes of text data as characters (for example, single-byte as opposed to multi-byte characters in arguments).

LC_MESSAGES

Determine the locale that should be used to affect the format and contents of diagnostic messages written to standard error.

NLSPATH

Determine the location of message catalogs for the processing of *LC_MESSAGES*.

ASYNCHRONOUS EVENTS

Default.

STDOUT

Not used.

STDERR

The standard error shall be used only for diagnostic messages.

OUTPUT FILES

None.

EXTENDED DESCRIPTION

The *mode* operand shall be either a *symbolic_mode* expression or a non-negative octal integer. The *symbolic_mode* form is described by the grammar later in this section.

Each **clause** shall specify an operation to be performed on the current file mode bits of each *file*. The operations shall be performed on each *file* in the order in which the **clauses** are specified.

The **who** symbols **u**, **g**, and **o** shall specify the *user*, *group*, and *other* parts of the file mode bits, respectively. A **who** consisting of the symbol **a** shall be equivalent to **ugo**.

The **perm** symbols **r**, **w**, and **x** represent the *read*, *write*, and *execute/ search* portions of file mode bits, respectively. The **perm** symbol **s** shall represent the *set-user-ID-on-execution* (when **who** contains or implies **u**) and *set-group-ID-on-execution* (when **who** contains or implies **g**) bits.

The **perm** symbol **X** shall represent the execute/search portion of the file mode bits if the file is a directory or if the current (unmodified) file mode bits have at least one of the execute bits (S_IXUSR, S_IXGRP, or S_IXOTH) set. It shall be ignored if the file is not a directory and none of the execute bits are set in the current file mode bits.

The **permcop** symbols **u**, **g**, and **o** shall represent the current permissions associated with the user, group, and other parts of the file mode bits, respectively. For the remainder of this section, **perm** refers to the non-terminals **perm** and **permcop** in the grammar.

If multiple **actionlists** are grouped with a single **wholist** in the grammar, each **actionlist** shall be applied in the order specified with that **wholist**. The *op* symbols shall represent the operation performed, as follows:

+ If **perm** is not specified, the '+' operation shall not change the file mode bits.

If **who** is not specified, the file mode bits represented by **perm** for the owner, group, and other permissions, except for those with corresponding bits in the file mode creation mask of the invoking process, shall be set.

Otherwise, the file mode bits represented by the specified **who** and **perm** values shall be set.

- If **perm** is not specified, the '-' operation shall not change the file mode bits.

If **who** is not specified, the file mode bits represented by **perm** for the owner, group, and other permissions, except for those with corresponding bits in the file mode creation mask of the invoking process, shall be cleared.

Otherwise, the file mode bits represented by the specified **who** and **perm** values shall be cleared.

= Clear the file mode bits specified by the **who** value, or, if no **who** value is specified, all of the file mode bits specified in this volume of IEEE Std 1003.1-2001.

If **perm** is not specified, the '=' operation shall make no further modifications to the file mode bits.

If **who** is not specified, the file mode bits represented by **perm** for the owner, group, and other permissions, except for those with corresponding bits in the file mode creation mask of the invoking process, shall be set.

Otherwise, the file mode bits represented by the specified **who** and **perm** values shall be set.

When using the symbolic mode form on a regular file, it is implementation-defined whether or not:

Requests to set the set-user-ID-on-execution or set-group-ID-on-execution bit when all execute bits are currently clear and none are being set are ignored.

Requests to clear all execute bits also clear the set-user-ID-on-execution and set-group-ID-on-execution bits.

Requests to clear the set-user-ID-on-execution or set-group-ID-on-execution bits when all execute bits are currently clear are ignored. However, if the command *ls -l file* writes an *s* in the position indicating that the set-user-ID-on-execution or set-group-ID-on-execution is set, the commands *chmod u-s file* or *chmod g-s file*, respectively, shall not be ignored.

When using the symbolic mode form on other file types, it is implementation-defined whether or not requests to set or clear the set-user-ID-on-execution or set-group-ID-on-execution bits are honored.

If the **who** symbol **o** is used in conjunction with the **perm** symbol **s** with no other **who** symbols being specified, the set-user-ID-on-execution and set-group-ID-on-execution bits shall not be modified. It shall not be an error to specify the **who** symbol **o** in conjunction with the **perm** symbol **s**.

The **perm** symbol **t** shall specify the S_ISVTX bit. When used with a file of type directory, it can be used with the **who** symbol **a**, or with no **who** symbol. It shall not be an error to specify a **who** symbol of **u**, **g**, or **o** in conjunction with the **perm** symbol **t**, but the meaning of these combinations is unspecified. The effect when using the **perm** symbol **t** with any file type other than directory is unspecified.

For an octal integer *mode* operand, the file mode bits shall be set absolutely.

For each bit set in the octal number, the corresponding file permission bit shown in the following table shall be set; all other file permission bits shall be cleared. For regular files, for each bit set in the octal number corresponding to the set-user-ID-on-execution or the set-group-ID-on-execution, bits shown in the following table shall be set; if these bits are not set in the octal number, they are cleared. For other file types, it is implementation-defined whether or not requests to set or clear the set-user-ID-on-execution or set-group-ID-on-execution bits are honored.

Octal	Mode Bit	Octal	Mode Bit	Octal	Mode Bit	Octal	Mode Bit
4000	S_ISUID	0400	S_IRUSR	0040	S_IRGRP	0004	S_IROTH
2000	S_ISGID	0200	S_IWUSR	0020	S_IWGRP	0002	S_IWOTH
1000	S_ISVTX	0100	S_IXUSR	0010	S_IXGRP	0001	S_IXOTH

When bits are set in the octal number other than those listed in the table above, the behavior is unspecified.

Grammar for chmod

The grammar and lexical conventions in this section describe the syntax for the *symbolic_mode* operand. The general conventions for this style of grammar are described in *Grammar Conventions*. A valid *symbolic_mode* can be represented as the non-terminal symbol *symbolic_mode* in the grammar. This formal syntax shall take precedence over the preceding text syntax description.

The lexical processing is based entirely on single characters. Implementations need not allow <blank>s within the single argument being processed.

%start *symbolic_mode*

%%

symbolic_mode : clause
 | *symbolic_mode* ',' clause
 ;

clause : actionlist
 | **wholist** actionlist
 ;

wholist : **who**
 | **wholist** **who**
 ;

who : 'u' | 'g' | 'o' | 'a'


```

;

actionlist    : action
               | actionlist action
               ;

action        : op
               | op permlist
               | op permcopy
               ;

permcopy      : 'u' | 'g' | 'o'
               ;

op            : '+' | '-' | '='
               ;

permlist      : perm
               | perm permlist
               ;

perm          : 'r' | 'w' | 'x' | 'X' | 's' | 't'
               ;

```

EXIT STATUS

The following exit values shall be returned:

- 0 The utility executed successfully and all requested changes were made.
- >0 An error occurred.

CONSEQUENCES OF ERRORS

Default.

The following sections are informative.

APPLICATION USAGE

Some implementations of the *chmod* utility change the mode of a directory before the files in the directory when performing a recursive (**-R** option) change; others change the directory mode after the files in the directory. If an application tries to remove read or search permission for a file hierarchy, the removal attempt fails if the directory is changed first; on the other hand, trying to re-enable permissions to a restricted hierarchy fails if directories are changed last. Users should not try to make a hierarchy inaccessible to themselves.

Some implementations of *chmod* never used the process' *umask* when changing modes; systems conformant with this volume of IEEE Std 1003.1-2001 do so when **who** is not specified. Note the difference between:

```
chmod a-w file
```

which removes all write permissions, and:

```
chmod -- -w file
```

which removes write permissions that would be allowed if **file** was created with the same *umask*.

Conforming applications should never assume that they know how the set-user-ID and set-group-ID bits on directories are interpreted.

EXAMPLES

Mode	Results
<i>a+=</i>	Equivalent to <i>a+</i> , <i>a=</i> ; clears all file mode bits.
<i>go+-w</i>	Equivalent to <i>go+</i> , <i>go- w</i> ; clears group and other write bits.
<i>g=o-w</i>	Equivalent to <i>g= o</i> , <i>g- w</i> ; sets group bit to match other bits and then clears group write bit.
<i>g-r+w</i>	Equivalent to <i>g- r</i> , <i>g+ w</i> ; clears group read bit and sets group write bit.
<i>uo=g</i>	Sets owner bits to match group bits and sets other bits to match group bits.

RATIONALE

The functionality of *chmod* is described substantially through references to concepts defined in the System Interfaces volume of IEEE Std 1003.1-2001. In this way, there is less duplication of effort required for describing the interactions of permissions. However, the behavior of this utility is not described in terms of the *chmod()* function from the System Interfaces volume of IEEE Std 1003.1-2001 because that specification requires certain side effects upon alternate file access control mechanisms that might not be appropriate, depending on the implementation.

Implementations that support mandatory file and record locking as specified by the 1984 /usr/group standard historically used the combination of set-group-ID bit set and group execute bit clear to indicate mandatory locking. This condition is usually set or cleared with the symbolic mode **perm** symbol **l** instead of the **perm** symbols **s** and **x** so that the mandatory locking mode is not changed without explicit indication that that was what the user intended. Therefore, the details on how the implementation treats these conditions must be defined in the documentation. This volume of IEEE Std 1003.1-2001 does not require mandatory locking (nor does the System Interfaces volume of IEEE Std 1003.1-2001), but does allow it as an extension. However, this volume of IEEE Std 1003.1-2001 does require that the *ls* and *chmod* utilities work consistently in this area. If *ls -l file* indicates that the set-group-ID bit is set, *chmod g-s file* must clear it (assuming appropriate privileges exist to change modes).

The System V and BSD versions use different exit status codes. Some implementations used the exit status as a count of the number of errors that occurred; this practice is unworkable since it can overflow the range of valid exit status values. This problem is avoided here by specifying only 0 and >0 as exit values.

The System Interfaces volume of IEEE Std 1003.1-2001 indicates that implementation-defined restrictions may cause the **S_ISUID** and **S_ISGID** bits to be ignored. This volume of IEEE Std 1003.1-2001 allows the *chmod* utility to choose to modify these bits before calling *chmod()* (or some function providing equivalent capabilities) for non-regular files. Among other things, this allows implementations that use the set-user-ID and set-group-ID bits on directories to enable extended features to handle these extensions in an intelligent manner.

The **X perm** symbol was adopted from BSD-based systems because it provides commonly desired functionality when doing recursive (**-R** option) modifications. Similar functionality is not provided by the *find* utility. Historical BSD versions of *chmod*, however, only supported **X** with *op+*; it has been extended in this volume of IEEE Std 1003.1-2001 because it is also useful with *op=*. (It has also been added for *op-* even though it duplicates **x**, in this case, because it is intuitive and easier to explain.)

The grammar was extended with the *permcop*y non-terminal to allow historical-practice forms of symbolic modes like **o= u -g** (that is, set the "other" permissions to the permissions of "owner" minus the permissions of "group").

FUTURE DIRECTIONS

None.

SEE ALSO

ls, *umask*, the System Interfaces volume of IEEE Std 1003.1-2001, *chmod()*

COPYRIGHT

Portions of this text are reprinted and reproduced in electronic form from IEEE Std 1003.1, 2003 Edition, Standard for Information Technology -- Portable Operating System Interface (POSIX), The Open Group Base Specifications Issue 6, Copyright (C) 2001-2003 by the Institute of Electrical and

Electronics Engineers, Inc and The Open Group. In the event of any discrepancy between this version and the original IEEE and The Open Group Standard, the original IEEE and The Open Group Standard is the referee document. The original Standard can be obtained online at <http://www.open-group.org/unix/online.html> .

NAME

chown - change the file ownership

SYNOPSIS

chown [-hR] owner[:group] file ...

chown -R [-H | -L | -P] owner[:group] file ...

DESCRIPTION

The *chown* utility shall set the user ID of the file named by each *file* operand to the user ID specified by the *owner* operand.

For each *file* operand, or, if the **-R** option is used, each file encountered while walking the directory trees specified by the *file* operands, the *chown* utility shall perform actions equivalent to the *chown()* function defined in the System Interfaces volume of IEEE Std 1003.1-2001, called with the following arguments:

The *file* operand shall be used as the *path* argument.

The user ID indicated by the *owner* portion of the first operand shall be used as the *owner* argument.

If the *group* portion of the first operand is given, the group ID indicated by it shall be used as the *group* argument; otherwise, the group ownership shall not be changed.

Unless *chown* is invoked by a process with appropriate privileges, the set-user-ID and set-group-ID bits of a regular file shall be cleared upon successful completion; the set-user-ID and set-group-ID bits of other file types may be cleared.

OPTIONS

The *chown* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section 12.2, Utility Syntax Guidelines.

The following options shall be supported by the implementation:

- h** If the system supports user IDs for symbolic links, for each *file* operand that names a file of type symbolic link, *chown* shall attempt to set the user ID of the symbolic link. If the system supports group IDs for symbolic links, and a group ID was specified, for each *file* operand that names a file of type symbolic link, *chown* shall attempt to set the group ID of the symbolic link. If the system does not support user or group IDs for symbolic links, for each *file* operand that names a file of type symbolic link, *chown* shall do nothing more with the current file and shall go on to any remaining files.
- H** If the **-R** option is specified and a symbolic link referencing a file of type directory is specified on the command line, *chown* shall change the user ID (and group ID, if specified) of the directory referenced by the symbolic link and all files in the file hierarchy below it.
- L** If the **-R** option is specified and a symbolic link referencing a file of type directory is specified on the command line or encountered during the traversal of a file hierarchy, *chown* shall change the user ID (and group ID, if specified) of the directory referenced by the symbolic link and all files in the file hierarchy below it.
- P** If the **-R** option is specified and a symbolic link is specified on the command line or encountered during the traversal of a file hierarchy, *chown* shall change the owner ID (and group ID, if specified) of the symbolic link if the system supports this operation. The *chown* utility shall not follow the symbolic link to any other part of the file hierarchy.
- R** Recursively change file user and group IDs. For each *file* operand that names a directory, *chown* shall change the user ID (and group ID, if specified) of the directory and all files in the file hierarchy below it. Unless a **-H**, **-L**, or **-P** option is specified, it is unspecified which of these options will be used as the default.

Specifying more than one of the mutually-exclusive options **-H**, **-L**, and **-P** shall not be considered an error. The last option specified shall determine the behavior of the utility.

OPERANDS

The following operands shall be supported:

owner[:group]

A user ID and optional group ID to be assigned to *file*. The *owner* portion of this operand shall be a user name from the user database or a numeric user ID. Either specifies a user ID which shall be given to each file named by one of the *file* operands. If a numeric *owner* operand exists in the user database as a user name, the user ID number associated with that user name shall be used as the user ID. Similarly, if the *group* portion of this operand is present, it shall be a group name from the group database or a numeric group ID. Either specifies a group ID which shall be given to each file. If a numeric group operand exists in the group database as a group name, the group ID number associated with that group name shall be used as the group ID.

file A pathname of a file whose user ID is to be modified.

STDIN

Not used.

INPUT FILES

None.

ENVIRONMENT VARIABLES

The following environment variables shall affect the execution of *chown*:

LANG Provide a default value for the internationalization variables that are unset or null. (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2, Internationalization Variables for the precedence of internationalization variables used to determine the values of locale categories.)

LC_ALL

If set to a non-empty string value, override the values of all the other internationalization variables.

LC_CTYPE

Determine the locale for the interpretation of sequences of bytes of text data as characters (for example, single-byte as opposed to multi-byte characters in arguments).

LC_MESSAGES

Determine the locale that should be used to affect the format and contents of diagnostic messages written to standard error.

NLSPATH

Determine the location of message catalogs for the processing of *LC_MESSAGES*.

ASYNCHRONOUS EVENTS

Default.

STDOUT

Not used.

STDERR

The standard error shall be used only for diagnostic messages.

OUTPUT FILES

None.

EXTENDED DESCRIPTION

None.

EXIT STATUS

The following exit values shall be returned:

- 0 The utility executed successfully and all requested changes were made.
- >0 An error occurred.

CONSEQUENCES OF ERRORS

Default.

The following sections are informative.

APPLICATION USAGE

Only the owner of a file or the user with appropriate privileges may change the owner or group of a file.

Some implementations restrict the use of *chown* to a user with appropriate privileges.

EXAMPLES

None.

RATIONALE

The System V and BSD versions use different exit status codes. Some implementations used the exit status as a count of the number of errors that occurred; this practice is unworkable since it can overflow the range of valid exit status values. These are masked by specifying only 0 and >0 as exit values.

The functionality of *chown* is described substantially through references to functions in the System Interfaces volume of IEEE Std 1003.1-2001. In this way, there is no duplication of effort required for describing the interactions of permissions, multiple groups, and so on.

The 4.3 BSD method of specifying both owner and group was included in this volume of IEEE Std 1003.1-2001 because:

There are cases where the desired end condition could not be achieved using the *chgrp* and *chown* (that only changed the user ID) utilities. (If the current owner is not a member of the desired group and the desired owner is not a member of the current group, the *chown()* function could fail unless both owner and group are changed at the same time.)

Even if they could be changed independently, in cases where both are being changed, there is a 100% performance penalty caused by being forced to invoke both utilities.

The BSD syntax *user[. group]* was changed to *user[: group]* in this volume of IEEE Std 1003.1-2001 because the period is a valid character in login names (as specified by the Base Definitions volume of IEEE Std 1003.1-2001, login names consist of characters in the portable filename character set). The colon character was chosen as the replacement for the period character because it would never be allowed as a character in a user name or group name on historical implementations.

The **-R** option is considered by some observers as an undesirable departure from the historical UNIX system tools approach; since a tool, *find*, already exists to recurse over directories, there seemed to be no good reason to require other tools to have to duplicate that functionality. However, the **-R** option was deemed an important user convenience, is far more efficient than forking a separate process for each element of the directory hierarchy, and is in widespread historical use.

FUTURE DIRECTIONS

None.

SEE ALSO

chmod, *chgrp*, the System Interfaces volume of IEEE Std 1003.1-2001, *chown()*

COPYRIGHT

Portions of this text are reprinted and reproduced in electronic form from IEEE Std 1003.1, 2003 Edition, Standard for Information Technology -- Portable Operating System Interface (POSIX), The Open Group Base Specifications Issue 6, Copyright (C) 2001-2003 by the Institute of Electrical and Electronics Engineers, Inc and The Open Group. In the event of any discrepancy between this version and the original IEEE and The Open Group Standard, the original IEEE and The Open Group Standard is the referee document. The original Standard can be obtained online at <http://www.open-group.org/unix/online.html>.

NAME

cksum - write file checksums and sizes

SYNOPSIS

cksum [*file ...*]

DESCRIPTION

The *cksum* utility shall calculate and write to standard output a cyclic redundancy check (CRC) for each input file, and also write to standard output the number of octets in each file. The CRC used is based on the polynomial used for CRC error checking in the ISO/IEC 8802-3:1996 standard (Ethernet).

The encoding for the CRC checksum is defined by the generating polynomial:

$$G(x)=x+x+x+x+x+x+x+x+x+x+x+x+x+x+1$$

Mathematically, the CRC value corresponding to a given file shall be defined by the following procedure:

The n bits to be evaluated are considered to be the coefficients of a mod 2 polynomial $M(x)$ of degree $n-1$. These n bits are the bits from the file, with the most significant bit being the most significant bit of the first octet of the file and the last bit being the least significant bit of the last octet, padded with zero bits (if necessary) to achieve an integral number of octets, followed by one or more octets representing the length of the file as a binary value, least significant octet first. The smallest number of octets capable of representing this integer shall be used.

$M(x)$ is multiplied by x (that is, shifted left 32 bits) and divided by $G(x)$ using mod 2 division, producing a remainder $R(x)$ of degree ≤ 31 .

The coefficients of $R(x)$ are considered to be a 32-bit sequence.

The bit sequence is complemented and the result is the CRC.

OPTIONS

None.

OPERANDS

The following operand shall be supported:

file A pathname of a file to be checked. If no *file* operands are specified, the standard input shall be used.

STDIN

The standard input shall be used only if no *file* operands are specified. See the INPUT FILES section.

INPUT FILES

The input files can be any file type.

ENVIRONMENT VARIABLES

The following environment variables shall affect the execution of *cksum*:

LANG Provide a default value for the internationalization variables that are unset or null. (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2, Internationalization Variables for the precedence of internationalization variables used to determine the values of locale categories.)

LC_ALL

If set to a non-empty string value, override the values of all the other internationalization variables.

LC CTYPE

Determine the locale for the interpretation of sequences of bytes of text data as characters (for example, single-byte as opposed to multi-byte characters in arguments).

LC_MESSAGES

Determine the locale that should be used to affect the format and contents of diagnostic messages written to standard error.

NLSPATH

Determine the location of message catalogs for the processing of *LC_MESSAGES*.

ASYNCHRONOUS EVENTS

Default.

STDOUT

For each file processed successfully, the *cksum* utility shall write in the following format:

```
"%u %d %s\n", <checksum>, <# of octets>, <pathname>
```

If no *file* operand was specified, the pathname and its leading <space> shall be omitted.

STDERR

The standard error shall be used only for diagnostic messages.

OUTPUT FILES

None.

EXTENDED DESCRIPTION

None.

EXIT STATUS

The following exit values shall be returned:

- 0 All files were processed successfully.
- >0 An error occurred.

CONSEQUENCES OF ERRORS

Default.

The following sections are informative.

APPLICATION USAGE

The *cksum* utility is typically used to quickly compare a suspect file against a trusted version of the same, such as to ensure that files transmitted over noisy media arrive intact. However, this comparison cannot be considered cryptographically secure. The chances of a damaged file producing the same CRC as the original are small; deliberate deception is difficult, but probably not impossible.

Although input files to *cksum* can be any type, the results need not be what would be expected on character special device files or on file types not described by the System Interfaces volume of IEEE Std 1003.1-2001. Since this volume of IEEE Std 1003.1-2001 does not specify the block size used when doing input, checksums of character special files need not process all of the data in those files.

The algorithm is expressed in terms of a bitstream divided into octets. If a file is transmitted between two systems and undergoes any data transformation (such as changing little-endian byte ordering to big-endian), identical CRC values cannot be expected. Implementations performing such transformations may extend *cksum* to handle such situations.

EXAMPLES

None.

RATIONALE

The following C-language program can be used as a model to describe the algorithm. It assumes that a **char** is one octet. It also assumes that the entire file is available for one pass through the function. This was done for simplicity in demonstrating the algorithm, rather than as an implementation model.

```
static unsigned long crctab[] = {
    0x00000000,
```



```

0x04c11db7, 0x09823b6e, 0x0d4326d9, 0x130476dc, 0x17c56b6b,
0x1a864db2, 0x1e475005, 0x2608edb8, 0x22c9f00f, 0x2f8ad6d6,
0x2b4bcb61, 0x350c9b64, 0x31cd86d3, 0x3c8ea00a, 0x384fbbdb,
0x4c11db70, 0x48d0c6c7, 0x4593e01e, 0x4152fda9, 0x5f15adac,
0x5bd4b01b, 0x569796c2, 0x52568b75, 0x6a1936c8, 0x6ed82b7f,
0x639b0da6, 0x675a1011, 0x791d4014, 0x7ddc5da3, 0x709f7b7a,
0x745e66cd, 0x9823b6e0, 0x9ce2ab57, 0x91a18d8e, 0x95609039,
0x8b27c03c, 0x8fe6dd8b, 0x82a5fb52, 0x8664e6e5, 0xbe2b5b58,
0xbaea46ef, 0xb7a96036, 0xb3687d81, 0xad2f2d84, 0xa9ee3033,
0xa4ad16ea, 0xa06c0b5d, 0xd4326d90, 0xd0f37027, 0xddb056fe,
0xd9714b49, 0xc7361b4c, 0xc3f706fb, 0xceb42022, 0xca753d95,
0xf23a8028, 0xf6fb9d9f, 0xfbb8bb46, 0xff79a6f1, 0xe13ef6f4,
0xe5ffe43, 0xe8bccd9a, 0xec7dd02d, 0x34867077, 0x30476dc0,
0x3d044b19, 0x39c556ae, 0x278206ab, 0x23431b1c, 0x2e003dc5,
0x2ac12072, 0x128e9dcf, 0x164f8078, 0x1b0ca6a1, 0x1fcdabb16,
0x018aeb13, 0x054bf6a4, 0x0808d07d, 0x0cc9cdca, 0x7897ab07,
0x7c56b6b0, 0x71159069, 0x75d48dde, 0x6b93dddb, 0x6f52c06c,
0x6211e6b5, 0x66d0fb02, 0x5e9f46bf, 0x5a5e5b08, 0x571d7dd1,
0x53dc6066, 0x4d9b3063, 0x495a2dd4, 0x44190b0d, 0x40d816ba,
0xaca5c697, 0xa864db20, 0xa527fd9, 0xa1e6e04e, 0xbfa1b04b,
0xbb60adfc, 0xb6238b25, 0xb2e29692, 0x8aad2b2f, 0x8e6c3698,
0x832f1041, 0x87ee0df6, 0x99a95df3, 0x9d684044, 0x902b669d,
0x94ea7b2a, 0xe0b41de7, 0xe4750050, 0xe9362689, 0xedf73b3e,
0xf3b06b3b, 0xf771768c, 0xfa325055, 0xfef34de2, 0xc6bcf05f,
0xc27dede8, 0xcf3ecb31, 0xcbffd686, 0xd5b88683, 0xd1799b34,
0xdc3abded, 0xd8fba05a, 0x690ce0ee, 0x6dcd9d59, 0x608edb80,
0x644fc637, 0x7a089632, 0x7ec98b85, 0x738aad5c, 0x774bb0eb,
0x4f040d56, 0x4bc510e1, 0x46863638, 0x42472b8f, 0x5c007b8a,
0x58c1663d, 0x558240e4, 0x51435d53, 0x251d3b9e, 0x21dc2629,
0x2c9f00f0, 0x285e1d47, 0x36194d42, 0x32d850f5, 0x3f9b762c,
0x3b5a6b9b, 0x0315d626, 0x07d4cb91, 0x0a97ed48, 0x0e56f0ff,
0x1011a0fa, 0x14d0bd4d, 0x19939b94, 0x1d528623, 0xf12f560e,
0xf5ee4bb9, 0xf8ad6d60, 0xfc6c70d7, 0xe22b20d2, 0xe6ea3d65,
0xeba91bbc, 0xef68060b, 0xd727bbb6, 0xd3e6a601, 0xdea580d8,
0xda649d6f, 0xc423cd6a, 0xc0e2d0dd, 0xcda1f604, 0xc960ebb3,
0xbd3e8d7e, 0xb9ff90c9, 0xb4bcb610, 0xb07daba7, 0xae3afba2,
0xaafbe615, 0xa7b8c0cc, 0xa379dd7b, 0x9b3660c6, 0x9ff77d71,
0x92b45ba8, 0x9675461f, 0x8832161a, 0x8cf30bad, 0x81b02d74,
0x857130c3, 0x5d8a9099, 0x594b8d2e, 0x5408abf7, 0x50c9b640,
0x4e8ee645, 0x4a4ffbf2, 0x470cdd2b, 0x43cdc09c, 0x7b827d21,
0x7f436096, 0x7200464f, 0x76c15bf8, 0x68860bfd, 0x6c47164a,
0x61043093, 0x65c52d24, 0x119b4be9, 0x155a565e, 0x18197087,
0x1cd86d30, 0x029f3d35, 0x065e2082, 0x0b1d065b, 0x0fdc1bec,
0x3793a651, 0x3352bbe6, 0x3e119d3f, 0x3ad08088, 0x2497d08d,
0x2056cd3a, 0x2d15ebe3, 0x29d4f654, 0xc5a92679, 0xc1683bce,
0xcc2b1d17, 0xc8ea00a0, 0xd6ad50a5, 0xd26c4d12, 0xdf2f6bcb,
0xdbee767c, 0xe3a1cbc1, 0xe760d676, 0xea23f0af, 0xeee2ed18,
0xf0a5bd1d, 0xf464a0aa, 0xf9278673, 0xfde69bc4, 0x89b8fd09,
0x8d79e0be, 0x803ac667, 0x84fbbdb0, 0x9abc8bd5, 0x9e7d9662,
0x933eb0bb, 0x97ffad0c, 0xafb010b1, 0xab710d06, 0xa6322bdf,
0xa2f33668, 0xbcb4666d, 0xb8757bda, 0xb5365d03, 0xb1f740b4
};

```

unsigned long memcrc(const unsigned char *b, size_t n)

```
{
```

```
/* Input arguments:
```

```
* const char* b == byte sequence to checksum
```

```
* size_t n == length of sequence
```

```

*/

register unsigned i, c, s = 0;

for (i = n; i > 0; --i) {
    c = (unsigned)(*b++);
    s = (s << 8) ^ crctab[(s >> 24) ^ c];
}

/* Extend with the length of the string. */
while (n != 0) {
    c = n & 0377;
    n >>= 8;
    s = (s << 8) ^ crctab[(s >> 24) ^ c];
}

return ~s;
}

```

The historical practice of writing the number of "blocks" has been changed to writing the number of octets, since the latter is not only more useful, but also since historical implementations have not been consistent in defining what a "block" meant. Octets are used instead of bytes because bytes can differ in size between systems.

The algorithm used was selected to increase the operational robustness of *cksum*. Neither the System V nor BSD *sum* algorithm was selected. Since each of these was different and each was the default behavior on those systems, no realistic compromise was available if either were selected—some set of historical applications would break. Therefore, the name was changed to *cksum*. Although the historical *sum* commands will probably continue to be provided for many years, programs designed for portability across systems should use the new name.

The algorithm selected is based on that used by the ISO/IEC 8802-3:1996 standard (Ethernet) for the frame check sequence field. The algorithm used does not match the technical definition of a *checksum*; the term is used for historical reasons. The length of the file is included in the CRC calculation because this parallels inclusion of a length field by Ethernet in its CRC, but also because it guards against inadvertent collisions between files that begin with different series of zero octets. The chance that two different files produce identical CRCs is much greater when their lengths are not considered. Keeping the length and the checksum of the file itself separate would yield a slightly more robust algorithm, but historical usage has always been that a single number (the checksum as printed) represents the signature of the file. It was decided that historical usage was the more important consideration.

Early proposals contained modifications to the Ethernet algorithm that involved extracting table values whenever an intermediate result became zero. This was demonstrated to be less robust than the current method and mathematically difficult to describe or justify.

The calculation used is identical to that given in pseudo-code in the referenced Sarwate article. The pseudo-code rendition is:

```

X <- 0; Y <- 0;
for i <- m - 1 step -1 until 0 do
    begin
        T <- X(1) ^ A[i];
        X(1) <- X(0); X(0) <- Y(1); Y(1) <- Y(0); Y(0) <- 0;
        comment: f[T] and f'[T] denote the T-th words in the
            table f and f' ;
        X <- X ^ f[T]; Y <- Y ^ f'[T];
    end

```

The pseudo-code is reproduced exactly as given; however, note that in the case of *cksum*, **A[i]** represents a byte of the file, the words **X** and **Y** are treated as a single 32-bit value, and the tables **f** and **f'** are a single table containing 32-bit values.

The referenced Sarwate article also discusses generating the table.

FUTURE DIRECTIONS

None.

SEE ALSO

None.

COPYRIGHT

Portions of this text are reprinted and reproduced in electronic form from IEEE Std 1003.1, 2003 Edition, Standard for Information Technology -- Portable Operating System Interface (POSIX), The Open Group Base Specifications Issue 6, Copyright (C) 2001-2003 by the Institute of Electrical and Electronics Engineers, Inc and The Open Group. In the event of any discrepancy between this version and the original IEEE and The Open Group Standard, the original IEEE and The Open Group Standard is the referee document. The original Standard can be obtained online at <http://www.open-group.org/unix/online.html>.

NAME

comm - select or reject lines common to two files

SYNOPSIS

comm [-123] *file1 file2*

DESCRIPTION

The *comm* utility shall read *file1* and *file2*, which should be ordered in the current collating sequence, and produce three text columns as output: lines only in *file1*, lines only in *file2*, and lines in both files.

If the lines in both files are not ordered according to the collating sequence of the current locale, the results are unspecified.

OPTIONS

The *comm* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section 12.2, Utility Syntax Guidelines.

The following options shall be supported:

- 1** Suppress the output column of lines unique to *file1*.
- 2** Suppress the output column of lines unique to *file2*.
- 3** Suppress the output column of lines duplicated in *file1* and *file2*.

OPERANDS

The following operands shall be supported:

- file1* A pathname of the first file to be compared. If *file1* is '-', the standard input shall be used.
- file2* A pathname of the second file to be compared. If *file2* is '-', the standard input shall be used.

If both *file1* and *file2* refer to standard input or to the same FIFO special, block special, or character special file, the results are undefined.

STDIN

The standard input shall be used only if one of the *file1* or *file2* operands refers to standard input. See the INPUT FILES section.

INPUT FILES

The input files shall be text files.

ENVIRONMENT VARIABLES

The following environment variables shall affect the execution of *comm*:

LANG Provide a default value for the internationalization variables that are unset or null. (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2, Internationalization Variables for the precedence of internationalization variables used to determine the values of locale categories.)

LC_ALL

If set to a non-empty string value, override the values of all the other internationalization variables.

LC_COLLATE

Determine the locale for the collating sequence *comm* expects to have been used when the input files were sorted.

LC_CTYPE

Determine the locale for the interpretation of sequences of bytes of text data as characters (for example, single-byte as opposed to multi-byte characters in arguments and input files).

LC_MESSAGES

Determine the locale that should be used to affect the format and contents of diagnostic messages written to standard error.

NLSPATH

Determine the location of message catalogs for the processing of *LC_MESSAGES*.

ASYNCHRONOUS EVENTS

Default.

STDOUT

The *comm* utility shall produce output depending on the options selected. If the **-1**, **-2**, and **-3** options are all selected, *comm* shall write nothing to standard output.

If the **-1** option is not selected, lines contained only in *file1* shall be written using the format:

"%s\n", <line in file1>

If the **-2** option is not selected, lines contained only in *file2* are written using the format:

"%s%s\n", <lead>, <line in file2>

where the string <lead> is as follows:

<tab> The **-1** option is not selected.

null string

The **-1** option is selected.

If the **-3** option is not selected, lines contained in both files shall be written using the format:

"%s%s\n", <lead>, <line in both>

where the string <lead> is as follows:

<tab><tab>

Neither the **-1** nor the **-2** option is selected.

<tab> Exactly one of the **-1** and **-2** options is selected.

null string

Both the **-1** and **-2** options are selected.

If the input files were ordered according to the collating sequence of the current locale, the lines written shall be in the collating sequence of the original lines.

STDERR

The standard error shall be used only for diagnostic messages.

OUTPUT FILES

None.

EXTENDED DESCRIPTION

None.

EXIT STATUS

The following exit values shall be returned:

0 All input files were successfully output as specified.

>0 An error occurred.

CONSEQUENCES OF ERRORS

Default.

The following sections are informative.

APPLICATION USAGE

If the input files are not properly presorted, the output of *comm* might not be useful.

EXAMPLES

If a file named **xcu** contains a sorted list of the utilities in this volume of IEEE Std 1003.1-2001, a file named **xpg3** contains a sorted list of the utilities specified in the X/Open Portability Guide, Issue 3, and a file named **svid89** contains a sorted list of the utilities in the System V Interface Definition Third Edition:

comm -23 xcu xpg3 | comm -23 - svid89

would print a list of utilities in this volume of IEEE Std 1003.1-2001 not specified by either of the other documents:

comm -12 xcu xpg3 | comm -12 - svid89

would print a list of utilities specified by all three documents, and:

comm -12 xpg3 svid89 | comm -23 - xcu

would print a list of utilities specified by both XPG3 and the SVID, but not specified in this volume of IEEE Std 1003.1-2001.

RATIONALE

None.

FUTURE DIRECTIONS

None.

SEE ALSO

cmp , *diff* , *sort* , *uniq*

COPYRIGHT

Portions of this text are reprinted and reproduced in electronic form from IEEE Std 1003.1, 2003 Edition, Standard for Information Technology -- Portable Operating System Interface (POSIX), The Open Group Base Specifications Issue 6, Copyright (C) 2001-2003 by the Institute of Electrical and Electronics Engineers, Inc and The Open Group. In the event of any discrepancy between this version and the original IEEE and The Open Group Standard, the original IEEE and The Open Group Standard is the referee document. The original Standard can be obtained online at <http://www.open-group.org/unix/online.html> .

NAME

cp - copy files

SYNOPSIS

cp [-**fi**p] *source_file* *target_file*

cp [-**fi**p] *source_file* ... *target*

cp -**R** [-**H** | -**L** | -**P**][-**fi**p] *source_file* ... *target*

cp -**r** [-**H** | -**L** | -**P**][-**fi**p] *source_file* ... *target*

DESCRIPTION

The first synopsis form is denoted by two operands, neither of which are existing files of type directory. The *cp* utility shall copy the contents of *source_file* (or, if *source_file* is a file of type symbolic link, the contents of the file referenced by *source_file*) to the destination path named by *target_file*.

The second synopsis form is denoted by two or more operands where the -**R** or -**r** options are not specified and the first synopsis form is not applicable. It shall be an error if any *source_file* is a file of type directory, if *target* does not exist, or if *target* is a file of a type defined by the System Interfaces volume of IEEE Std 1003.1-2001, but is not a file of type directory. The *cp* utility shall copy the contents of each *source_file* (or, if *source_file* is a file of type symbolic link, the contents of the file referenced by *source_file*) to the destination path named by the concatenation of *target*, a slash character, and the last component of *source_file*.

The third and fourth synopsis forms are denoted by two or more operands where the -**R** or -**r** options are specified. The *cp* utility shall copy each file in the file hierarchy rooted in each *source_file* to a destination path named as follows:

If *target* exists and is a file of type directory, the name of the corresponding destination path for each file in the file hierarchy shall be the concatenation of *target*, a slash character, and the pathname of the file relative to the directory containing *source_file*.

If *target* does not exist and two operands are specified, the name of the corresponding destination path for *source_file* shall be *target*; the name of the corresponding destination path for all other files in the file hierarchy shall be the concatenation of *target*, a slash character, and the pathname of the file relative to *source_file*.

It shall be an error if *target* does not exist and more than two operands are specified, or if *target* exists and is a file of a type defined by the System Interfaces volume of IEEE Std 1003.1-2001, but is not a file of type directory.

In the following description, the term *dest_file* refers to the file named by the destination path. The term *source_file* refers to the file that is being copied, whether specified as an operand or a file in a file hierarchy rooted in a *source_file* operand. If *source_file* is a file of type symbolic link:

If neither the -**R** nor -**r** options were specified, *cp* shall take actions based on the type and contents of the file referenced by the symbolic link, and not by the symbolic link itself.

If the -**R** option was specified:

If none of the options -**H**, -**L**, nor -**P** were specified, it is unspecified which of -**H**, -**L**, or -**P** will be used as a default.

If the -**H** option was specified, *cp* shall take actions based on the type and contents of the file referenced by any symbolic link specified as a *source_file* operand.

If the -**L** option was specified, *cp* shall take actions based on the type and contents of the file referenced by any symbolic link specified as a *source_file* operand or any symbolic links encountered during traversal of a file hierarchy.

If the -**P** option was specified, *cp* shall copy any symbolic link specified as a *source_file* operand and any symbolic links encountered during traversal of a file hierarchy, and shall not follow any symbolic links.

If the **-r** option was specified, the behavior is implementation-defined.

For each *source_file*, the following steps shall be taken:

If *source_file* references the same file as *dest_file*, *cp* may write a diagnostic message to standard error; it shall do nothing more with *source_file* and shall go on to any remaining files.

If *source_file* is of type directory, the following steps shall be taken: <ol type="a">

If neither the **-R** or **-r** options were specified, *cp* shall write a diagnostic message to standard error, do nothing more with *source_file*, and go on to any remaining files.

If *source_file* was not specified as an operand and *source_file* is dot or dot-dot, *cp* shall do nothing more with *source_file* and go on to any remaining files.

If *dest_file* exists and it is a file type not specified by the System Interfaces volume of IEEE Std 1003.1-2001, the behavior is implementation-defined.

If *dest_file* exists and it is not of type directory, *cp* shall write a diagnostic message to standard error, do nothing more with *source_file* or any files below *source_file* in the file hierarchy, and go on to any remaining files.

If the directory *dest_file* does not exist, it shall be created with file permission bits set to the same value as those of *source_file*, modified by the file creation mask of the user if the **-p** option was not specified, and then bitwise-inclusively OR'ed with S_IRWXU. If *dest_file* cannot be created, *cp* shall write a diagnostic message to standard error, do nothing more with *source_file*, and go on to any remaining files. It is unspecified if *cp* attempts to copy files in the file hierarchy rooted in *source_file*.

The files in the directory *source_file* shall be copied to the directory *dest_file*, taking the four steps (1 to 4) listed here with the files as *source_files*.

If *dest_file* was created, its file permission bits shall be changed (if necessary) to be the same as those of *source_file*, modified by the file creation mask of the user if the **-p** option was not specified.

The *cp* utility shall do nothing more with *source_file* and go on to any remaining files.

If *source_file* is of type regular file, the following steps shall be taken: <ol type="a">

If *dest_file* exists, the following steps shall be taken: <ol type="i">

If the **-i** option is in effect, the *cp* utility shall write a prompt to the standard error and read a line from the standard input. If the response is not affirmative, *cp* shall do nothing more with *source_file* and go on to any remaining files.

A file descriptor for *dest_file* shall be obtained by performing actions equivalent to the *open()* function defined in the System Interfaces volume of IEEE Std 1003.1-2001 called using *dest_file* as the *path* argument, and the bitwise-inclusive OR of O_WRONLY and O_TRUNC as the *oflag* argument.

If the attempt to obtain a file descriptor fails and the **-f** option is in effect, *cp* shall attempt to remove the file by performing actions equivalent to the *unlink()* function defined in the System Interfaces volume of IEEE Std 1003.1-2001 called using *dest_file* as the *path* argument. If this attempt succeeds, *cp* shall continue with step 3b.

If *dest_file* does not exist, a file descriptor shall be obtained by performing actions equivalent to the *open()* function defined in the System Interfaces volume of IEEE Std 1003.1-2001 called using *dest_file* as the *path* argument, and the bitwise-inclusive OR of O_WRONLY and O_CREAT as the *oflag* argument. The file permission bits of *source_file* shall be the *mode* argument.

If the attempt to obtain a file descriptor fails, *cp* shall write a diagnostic message to standard error, do nothing more with *source_file*, and go on to any remaining files.

The contents of *source_file* shall be written to the file descriptor. Any write errors shall cause *cp* to write a diagnostic message to standard error and continue to step 3e.

The file descriptor shall be closed.

The *cp* utility shall do nothing more with *source_file*. If a write error occurred in step 3d, it is unspecified if *cp* continues with any remaining files. If no write error occurred in step 3d, *cp* shall go on to any remaining files.

Otherwise, the following steps shall be taken: <ol type="a">

If the **-r** option was specified, the behavior is implementation-defined.

If the **-R** option was specified, the following steps shall be taken: <ol type="i">

The *dest_file* shall be created with the same file type as *source_file*.

If *source_file* is a file of type FIFO, the file permission bits shall be the same as those of *source_file*, modified by the file creation mask of the user if the **-p** option was not specified. Otherwise, the permissions, owner ID, and group ID of *dest_file* are implementation-defined.

If this creation fails for any reason, *cp* shall write a diagnostic message to standard error, do nothing more with *source_file*, and go on to any remaining files.

If *source_file* is a file of type symbolic link, the pathname contained in *dest_file* shall be the same as the pathname contained in *source_file*.

If this fails for any reason, *cp* shall write a diagnostic message to standard error, do nothing more with *source_file*, and go on to any remaining files.

If the implementation provides additional or alternate access control mechanisms (see the Base Definitions volume of IEEE Std 1003.1-2001, Section 4.4, File Access Permissions), their effect on copies of files is implementation-defined.

OPTIONS

The *cp* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section 12.2, Utility Syntax Guidelines.

The following options shall be supported:

- f** If a file descriptor for a destination file cannot be obtained, as described in step 3.a.ii., attempt to unlink the destination file and proceed.
- H** Take actions based on the type and contents of the file referenced by any symbolic link specified as a *source_file* operand.
- i** Write a prompt to standard error before copying to any existing destination file. If the response from the standard input is affirmative, the copy shall be attempted; otherwise, it shall not.
- L** Take actions based on the type and contents of the file referenced by any symbolic link specified as a *source_file* operand or any symbolic links encountered during traversal of a file hierarchy.
- P** Take actions on any symbolic link specified as a *source_file* operand or any symbolic link encountered during traversal of a file hierarchy.
- p** Duplicate the following characteristics of each source file in the corresponding destination file:

The time of last data modification and time of last access. If this duplication fails for any reason, *cp* shall write a diagnostic message to standard error.

The user ID and group ID. If this duplication fails for any reason, it is unspecified whether *cp* writes a diagnostic message to standard error.

The file permission bits and the S_ISUID and S_ISGID bits. Other, implementation-defined, bits may be duplicated as well. If this duplication fails for any reason, *cp* shall write a diagnostic message to standard error.

If the user ID or the group ID cannot be duplicated, the file permission bits S_ISUID and S_ISGID shall be cleared. If these bits are present in the source file but are not duplicated in the destination file, it is unspecified whether *cp* writes a diagnostic message to standard error.

The order in which the preceding characteristics are duplicated is unspecified. The *dest_file* shall not be deleted if these characteristics cannot be preserved.

- R** Copy file hierarchies.
- r** Copy file hierarchies. The treatment of special files is implementation-defined.

Specifying more than one of the mutually-exclusive options **-H**, **-L**, and **-P** shall not be considered an error. The last option specified shall determine the behavior of the utility.

OPERANDS

The following operands shall be supported:

source_file

A pathname of a file to be copied.

target_file

A pathname of an existing or nonexistent file, used for the output when a single file is copied.

target

A pathname of a directory to contain the copied files.

STDIN

The standard input shall be used to read an input line in response to each prompt specified in the **STDERR** section. Otherwise, the standard input shall not be used.

INPUT FILES

The input files specified as operands may be of any file type.

ENVIRONMENT VARIABLES

The following environment variables shall affect the execution of *cp*:

LANG Provide a default value for the internationalization variables that are unset or null. (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2, Internationalization Variables for the precedence of internationalization variables used to determine the values of locale categories.)

LC_ALL

If set to a non-empty string value, override the values of all the other internationalization variables.

LC_COLLATE

Determine the locale for the behavior of ranges, equivalence classes, and multi-character collating elements used in the extended regular expression defined for the **yesexpr** locale keyword in the **LC_MESSAGES** category.

LC_CTYPE

Determine the locale for the interpretation of sequences of bytes of text data as characters (for example, single-byte as opposed to multi-byte characters in arguments and input files) and the behavior of character classes used in the extended regular expression defined for the **yesexpr** locale keyword in the **LC_MESSAGES** category.

LC_MESSAGES

Determine the locale for the processing of affirmative responses that should be used to affect the format and contents of diagnostic messages written to standard error.

NLSPATH

Determine the location of message catalogs for the processing of **LC_MESSAGES**.

ASYNCHRONOUS EVENTS

Default.

STDOUT

Not used.

STDERR

A prompt shall be written to standard error under the conditions specified in the **DESCRIPTION** section. The prompt shall contain the destination pathname, but its format is otherwise unspecified. Otherwise, the standard error shall be used only for diagnostic messages.

OUTPUT FILES

The output files may be of any type.

EXTENDED DESCRIPTION

None.

EXIT STATUS

The following exit values shall be returned:

- 0 All files were copied successfully.
- >0 An error occurred.

CONSEQUENCES OF ERRORS

If *cp* is prematurely terminated by a signal or error, files or file hierarchies may be only partially copied and files and directories may have incorrect permissions or access and modification times.

The following sections are informative.

APPLICATION USAGE

The difference between **-R** and **-r** is in the treatment by *cp* of file types other than regular and directory. The original **-r** flag, for historic reasons, does not handle special files any differently from regular files, but always reads the file and copies its contents. This has obvious problems in the presence of special file types; for example, character devices, FIFOs, and sockets. The **-R** option is intended to recreate the file hierarchy and the **-r** option supports historical practice. It was anticipated that a future version of this volume of IEEE Std 1003.1-2001 would deprecate the **-r** option, and for that reason, there has been no attempt to fix its behavior with respect to FIFOs or other file types where copying the file is clearly wrong. However, some implementations support **-r** with the same abilities as the **-R** defined in this volume of IEEE Std 1003.1-2001. To accommodate them as well as systems that do not, the differences between **-r** and **-R** are implementation-defined. Implementations may make them identical. The **-r** option is marked obsolescent.

The set-user-ID and set-group-ID bits are explicitly cleared when files are created. This is to prevent users from creating programs that are set-user-ID or set-group-ID to them when copying files or to make set-user-ID or set-group-ID files accessible to new groups of users. For example, if a file is set-user-ID and the copy has a different group ID than the source, a new group of users has execute permission to a set-user-ID program than did previously. In particular, this is a problem for superusers copying users' trees.

EXAMPLES

None.

RATIONALE

The **-i** option exists on BSD systems, giving applications and users a way to avoid accidentally removing files when copying. Although the 4.3 BSD version does not prompt if the standard input is not a terminal, the standard developers decided that use of **-i** is a request for interaction, so when the destination path exists, the utility takes instructions from whatever responds on standard input.

The exact format of the interactive prompts is unspecified. Only the general nature of the contents of prompts are specified because implementations may desire more descriptive prompts than those used on historical implementations. Therefore, an application using the **-i** option relies on the system to provide the most suitable dialog directly with the user, based on the behavior specified.

The **-p** option is historical practice on BSD systems, duplicating the time of last data modification and time of last access. This volume of IEEE Std 1003.1-2001 extends it to preserve the user and group IDs, as well as the file permissions. This requirement has obvious problems in that the directories are almost certainly modified after being copied. This volume of IEEE Std 1003.1-2001 requires that the modification times be preserved. The statement that the order in which the characteristics are duplicated is unspecified is to permit implementations to provide the maximum amount of security for the user. Implementations should take into account the obvious security issues involved in setting the owner, group, and mode in the wrong order or creating files with an owner, group, or mode different from the final value.

It is unspecified whether *cp* writes diagnostic messages when the user and group IDs cannot be set due to the widespread practice of users using **-p** to duplicate some portion of the file characteristics, indifferent to the duplication of others. Historic implementations only write diagnostic messages on errors other than [EPERM].

The **-r** option is historical practice on BSD and BSD-derived systems, copying file hierarchies as opposed to single files. This functionality is used heavily in historical applications, and its loss would significantly decrease consensus. The **-R** option was added as a close synonym to the **-r** option,

selected for consistency with all other options in this volume of IEEE Std 1003.1-2001 that do recursive directory descent.

When a failure occurs during the copying of a file hierarchy, *cp* is required to attempt to copy files that are on the same level in the hierarchy or above the file where the failure occurred. It is unspecified if *cp* shall attempt to copy files below the file where the failure occurred (which cannot succeed in any case).

Permissions, owners, and groups of created special file types have been deliberately left as implementation-defined. This is to allow systems to satisfy special requirements (for example, allowing users to create character special devices, but requiring them to be owned by a certain group). In general, it is strongly suggested that the permissions, owner, and group be the same as if the user had run the historical *mknod*, *ln*, or other utility to create the file. It is also probable that additional privileges are required to create block, character, or other implementation-defined special file types.

Additionally, the **-p** option explicitly requires that all set-user-ID and set-group-ID permissions be discarded if any of the owner or group IDs cannot be set. This is to keep users from unintentionally giving away special privilege when copying programs.

When creating regular files, historical versions of *cp* use the mode of the source file as modified by the file mode creation mask. Other choices would have been to use the mode of the source file unmodified by the creation mask or to use the same mode as would be given to a new file created by the user (plus the execution bits of the source file) and then modify it by the file mode creation mask. In the absence of any strong reason to change historic practice, it was in large part retained.

When creating directories, historical versions of *cp* use the mode of the source directory, plus read, write, and search bits for the owner, as modified by the file mode creation mask. This is done so that *cp* can copy trees where the user has read permission, but the owner does not. A side effect is that if the file creation mask denies the owner permissions, *cp* fails. Also, once the copy is done, historical versions of *cp* set the permissions on the created directory to be the same as the source directory, unmodified by the file creation mask.

This behavior has been modified so that *cp* is always able to create the contents of the directory, regardless of the file creation mask. After the copy is done, the permissions are set to be the same as the source directory, as modified by the file creation mask. This latter change from historical behavior is to prevent users from accidentally creating directories with permissions beyond those they would normally set and for consistency with the behavior of *cp* in creating files.

It is not a requirement that *cp* detect attempts to copy a file to itself; however, implementations are strongly encouraged to do so. Historical implementations have detected the attempt in most cases.

There are two methods of copying subtrees in this volume of IEEE Std 1003.1-2001. The other method is described as part of the *pax* utility (see *pax*). Both methods are historical practice. The *cp* utility provides a simpler, more intuitive interface, while *pax* offers a finer granularity of control. Each provides additional functionality to the other; in particular, *pax* maintains the hard-link structure of the hierarchy, while *cp* does not. It is the intention of the standard developers that the results be similar (using appropriate option combinations in both utilities). The results are not required to be identical; there seemed insufficient gain to applications to balance the difficulty of implementations having to guarantee that the results would be exactly identical.

The wording allowing *cp* to copy a directory to implementation-defined file types not specified by the System Interfaces volume of IEEE Std 1003.1-2001 is provided so that implementations supporting symbolic links are not required to prohibit copying directories to symbolic links. Other extensions to the System Interfaces volume of IEEE Std 1003.1-2001 file types may need to use this loophole as well.

FUTURE DIRECTIONS

The **-r** option may be removed; use **-R** instead.

SEE ALSO

mv, *find*, *ln*, *pax*, the System Interfaces volume of IEEE Std 1003.1-2001, *open()*, *unlink()*

COPYRIGHT

Portions of this text are reprinted and reproduced in electronic form from IEEE Std 1003.1, 2003 Edition, Standard for Information Technology -- Portable Operating System Interface (POSIX), The Open Group Base Specifications Issue 6, Copyright (C) 2001-2003 by the Institute of Electrical and Electronics Engineers, Inc and The Open Group. In the event of any discrepancy between this version and the

original IEEE and The Open Group Standard, the original IEEE and The Open Group Standard is the referee document. The original Standard can be obtained online at <http://www.open-group.org/unix/online.html> .

NAME

csplit - split files based on context

SYNOPSIS

csplit [-ks][-f *prefix*][-n *number*] *file* *arg1* ...*argn*

DESCRIPTION

The *csplit* utility shall read the file named by the *file* operand, write all or part of that file into other files as directed by the *arg* operands, and write the sizes of the files.

OPTIONS

The *csplit* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section 12.2, Utility Syntax Guidelines.

The following options shall be supported:

- f** *prefix*
Name the created files *prefix* **00**, *prefix* **01**, ..., *prefix**n*. The default is **xx00** ... **xx** *n*. If the *prefix* argument would create a filename exceeding {NAME_MAX} bytes, an error shall result, *csplit* shall exit with a diagnostic message, and no files shall be created.
- k**
Leave previously created files intact. By default, *csplit* shall remove created files if an error occurs.
- n** *number*
Use *number* decimal digits to form filenames for the file pieces. The default shall be 2.
- s**
Suppress the output of file size messages.

OPERANDS

The following operands shall be supported:

file The pathname of a text file to be split. If *file* is '-', the standard input shall be used.

The operands *arg1* ... *argn* can be a combination of the following:

/regexp[offset]

A file shall be created using the content of the lines from the current line up to, but not including, the line that results from the evaluation of the regular expression with *offset*, if any, applied. The regular expression *regexp* shall follow the rules for basic regular expressions described in the Base Definitions volume of IEEE Std 1003.1-2001, Section 9.3, Basic Regular Expressions. The application shall use the sequence "/" to specify a slash character within the *regexp*. The optional offset shall be a positive or negative integer value representing a number of lines. A positive integer value can be preceded by '+'. If the selection of lines from an *offset* expression of this type would create a file with zero lines, or one with greater than the number of lines left in the input file, the results are unspecified. After the section is created, the current line shall be set to the line that results from the evaluation of the regular expression with any offset applied. If the current line is the first line in the file and a regular expression operation has not yet been performed, the pattern match of *regexp* shall be applied from the current line to the end of the file. Otherwise, the pattern match of *regexp* shall be applied from the line following the current line to the end of the file.

%*regexp*%*[offset]*

Equivalent to */regexp[offset]*, except that no file shall be created for the selected section of the input file. The application shall use the sequence "%" to specify a percent-sign character within the *regexp*.

line_no Create a file from the current line up to (but not including) the line number *line_no*. Lines in the file shall be numbered starting at one. The current line becomes *line_no*.

{*num*} Repeat operand. This operand can follow any of the operands described previously. If it follows a *regexp* type operand, that operand shall be applied *num* more times. If it follows a *line_no*

operand, the file shall be split every *line_no* lines, *num* times, from that point.

An error shall be reported if an operand does not reference a line between the current position and the end of the file.

STDIN

See the INPUT FILES section.

INPUT FILES

The input file shall be a text file.

ENVIRONMENT VARIABLES

The following environment variables shall affect the execution of *csplit*:

LANG Provide a default value for the internationalization variables that are unset or null. (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2, Internationalization Variables for the precedence of internationalization variables used to determine the values of locale categories.)

LC_ALL

If set to a non-empty string value, override the values of all the other internationalization variables.

LC_COLLATE

Determine the locale for the behavior of ranges, equivalence classes, and multi-character collating elements within regular expressions.

LC_CTYPE

Determine the locale for the interpretation of sequences of bytes of text data as characters (for example, single-byte as opposed to multi-byte characters in arguments and input files) and the behavior of character classes within regular expressions.

LC_MESSAGES

Determine the locale that should be used to affect the format and contents of diagnostic messages written to standard error.

NLSPATH

Determine the location of message catalogs for the processing of *LC_MESSAGES*.

ASYNCHRONOUS EVENTS

If the **-k** option is specified, created files shall be retained. Otherwise, the default action occurs.

STDOUT

Unless the **-s** option is used, the standard output shall consist of one line per file created, with a format as follows:

"%d\n", *<file size in bytes>*

STDERR

The standard error shall be used only for diagnostic messages.

OUTPUT FILES

The output files shall contain portions of the original input file; otherwise, unchanged.

EXTENDED DESCRIPTION

None.

EXIT STATUS

The following exit values shall be returned:

- 0 Successful completion.
- >0 An error occurred.

CONSEQUENCES OF ERRORS

By default, created files shall be removed if an error occurs. When the **-k** option is specified, created files shall not be removed if an error occurs.

The following sections are informative.

APPLICATION USAGE

None.

EXAMPLES

This example creates four files, **cobol00** ... **cobol03**:

```
csplit -f cobol file '/procedure division/' /par5./ /par16./
```

After editing the split files, they can be recombined as follows:

```
cat cobol0[0-3] > file
```

Note that this example overwrites the original file.

This example would split the file after the first 99 lines, and every 100 lines thereafter, up to 9999 lines; this is because lines in the file are numbered from 1 rather than zero, for historical reasons:

```
csplit -k file 100 {99}
```

Assuming that **prog.c** follows the C-language coding convention of ending routines with a **'}'** at the beginning of the line, this example creates a file containing each separate C routine (up to 21) in **prog.c**:

```
csplit -k prog.c '%main(%%' '^}'+1' {20}
```

RATIONALE

The **-n** option was added to extend the range of filenames that could be handled.

Consideration was given to adding a **-a** flag to use the alphabetic filename generation used by the historical *split* utility, but the functionality added by the **-n** option was deemed to make alphabetic naming unnecessary.

FUTURE DIRECTIONS

None.

SEE ALSO

sed, *split*

COPYRIGHT

Portions of this text are reprinted and reproduced in electronic form from IEEE Std 1003.1, 2003 Edition, Standard for Information Technology -- Portable Operating System Interface (POSIX), The Open Group Base Specifications Issue 6, Copyright (C) 2001-2003 by the Institute of Electrical and Electronics Engineers, Inc and The Open Group. In the event of any discrepancy between this version and the original IEEE and The Open Group Standard, the original IEEE and The Open Group Standard is the referee document. The original Standard can be obtained online at <http://www.open-group.org/unix/online.html>.

NAME

cut - cut out selected fields of each line of a file

SYNOPSIS

cut -b *list* [-n] [*file* ...]

cut -c *list* [*file* ...]

cut -f *list* [-d *delim*][-s][*file* ...]

DESCRIPTION

The *cut* utility shall cut out bytes (**-b** option), characters (**-c** option), or character-delimited fields (**-f** option) from each line in one or more files, concatenate them, and write them to standard output.

OPTIONS

The *cut* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section 12.2, Utility Syntax Guidelines.

The application shall ensure that the option-argument *list* (see options **-b**, **-c**, and **-f** below) is a comma-separated list or <blank>-separated list of positive numbers and ranges. Ranges can be in three forms. The first is two positive numbers separated by a hyphen (*low-high*), which represents all fields from the first number to the second number. The second is a positive number preceded by a hyphen (*-high*), which represents all fields from field number 1 to that number. The third is a positive number followed by a hyphen (*low-*), which represents that number to the last field, inclusive. The elements in *list* can be repeated, can overlap, and can be specified in any order, but the bytes, characters, or fields selected shall be written in the order of the input data. If an element appears in the selection list more than once, it shall be written exactly once.

The following options shall be supported:

-b *list* Cut based on a *list* of bytes. Each selected byte shall be output unless the **-n** option is also specified. It shall not be an error to select bytes not present in the input line.

-c *list* Cut based on a *list* of characters. Each selected character shall be output. It shall not be an error to select characters not present in the input line.

-d *delim*

Set the field delimiter to the character *delim*. The default is the <tab>.

-f *list* Cut based on a *list* of fields, assumed to be separated in the file by a delimiter character (see **-d**). Each selected field shall be output. Output fields shall be separated by a single occurrence of the field delimiter character. Lines with no field delimiters shall be passed through intact, unless **-s** is specified. It shall not be an error to select fields not present in the input line.

-n Do not split characters. When specified with the **-b** option, each element in *list* of the form *low-high* (hyphen-separated numbers) shall be modified as follows:

If the byte selected by *low* is not the first byte of a character, *low* shall be decremented to select the first byte of the character originally selected by *low*. If the byte selected by *high* is not the last byte of a character, *high* shall be decremented to select the last byte of the character prior to the character originally selected by *high*, or zero if there is no prior character. If the resulting range element has *high* equal to zero or *low* greater than *high*, the list element shall be dropped from *list* for that input line without causing an error.

Each element in *list* of the form *low-* shall be treated as above with *high* set to the number of bytes in the current line, not including the terminating <newline>. Each element in *list* of the form *-high* shall be treated as above with *low* set to 1. Each element in *list* of the form *num* (a single number) shall be treated as above with *low* set to *num* and *high* set to *num*.

-s Suppress lines with no delimiter characters, when used with the **-f** option. Unless specified, lines with no delimiters shall be passed through untouched.

OPERANDS

The following operand shall be supported:

file A pathname of an input file. If no *file* operands are specified, or if a *file* operand is '-' , the standard input shall be used.

STDIN

The standard input shall be used only if no *file* operands are specified, or if a *file* operand is '-' . See the INPUT FILES section.

INPUT FILES

The input files shall be text files, except that line lengths shall be unlimited.

ENVIRONMENT VARIABLES

The following environment variables shall affect the execution of *cut*:

LANG Provide a default value for the internationalization variables that are unset or null. (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2, Internationalization Variables for the precedence of internationalization variables used to determine the values of locale categories.)

LC_ALL

If set to a non-empty string value, override the values of all the other internationalization variables.

LC_CTYPE

Determine the locale for the interpretation of sequences of bytes of text data as characters (for example, single-byte as opposed to multi-byte characters in arguments and input files).

LC_MESSAGES

Determine the locale that should be used to affect the format and contents of diagnostic messages written to standard error.

NLSPATH

Determine the location of message catalogs for the processing of *LC_MESSAGES* .

ASYNCHRONOUS EVENTS

Default.

STDOUT

The *cut* utility output shall be a concatenation of the selected bytes, characters, or fields (one of the following):

`"%s\n", <concatenation of bytes>`

`"%s\n", <concatenation of characters>`

`"%s\n", <concatenation of fields and field delimiters>`

STDERR

The standard error shall be used only for diagnostic messages.

OUTPUT FILES

None.

EXTENDED DESCRIPTION

None.

EXIT STATUS

The following exit values shall be returned:

0 All input files were output successfully.

>0 An error occurred.

CONSEQUENCES OF ERRORS

Default.

The following sections are informative.

APPLICATION USAGE

Earlier versions of the *cut* utility worked in an environment where bytes and characters were considered equivalent (modulo <backspace> and <tab> processing in some implementations). In the extended world of multi-byte characters, the new **-b** option has been added. The **-n** option (used with **-b**) allows it to be used to act on bytes rounded to character boundaries. The algorithm specified for **-n** guarantees that:

```
cut -b 1-500 -n file > file1
cut -b 501- -n file > file2
```

ends up with all the characters in **file** appearing exactly once in **file1** or **file2**. (There is, however, a <newline> in both **file1** and **file2** for each <newline> in **file**.)

EXAMPLES

Examples of the option qualifier list:

- 1,4,7 Select the first, fourth, and seventh bytes, characters, or fields and field delimiters.
- 1-3,8 Equivalent to 1,2,3,8.
- 5,10 Equivalent to 1,2,3,4,5,10.
- 3- Equivalent to third to last, inclusive.

The *low- high* forms are not always equivalent when used with **-b** and **-n** and multi-byte characters; see the description of **-n**.

The following command:

```
cut -d : -f 1,6 /etc/passwd
```

reads the System V password file (user database) and produces lines of the form:

```
<user ID>:<home directory>
```

Most utilities in this volume of IEEE Std 1003.1-2001 work on text files. The *cut* utility can be used to turn files with arbitrary line lengths into a set of text files containing the same data. The *paste* utility can be used to create (or recreate) files with arbitrary line lengths. For example, if **file** contains long lines:

```
cut -b 1-500 -n file > file1
cut -b 501- -n file > file2
```

creates **file1** (a text file) with lines no longer than 500 bytes (plus the <newline>) and **file2** that contains the remainder of the data from **file**. (Note that **file2** is not a text file if there are lines in **file** that are longer than 500 + {LINE_MAX} bytes.) The original file can be recreated from **file1** and **file2** using the command:

```
paste -d ""\0" file1 file2 > file
```

RATIONALE

Some historical implementations do not count <backspace>s in determining character counts with the **-c** option. This may be useful for using *cut* for processing *nroff* output. It was deliberately decided not to have the **-c** option treat either <backspace>s or <tab>s in any special fashion. The *fold* utility does

treat these characters specially.

Unlike other utilities, some historical implementations of *cut* exit after not finding an input file, rather than continuing to process the remaining *file* operands. This behavior is prohibited by this volume of IEEE Std 1003.1-2001, where only the exit status is affected by this problem.

The behavior of *cut* when provided with either mutually-exclusive options or options that do not work logically together has been deliberately left unspecified in favor of global wording in *Utility Description Defaults*.

The OPTIONS section was changed in response to IEEE PASC Interpretation 1003.2 #149. The change represents historical practice on all known systems. The original standard was ambiguous on the nature of the output.

The *list* option-arguments are historically used to select the portions of the line to be written, but do not affect the order of the data. For example:

```
echo abcdefghi | cut -c6,2,4-7,1
```

yields "abdefg" .

A proposal to enhance *cut* with the following option:

- o** Preserve the selected field order. When this option is specified, each byte, character, or field (or ranges of such) shall be written in the order specified by the *list* option-argument, even if this requires multiple outputs of the same bytes, characters, or fields.

was rejected because this type of enhancement is outside the scope of the IEEE P1003.2b draft standard.

FUTURE DIRECTIONS

None.

SEE ALSO

grep , *paste* , *Parameters and Variables*

COPYRIGHT

Portions of this text are reprinted and reproduced in electronic form from IEEE Std 1003.1, 2003 Edition, Standard for Information Technology -- Portable Operating System Interface (POSIX), The Open Group Base Specifications Issue 6, Copyright (C) 2001-2003 by the Institute of Electrical and Electronics Engineers, Inc and The Open Group. In the event of any discrepancy between this version and the original IEEE and The Open Group Standard, the original IEEE and The Open Group Standard is the referee document. The original Standard can be obtained online at <http://www.open-group.org/unix/online.html> .

NAME

date - write the date and time

SYNOPSIS

date [-u] [+format]

date [-u] *mmddhhmm*[[*cc*]*yy*]

DESCRIPTION

The *date* utility shall write the date and time to standard output or attempt to set the system date and time. By default, the current date and time shall be written. If an operand beginning with '+' is specified, the output format of *date* shall be controlled by the conversion specifications and other text in the operand.

OPTIONS

The *date* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section 12.2, Utility Syntax Guidelines.

The following option shall be supported:

-u Perform operations as if the *TZ* environment variable was set to the string "**UTC0**", or its equivalent historical value of "**GMT0**". Otherwise, *date* shall use the timezone indicated by the *TZ* environment variable or the system default if that variable is unset or null.

OPERANDS

The following operands shall be supported:

+format

When the format is specified, each conversion specifier shall be replaced in the standard output by its corresponding value. All other characters shall be copied to the output without change. The output shall always be terminated with a <newline>.

Conversion Specifications

%a	Locale's abbreviated weekday name.
%A	Locale's full weekday name.
%b	Locale's abbreviated month name.
%B	Locale's full month name.
%c	Locale's appropriate date and time representation.
%C	Century (a year divided by 100 and truncated to an integer) as a decimal number [00,99].
%d	Day of the month as a decimal number [01,31].
%D	Date in the format <i>mm/dd/yy</i> .
%e	Day of the month as a decimal number [1,31] in a two-digit field with leading space character fill.
%h	A synonym for %b .
%H	Hour (24-hour clock) as a decimal number [00,23].
%I	Hour (12-hour clock) as a decimal number [01,12].
%j	Day of the year as a decimal number [001,366].
%m	Month as a decimal number [01,12].
%M	Minute as a decimal number [00,59].
%n	A <newline>.

%p	Locale's equivalent of either AM or PM.
%r	12-hour clock time [01,12] using the AM/PM notation; in the POSIX locale, this shall be equivalent to %I : %M : %S %p .
%S	Seconds as a decimal number [00,60].
%t	A <tab>.
%T	24-hour clock time [00,23] in the format <i>HH:MM:SS</i> .
%u	Weekday as a decimal number [1,7] (1=Monday).
%U	Week of the year (Sunday as the first day of the week) as a decimal number [00,53]. All days in a new year preceding the first Sunday shall be considered to be in week 0.
%V	Week of the year (Monday as the first day of the week) as a decimal number [01,53]. If the week containing January 1 has four or more days in the new year, then it shall be considered week 1; otherwise, it shall be the last week of the previous year, and the next week shall be week 1.
%w	Weekday as a decimal number [0,6] (0=Sunday).
%W	Week of the year (Monday as the first day of the week) as a decimal number [00,53]. All days in a new year preceding the first Monday shall be considered to be in week 0.
%x	Locale's appropriate date representation.
%X	Locale's appropriate time representation.
%y	Year within century [00,99].
%Y	Year with century as a decimal number.
%Z	Timezone name, or no characters if no timezone is determinable.
%%	A percent sign character.

See the Base Definitions volume of IEEE Std 1003.1-2001, Section 7.3.5, LC_TIME for the conversion specifier values in the POSIX locale.

Modified Conversion Specifications

Some conversion specifiers can be modified by the **E** and **O** modifier characters to indicate a different format or specification as specified in the *LC_TIME* locale description (see the Base Definitions volume of IEEE Std 1003.1-2001, Section 7.3.5, LC_TIME). If the corresponding keyword (see **era**, **era_year**, **era_d_fmt**, and **alt_digits** in the Base Definitions volume of IEEE Std 1003.1-2001, Section 7.3.5, LC_TIME) is not specified or not supported for the current locale, the unmodified conversion specifier value shall be used.

%Ec	Locale's alternative appropriate date and time representation.
%EC	The name of the base year (period) in the locale's alternative representation.
%Ex	Locale's alternative date representation.
%EX	Locale's alternative time representation.
%Ey	Offset from %EC (year only) in the locale's alternative representation.
%EY	Full alternative year representation.
%Od	Day of month using the locale's alternative numeric symbols.
%Oe	Day of month using the locale's alternative numeric symbols.
%OH	Hour (24-hour clock) using the locale's alternative numeric symbols.
%OI	Hour (12-hour clock) using the locale's alternative numeric symbols.
%Om	Month using the locale's alternative numeric symbols.
%OM	Minutes using the locale's alternative numeric symbols.
%OS	Seconds using the locale's alternative numeric symbols.

- %Ou** Weekday as a number in the locale's alternative representation (Monday = 1).
- %OU** Week number of the year (Sunday as the first day of the week) using the locale's alternative numeric symbols.
- %OV** Week number of the year (Monday as the first day of the week, rules corresponding to **%V**), using the locale's alternative numeric symbols.
- %Ow** Weekday as a number in the locale's alternative representation (Sunday = 0).
- %OW** Week number of the year (Monday as the first day of the week) using the locale's alternative numeric symbols.
- %Oy** Year (offset from **%C**) in alternative representation.

mmddhhmm[[cc]yy]

Attempt to set the system date and time from the value given in the operand. This is only possible if the user has appropriate privileges and the system permits the setting of the system date and time. The first *mm* is the month (number); *dd* is the day (number); *hh* is the hour (number, 24-hour system); the second *mm* is the minute (number); *cc* is the century and is the first two digits of the year (this is optional); *yy* is the last two digits of the year and is optional. If century is not specified, then values in the range [69,99] shall refer to years 1969 to 1999 inclusive, and values in the range [00,68] shall refer to years 2000 to 2068 inclusive. The current year is the default if *yy* is omitted.

Note: It is expected that in a future version of IEEE Std 1003.1-2001 the default century inferred from a 2-digit year will change. (This would apply to all commands accepting a 2-digit year as input.)

STDIN

Not used.

INPUT FILES

None.

ENVIRONMENT VARIABLES

The following environment variables shall affect the execution of *date*:

LANG Provide a default value for the internationalization variables that are unset or null. (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2, Internationalization Variables for the precedence of internationalization variables used to determine the values of locale categories.)

LC_ALL

If set to a non-empty string value, override the values of all the other internationalization variables.

LC_CTYPE

Determine the locale for the interpretation of sequences of bytes of text data as characters (for example, single-byte as opposed to multi-byte characters in arguments).

LC_MESSAGES

Determine the locale that should be used to affect the format and contents of diagnostic messages written to standard error.

LC_TIME

Determine the format and contents of date and time strings written by *date*.

NLSPATH

Determine the location of message catalogs for the processing of *LC_MESSAGES*.

TZ

Determine the timezone in which the time and date are written, unless the **-u** option is specified. If the *TZ* variable is unset or null and **-u** is not specified, an unspecified system default timezone is used.

date(P)

date(P)

ASYNCHRONOUS EVENTS

Default.

STDOUT

When no formatting operand is specified, the output in the POSIX locale shall be equivalent to specifying:

date "+%a %b %e %H:%M:%S %Z %Y"

STDERR

The standard error shall be used only for diagnostic messages.

OUTPUT FILES

None.

EXTENDED DESCRIPTION

None.

EXIT STATUS

The following exit values shall be returned:

- 0 The date was written successfully.
- >0 An error occurred.

CONSEQUENCES OF ERRORS

Default.

The following sections are informative.

APPLICATION USAGE

Conversion specifiers are of unspecified format when not in the POSIX locale. Some of them can contain <newline>s in some locales, so it may be difficult to use the format shown in standard output for parsing the output of *date* in those locales.

The range of values for **%S** extends from 0 to 60 seconds to accommodate the occasional leap second.

Although certain of the conversion specifiers in the POSIX locale (such as the name of the month) are shown with initial capital letters, this need not be the case in other locales. Programs using these fields may need to adjust the capitalization if the output is going to be used at the beginning of a sentence.

The date string formatting capabilities are intended for use in Gregorian-style calendars, possibly with a different starting year (or years). The **%x** and **%c** conversion specifications, however, are intended for local representation; these may be based on a different, non-Gregorian calendar.

The **%C** conversion specification was introduced to allow a fallback for the **%EC** (alternative year format base year); it can be viewed as the base of the current subdivision in the Gregorian calendar. The century number is calculated as the year divided by 100 and truncated to an integer; it should not be confused with the use of ordinal numbers for centuries (for example, "twenty-first century".) Both the **%Ey** and **%y** can then be viewed as the offset from **%EC** and **%C**, respectively.

The **E** and **O** modifiers modify the traditional conversion specifiers, so that they can always be used, even if the implementation (or the current locale) does not support the modifier.

The **E** modifier supports alternative date formats, such as the Japanese Emperor's Era, as long as these are based on the Gregorian calendar system. Extending the **E** modifiers to other date elements may provide an implementation-defined extension capable of supporting other calendar systems, especially in combination with the **O** modifier.

The **O** modifier supports time and date formats using the locale's alternative numerical symbols, such as Kanji or Hindi digits or ordinal number representation.

Non-European locales, whether they use Latin digits in computational items or not, often have local forms of the digits for use in date formats. This is not totally unknown even in Europe; a variant of dates uses Roman numerals for the months: the third day of September 1991 would be written as 3.IX.1991. In Japan, Kanji digits are regularly used for dates; in Arabic-speaking countries, Hindi digits are used. The **%d**, **%e**, **%H**, **%I**, **%m**, **%S**, **%U**, **%w**, **%W**, and **%y** conversion

date(P)

date(P)

specifications always return the date and time field in Latin digits (that is, 0 to 9). The **%O** modifier was introduced to support the use for display purposes of non-Latin digits. In the *LC_TIME* category in *localedef*, the optional **alt_digits** keyword is intended for this purpose. As an example, assume the following (partial) *localedef* source:

```
alt_digits "";"I";"II";"III";"IV";"V";"VI";"VII";"VIII" \  
          "IX";"X";"XI";"XII"  
d_fmt     "%e.%Om.%Y"
```

With the above date, the command:

```
date "+%x"
```

would yield 3.IX.1991. With the same **d_fmt**, but without the **alt_digits**, the command would yield 3.9.1991.

EXAMPLES

The following are input/output examples of *date* used at arbitrary times in the POSIX locale:

```
$ date  
Tue Jun 26 09:58:10 PDT 1990
```

```
$ date "+DATE: %m/%d/%y%nTIME: %H:%M:%S"  
DATE: 11/02/91  
TIME: 13:36:16
```

```
$ date "+TIME: %r"  
TIME: 01:36:32 PM
```

Examples for Denmark, where the default date and time format is **%a %d %b %Y %T %Z** :

```
$ LANG=da_DK.iso_8859-1 date  
ons 02 okt 1991 15:03:32 CET
```

```
$ LANG=da_DK.iso_8859-1 \  
date "+DATO: %A den %e. %B %Y%nKLOKKEN: %H:%M:%S"  
DATO: onsdag den 2. oktober 1991  
KLOKKEN: 15:03:56
```

Examples for Germany, where the default date and time format is **%a %d . %h . %Y , %T %Z** :

```
$ LANG=De_DE.88591 date  
Mi 02.Okt.1991, 15:01:21 MEZ
```

```
$ LANG=De_DE.88591 date "+DATUM: %A, %d. %B %Y%nZEIT: %H:%M:%S"  
DATUM: Mittwoch, 02. Oktober 1991  
ZEIT: 15:02:02
```

Examples for France, where the default date and time format is **%a %d %h %Y %Z %T** :

```
$ LANG=Fr_FR.88591 date  
Mer 02 oct 1991 MET 15:03:32
```

```
$ LANG=Fr_FR.88591 date "+JOUR: %A %d %B %Y%nHEURE: %H:%M:%S"
JOUR: Mercredi 02 octobre 1991
HEURE: 15:03:56
```

RATIONALE

Some of the new options for formatting are from the ISO C standard. The **-u** option was introduced to allow portable access to Coordinated Universal Time (UTC). The string **"GMT0"** is allowed as an equivalent *TZ* value to be compatible with all of the systems using the BSD implementation, where this option originated.

The **%e** format conversion specification (adopted from System V) was added because the ISO C standard conversion specifications did not provide any way to produce the historical default *date* output during the first nine days of any month.

There are two varieties of day and week numbering supported (in addition to any others created with the locale-dependent **%E** and **%O** modifier characters):

The historical variety in which Sunday is the first day of the week and the weekdays preceding the first Sunday of the year are considered week 0. These are represented by **%w** and **%U**. A variant of this is **%W**, using Monday as the first day of the week, but still referring to week 0. This view of the calendar was retained because so many historical applications depend on it and the ISO C standard *strftime()* function, on which many *date* implementations are based, was defined in this way.

The international standard, based on the ISO 8601:2000 standard where Monday is the first weekday and the algorithm for the first week number is more complex: If the week (Monday to Sunday) containing January 1 has four or more days in the new year, then it is week 1; otherwise, it is week 53 of the previous year, and the next week is week 1. These are represented by the new conversion specifications **%u** and **%V**, added as a result of international comments.

FUTURE DIRECTIONS

None.

SEE ALSO

The System Interfaces volume of IEEE Std 1003.1-2001, *printf()*, *strftime()*

COPYRIGHT

Portions of this text are reprinted and reproduced in electronic form from IEEE Std 1003.1, 2003 Edition, Standard for Information Technology -- Portable Operating System Interface (POSIX), The Open Group Base Specifications Issue 6, Copyright (C) 2001-2003 by the Institute of Electrical and Electronics Engineers, Inc and The Open Group. In the event of any discrepancy between this version and the original IEEE and The Open Group Standard, the original IEEE and The Open Group Standard is the referee document. The original Standard can be obtained online at <http://www.open-group.org/unix/online.html>.

NAME

dd - convert and copy a file

SYNOPSIS

dd [*operand ...*]

DESCRIPTION

The *dd* utility shall copy the specified input file to the specified output file with possible conversions using specific input and output block sizes. It shall read the input one block at a time, using the specified input block size; it shall then process the block of data actually returned, which could be smaller than the requested block size. It shall apply any conversions that have been specified and write the resulting data to the output in blocks of the specified output block size. If the **bs= *expr*** operand is specified and no conversions other than **sync**, **noerror**, or **notrunc** are requested, the data returned from each input block shall be written as a separate output block; if the read returns less than a full block and the **sync** conversion is not specified, the resulting output block shall be the same size as the input block. If the **bs= *expr*** operand is not specified, or a conversion other than **sync**, **noerror**, or **notrunc** is requested, the input shall be processed and collected into full-sized output blocks until the end of the input is reached.

The processing order shall be as follows:

An input block is read.

If the input block is shorter than the specified input block size and the **sync** conversion is specified, null bytes shall be appended to the input data up to the specified size. (If either **block** or **unblock** is also specified, <space>s shall be appended instead of null bytes.) The remaining conversions and output shall include the pad characters as if they had been read from the input.

If the **bs= *expr*** operand is specified and no conversion other than **sync** or **noerror** is requested, the resulting data shall be written to the output as a single block, and the remaining steps are omitted.

If the **swab** conversion is specified, each pair of input data bytes shall be swapped. If there is an odd number of bytes in the input block, the last byte in the input record shall not be swapped.

Any remaining conversions (**block**, **unblock**, **lcase**, and **ucase**) shall be performed. These conversions shall operate on the input data independently of the input blocking; an input or output fixed-length record may span block boundaries.

The data resulting from input or conversion or both shall be aggregated into output blocks of the specified size. After the end of input is reached, any remaining output shall be written as a block without padding if **conv= sync** is not specified; thus, the final output block may be shorter than the output block size.

OPTIONS

None.

OPERANDS

All of the operands shall be processed before any input is read. The following operands shall be supported:

if=*file* Specify the input pathname; the default is standard input.

of=*file* Specify the output pathname; the default is standard output. If the **seek= *expr*** conversion is not also specified, the output file shall be truncated before the copy begins if an explicit **of= *file*** operand is specified, unless **conv= notrunc** is specified. If **seek= *expr*** is specified, but **conv= notrunc** is not, the effect of the copy shall be to preserve the blocks in the output file over which *dd* seeks, but no other portion of the output file shall be preserved. (If the size of the seek plus the size of the input file is less than the previous size of the output file, the output file shall be shortened by the copy.)

ibs=*expr*

Specify the input block size, in bytes, by *expr* (default is 512).

obs=*expr*

Specify the output block size, in bytes, by *expr* (default is 512).

bs=expr

Set both input and output block sizes to *expr* bytes, superseding **ibs=** and **obs=**. If no conversion other than **sync**, **noerror**, and **notrunc** is specified, each input block shall be copied to the output as a single block without aggregating short blocks.

cbs=expr

Specify the conversion block size for **block** and **unblock** in bytes by *expr* (default is zero). If **cbs=** is omitted or given a value of zero, using **block** or **unblock** produces unspecified results.

The application shall ensure that this operand is also specified if the **conv=** operand is specified with a value of **ascii**, **ebcdic**, or **ibm**. For a **conv=** operand with an **ascii** value, the input is handled as described for the **unblock** value, except that characters are converted to ASCII before any trailing <space>s are deleted. For **conv=** operands with **ebcdic** or **ibm** values, the input is handled as described for the **block** value except that the characters are converted to EBCDIC or IBM EBCDIC, respectively, after any trailing <space>s are added.

skip=n Skip *n* input blocks (using the specified input block size) before starting to copy. On seekable files, the implementation shall read the blocks or seek past them; on non-seekable files, the blocks shall be read and the data shall be discarded.

seek=n Skip *n* blocks (using the specified output block size) from the beginning of the output file before copying. On non-seekable files, existing blocks shall be read and space from the current end-of-file to the specified offset, if any, filled with null bytes; on seekable files, the implementation shall seek to the specified offset or read the blocks as described for non-seekable files.

count=n

Copy only *n* input blocks.

conv=value[,value ...]

Where *values* are comma-separated symbols from the following list:

ascii Convert EBCDIC to ASCII; see ASCII to EBCDIC Conversion .

ebcdic Convert ASCII to EBCDIC; see ASCII to EBCDIC Conversion .

ibm Convert ASCII to a different EBCDIC set; see ASCII to IBM EBCDIC Conversion .

The **ascii**, **ebcdic**, and **ibm** values are mutually-exclusive.

block Treat the input as a sequence of <newline>-terminated or end-of-file-terminated variable-length records independent of the input block boundaries. Each record shall be converted to a record with a fixed length specified by the conversion block size. Any <newline> shall be removed from the input line; <space>s shall be appended to lines that are shorter than their conversion block size to fill the block. Lines that are longer than the conversion block size shall be truncated to the largest number of characters that fit into that size; the number of truncated lines shall be reported (see the STDERR section).

The **block** and **unblock** values are mutually-exclusive.

unblock

Convert fixed-length records to variable length. Read a number of bytes equal to the conversion block size (or the number of bytes remaining in the input, if less than the conversion block size), delete all trailing <space>s, and append a <newline>.

lcase Map uppercase characters specified by the *LC_CTYPE* keyword **tolower** to the corresponding lowercase character. Characters for which no mapping is specified shall not be modified by this conversion.

The **lcase** and **ucase** symbols are mutually-exclusive.

ucase Map lowercase characters specified by the *LC_CTYPE* keyword **toupper** to the corresponding uppercase character. Characters for which no mapping is specified shall not be modified by this conversion.

swab Swap every pair of input bytes.

noerror

Do not stop processing on an input error. When an input error occurs, a diagnostic message shall be written on standard error, followed by the current input and output block counts in the same format as used at completion (see the **STDERR** section). If the **sync** conversion is specified, the missing input shall be replaced with null bytes and processed normally; otherwise, the input block shall be omitted from the output.

notrunc

Do not truncate the output file. Preserve blocks in the output file not explicitly written by this invocation of the *dd* utility. (See also the preceding **of=** *file* operand.)

sync Pad every input block to the size of the **ibs=** buffer, appending null bytes. (If either **block** or **unblock** is also specified, append <space>s, rather than null bytes.)

The behavior is unspecified if operands other than **conv=** are specified more than once.

For the **bs=**, **cbs=**, **ibs=**, and **obs=** operands, the application shall supply an expression specifying a size in bytes. The expression, *expr*, can be:

A positive decimal number

A positive decimal number followed by *k*, specifying multiplication by 1024

A positive decimal number followed by *b*, specifying multiplication by 512

Two or more positive decimal numbers (with or without *k* or *b*) separated by *x*, specifying the product of the indicated values

All of the operands are processed before any input is read.

The following two tables display the octal number character values used for the **ascii** and **ebcdic** conversions (first table) and for the **ibm** conversion (second table). In both tables, the ASCII values are the row and column headers and the EBCDIC values are found at their intersections. For example, ASCII 0012 (LF) is the second row, third column, yielding 0045 in EBCDIC. The inverted tables (for EBCDIC to ASCII conversion) are not shown, but are in one-to-one correspondence with these tables. The differences between the two tables are highlighted by small boxes drawn around five entries.

Table: ASCII to EBCDIC Conversion**Table: ASCII to IBM EBCDIC Conversion****STDIN**

If no **if=** operand is specified, the standard input shall be used. See the **INPUT FILES** section.

INPUT FILES

The input file can be any file type.

ENVIRONMENT VARIABLES

The following environment variables shall affect the execution of *dd*:

LANG Provide a default value for the internationalization variables that are unset or null. (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2, Internationalization Variables for the precedence of internationalization variables used to determine the values of locale categories.)

LC_ALL

If set to a non-empty string value, override the values of all the other internationalization variables.

LC_CTYPE

Determine the locale for the interpretation of sequences of bytes of text data as characters (for example, single-byte as opposed to multi-byte characters in arguments and input files), the classification of characters as uppercase or lowercase, and the mapping of characters from one case to the other.

LC_MESSAGES

Determine the locale that should be used to affect the format and contents of diagnostic messages written to standard error and informative messages written to standard output.

NLSPATH

Determine the location of message catalogs for the processing of *LC_MESSAGES* .

ASYNCHRONOUS EVENTS

For SIGINT, the *dd* utility shall interrupt its current processing, write status information to standard error, and exit as though terminated by SIGINT. It shall take the standard action for all other signals; see the ASYNCHRONOUS EVENTS section in *Utility Description Defaults* .

STDOUT

If no **of=** operand is specified, the standard output shall be used. The nature of the output depends on the operands selected.

STDERR

On completion, *dd* shall write the number of input and output blocks to standard error. In the POSIX locale the following formats shall be used:

"%u+%u records in\n", *<number of whole input blocks>*,
<number of partial input blocks>

"%u+%u records out\n", *<number of whole output blocks>*,
<number of partial output blocks>

A partial input block is one for which *read()* returned less than the input block size. A partial output block is one that was written with fewer bytes than specified by the output block size.

In addition, when there is at least one truncated block, the number of truncated blocks shall be written to standard error. In the POSIX locale, the format shall be:

"%u truncated %s\n", *<number of truncated blocks>*, **"record"** (if
<number of truncated blocks> is one) **"records"** (otherwise)

Diagnostic messages may also be written to standard error.

OUTPUT FILES

If the **of=** operand is used, the output shall be the same as described in the STDOUT section.

EXTENDED DESCRIPTION

None.

EXIT STATUS

The following exit values shall be returned:

- 0 The input file was copied successfully.
- >0 An error occurred.

CONSEQUENCES OF ERRORS

If an input error is detected and the **noerror** conversion has not been specified, any partial output block shall be written to the output file, a diagnostic message shall be written, and the copy operation shall be discontinued. If some other error is detected, a diagnostic message shall be written and the copy operation shall be discontinued.

The following sections are informative.

APPLICATION USAGE

The input and output block size can be specified to take advantage of raw physical I/O.

There are many different versions of the EBCDIC codesets. The ASCII and EBCDIC conversions specified for the *dd* utility perform conversions for the version specified by the tables.

EXAMPLES

The following command:

dd if=/dev/rmt0h of=/dev/rmt1h

copies from tape drive 0 to tape drive 1, using a common historical device naming convention.

The following command:

dd ibs=10 skip=1

strips the first 10 bytes from standard input.

This example reads an EBCDIC tape blocked ten 80-byte EBCDIC card images per block into the ASCII file **x**:

dd if=/dev/tape of=x ibs=800 cbs=80 conv=ascii,lcase

RATIONALE

The **OPTIONS** section is listed as "None" because there are no options recognized by historical *dd* utilities. Certainly, many of the operands could have been designed to use the Utility Syntax Guidelines, which would have resulted in the classic hyphenated option letters. In this version of this volume of IEEE Std 1003.1-2001, *dd* retains its curious JCL-like syntax due to the large number of applications that depend on the historical implementation.

A suggested implementation technique for **conv= noerror, sync** is to zero (or <space>-fill, if **blocking** or **unblocking**) the input buffer before each read and to write the contents of the input buffer to the output even after an error. In this manner, any data transferred to the input buffer before the error was detected is preserved. Another point is that a failed read on a regular file or a disk generally does not increment the file offset, and *dd* must then seek past the block on which the error occurred; otherwise, the input error occurs repetitively. When the input is a magnetic tape, however, the tape normally has passed the block containing the error when the error is reported, and thus no seek is necessary.

The default **ibs=** and **obs=** sizes are specified as 512 bytes because there are historical (largely portable) scripts that assume these values. If they were left unspecified, unusual results could occur if an implementation chose an odd block size.

Historical implementations of *dd* used *creat()* when processing **of= file**. This makes the **seek=** operand unusable except on special files. The **conv= notrunc** feature was added because more recent BSD-based implementations use *open()* (without **O_TRUNC**) instead of *creat()*, but they fail to delete output file contents after the data copied.

The *w* multiplier (historically meaning *word*), is used in System V to mean 2 and in 4.2 BSD to mean 4. Since *word* is inherently non-portable, its use is not supported by this volume of IEEE Std 1003.1-2001.

Standard EBCDIC does not have the characters '[' and ']'. The values used in the table are taken from a common print train that does contain them. Other than those characters, the print train values are not filled in, but appear to provide some of the motivation for the historical choice of translations reflected here.

The Standard EBCDIC table provides a 1:1 translation for all 256 bytes.

The IBM EBCDIC table does not provide such a translation. The marked cells in the tables differ in such a way that:

EBCDIC 0112 ('**cent**') and 0152 (broken pipe) do not appear in the table.

EBCDIC 0137 ('**not**') translates to/from ASCII 0236 ('""'). In the standard table, EBCDIC 0232 (no graphic) is used.

EBCDIC 0241 ('""') translates to/from ASCII 0176 ('""'). In the standard table, EBCDIC 0137 ('**not**') is used.

0255 ('[') and 0275 (']') appear twice, once in the same place as for the standard table and once in place of 0112 ('**cent**') and 0241 ('""').

In net result: EBCDIC 0275 (']') displaced EBCDIC 0241 ('""') in cell 0345.

That displaced EBCDIC 0137 ('**not**') in cell 0176.

That displaced EBCDIC 0232 (no graphic) in cell 0136.

That replaced EBCDIC 0152 (broken pipe) in cell 0313.

EBCDIC 0255 ('[') replaced EBCDIC 0112 ('cent').

This translation, however, reflects historical practice that (ASCII) '~' and 'not' were often mapped to each other, as were '[' and 'cent' ; and ']' and (EBCDIC) '~' .

The **cbs** operand is required if any of the **ascii**, **ebcdic**, or **ibm** operands are specified. For the **ascii** operand, the input is handled as described for the **unblock** operand except that characters are converted to ASCII before the trailing <space>s are deleted. For the **ebcdic** and **ibm** operands, the input is handled as described for the **block** operand except that the characters are converted to EBCDIC or IBM EBCDIC after the trailing <space>s are added.

The **block** and **unblock** keywords are from historical BSD practice.

The consistent use of the word **record** in standard error messages matches most historical practice. An earlier version of System V used **block**, but this has been updated in more recent releases.

Early proposals only allowed two numbers separated by **x** to be used in a product when specifying **bs=**, **cbs=**, **ibs=**, and **obs=** sizes. This was changed to reflect the historical practice of allowing multiple numbers in the product as provided by Version 7 and all releases of System V and BSD.

A change to the **swab** conversion is required to match historical practice and is the result of IEEE PASC Interpretations 1003.2 #03 and #04, submitted for the ISO POSIX-2:1993 standard.

A change to the handling of SIGINT is required to match historical practice and is the result of IEEE PASC Interpretation 1003.2 #06 submitted for the ISO POSIX-2:1993 standard.

FUTURE DIRECTIONS

None.

SEE ALSO

Utility Description Defaults , *sed* , *tr*


COPYRIGHT

Portions of this text are reprinted and reproduced in electronic form from IEEE Std 1003.1, 2003 Edition, Standard for Information Technology -- Portable Operating System Interface (POSIX), The Open Group Base Specifications Issue 6, Copyright (C) 2001-2003 by the Institute of Electrical and Electronics Engineers, Inc and The Open Group. In the event of any discrepancy between this version and the original IEEE and The Open Group Standard, the original IEEE and The Open Group Standard is the referee document. The original Standard can be obtained online at <http://www.open-group.org/unix/online.html> .


NAME

df - report free disk space

SYNOPSIS

df [-k][-P|-t][*file*...]

DESCRIPTION

The *df* utility shall write the amount of available space  and file slots for file systems on which the invoking user has appropriate read access. File systems shall be specified by the *file* operands; when none are specified, information shall be written for all file systems. The format of the default output from *df* is unspecified, but all space figures are reported in 512-byte units, unless the **-k** option is specified. This output shall contain at least the file system names, amount of available space on each of these file systems, and the number of free file slots, or *inodes*, available; when **-t** is specified, the output shall contain the total allocated space as well.

OPTIONS

The *df* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section 12.2, Utility Syntax Guidelines.

The following options shall be supported:

- k** Use 1024-byte units, instead of the default 512-byte units, when writing space figures.
- P** Produce output in the format described in the STDOUT section.
- t** Include total allocated-space figures in the output.

OPERANDS

The following operand shall be supported:

- file* A pathname of a file within the hierarchy of the desired file system. If a file other than a FIFO, a regular file, a directory, or a special file representing the device containing the file system (for example, **/dev/dsk/0s1**) is specified, the results are unspecified. Otherwise, *df* shall write the amount of free space in the file system containing the specified *file* operand.

STDIN

Not used.

INPUT FILES

None.

ENVIRONMENT VARIABLES

The following environment variables shall affect the execution of *df*:

LANG Provide a default value for the internationalization variables that are unset or null. (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2, Internationalization Variables for the precedence of internationalization variables used to determine the values of locale categories.)

LC_ALL

If set to a non-empty string value, override the values of all the other internationalization variables.

LC_CTYPE

Determine the locale for the interpretation of sequences of bytes of text data as characters (for example, single-byte as opposed to multi-byte characters in arguments).

LC_MESSAGES

Determine the locale that should be used to affect the format and contents of diagnostic messages written to standard error and informative messages written to standard output.

NLSPATH

Determine the location of message catalogs for the processing of **LC_MESSAGES**.

ASYNCHRONOUS EVENTS

Default.

STDOUT

When both the **-k** and **-P** options are specified, the following header line shall be written (in the POSIX locale):

"Filesystem 1024-blocks Used Available Capacity Mounted on\n"

When the **-P** option is specified without the **-k** option, the following header line shall be written (in the POSIX locale):

"Filesystem 512-blocks Used Available Capacity Mounted on\n"

The implementation may adjust the spacing of the header line and the individual data lines so that the information is presented in orderly columns.

The remaining output with **-P** shall consist of one line of information for each specified file system. These lines shall be formatted as follows:

```
"%s %d %d %d %d%% %s\n", <file system name>, <total space>,
    <space used>, <space free>, <percentage used>,
    <file system root>
```

In the following list, all quantities expressed in 512-byte units (1024-byte when **-k** is specified) shall be rounded up to the next higher unit. The fields are:

<file system name>

The name of the file system, in an implementation-defined format.

<total space>

The total size of the file system in 512-byte units. The exact meaning of this figure is implementation-defined, but should include *<space used>*, *<space free>*, plus any space reserved by the system not normally available to a user.

<space used>

The total amount of space allocated to existing files in the file system, in 512-byte units.

<space free>

The total amount of space available within the file system for the creation of new files by unprivileged users, in 512-byte units. When this figure is less than or equal to zero, it shall not be possible to create any new files on the file system without first deleting others, unless the process has appropriate privileges. The figure written may be less than zero.

<percentage used>

The percentage of the normally available space that is currently allocated to all files on the file system. This shall be calculated using the fraction:

$$\frac{\text{<space used>}}{(\text{<space used>} + \text{<space free>})}$$

expressed as a percentage. This percentage may be greater than 100 if *<space free>* is less than zero. The percentage value shall be expressed as a positive integer, with any fractional result causing it to be rounded to the next highest integer.

<file system root>

The directory below which the file system hierarchy appears.

The output format is unspecified when **-t** is used.

STDERR

The standard error shall be used only for diagnostic messages.

OUTPUT FILES

None.

EXTENDED DESCRIPTION

None.

EXIT STATUS

The following exit values shall be returned:

- 0 Successful completion.
- >0 An error occurred.

CONSEQUENCES OF ERRORS

Default.

The following sections are informative.

APPLICATION USAGE

On most systems, the "name of the file system, in an implementation-defined format" is the special file on which the file system is mounted.

On large file systems, the calculation specified for percentage used can create huge rounding errors.

EXAMPLES

The following example writes portable information about the **/usr** file system:

df -P /usr

Assuming that **/usr/src** is part of the **/usr** file system, the following produces the same output as the previous example:

df -P /usr/src

RATIONALE

The behavior of *df* with the **-P** option is the default action of the 4.2 BSD *df* utility. The uppercase **-P** was selected to avoid collision with a known industry extension using **-p**.

Historical *df* implementations vary considerably in their default output. It was therefore necessary to describe the default output in a loose manner to accommodate all known historical implementations and to add a portable option (**-P**) to provide information in a portable format.

The use of 512-byte units is historical practice and maintains compatibility with *ls* and other utilities in this volume of IEEE Std 1003.1-2001. This does not mandate that the file system itself be based on 512-byte blocks. The **-k** option was added as a compromise measure. It was agreed by the standard developers that 512 bytes was the best default unit because of its complete historical consistency on System V (*versus* the mixed 512/1024-byte usage on BSD systems), and that a **-k** option to switch to 1024-byte units was a good compromise. Users who prefer the more logical 1024-byte quantity can easily alias *df* to *df -k* without breaking many historical scripts relying on the 512-byte units.

It was suggested that *df* and the various related utilities be modified to access a *BLOCKSIZE* environment variable to achieve consistency and user acceptance. Since this is not historical practice on any system, it is left as a possible area for system extensions and will be re-evaluated in a future version if it is widely implemented.

FUTURE DIRECTIONS

None.

SEE ALSO

find

COPYRIGHT

Portions of this text are reprinted and reproduced in electronic form from IEEE Std 1003.1, 2003 Edition, Standard for Information Technology -- Portable Operating System Interface (POSIX), The Open Group Base Specifications Issue 6, Copyright (C) 2001-2003 by the Institute of Electrical and Electronics Engineers, Inc and The Open Group. In the event of any discrepancy between this version and the original IEEE and The Open Group Standard, the original IEEE and The Open Group Standard is the referee document. The original Standard can be obtained online at <http://www.open-group.org/unix/online.html> .

NAME

dirname - return the directory portion of a pathname

SYNOPSIS

dirname *string*

DESCRIPTION

The *string* operand shall be treated as a pathname, as defined in the Base Definitions volume of IEEE Std 1003.1-2001, Section 3.266, Pathname. The string *string* shall be converted to the name of the directory containing the filename corresponding to the last pathname component in *string*, performing actions equivalent to the following steps in order:

If *string* is *//*, skip steps 2 to 5.

If *string* consists entirely of slash characters, *string* shall be set to a single slash character. In this case, skip steps 3 to 8.

If there are any trailing slash characters in *string*, they shall be removed.

If there are no slash characters remaining in *string*, *string* shall be set to a single period character. In this case, skip steps 5 to 8.

If there are any trailing non-slash characters in *string*, they shall be removed.

If the remaining *string* is *//*, it is implementation-defined whether steps 7 and 8 are skipped or processed.

If there are any trailing slash characters in *string*, they shall be removed.

If the remaining *string* is empty, *string* shall be set to a single slash character.

The resulting string shall be written to standard output.

OPTIONS

None.

OPERANDS

The following operand shall be supported:

string A string.

STDIN

Not used.

INPUT FILES

None.

ENVIRONMENT VARIABLES

The following environment variables shall affect the execution of *dirname*:

LANG Provide a default value for the internationalization variables that are unset or null. (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2, Internationalization Variables for the precedence of internationalization variables used to determine the values of locale categories.)

LC_ALL

If set to a non-empty string value, override the values of all the other internationalization variables.

LC_CTYPE

Determine the locale for the interpretation of sequences of bytes of text data as characters (for example, single-byte as opposed to multi-byte characters in arguments).

LC_MESSAGES

Determine the locale that should be used to affect the format and contents of diagnostic messages written to standard error.

dirname(P)

dirname(P)

NLSPATH

Determine the location of message catalogs for the processing of *LC_MESSAGES*.

ASYNCHRONOUS EVENTS

Default.

STDOUT

The *dirname* utility shall write a line to the standard output in the following format:

"%s\n", <resulting string>

STDERR

The standard error shall be used only for diagnostic messages.

OUTPUT FILES

None.

EXTENDED DESCRIPTION

None.

EXIT STATUS

The following exit values shall be returned:

0	Successful completion.
>0	An error occurred.

CONSEQUENCES OF ERRORS

Default.

The following sections are informative.

APPLICATION USAGE

The definition of *pathname* specifies implementation-defined behavior for pathnames starting with two slash characters. Therefore, applications shall not arbitrarily add slashes to the beginning of a pathname unless they can ensure that there are more or less than two or are prepared to deal with the implementation-defined consequences.

EXAMPLES

Command	Results
<i>dirname /</i>	/
<i>dirname //</i>	/ or //
<i>dirname /a/b/</i>	/a
<i>dirname //a//b//</i>	//a
<i>dirname</i>	Unspecified
<i>dirname a</i>	.(\$? = 0)
<i>dirname ""</i>	.(\$? = 0)
<i>dirname /a</i>	/
<i>dirname /a/b</i>	/a
<i>dirname a/b</i>	a

RATIONALE

The *dirname* utility originated in System III. It has evolved through the System V releases to a version that matches the requirements specified in this description in System V Release 3. 4.3 BSD and earlier versions did not include *dirname*.

The behaviors of *basename* and *dirname* in this volume of IEEE Std 1003.1-2001 have been coordinated so that when *string* is a valid pathname:

\$(basename "*string*")

would be a valid filename for the file in the directory:

\$(dirname "string")

This would not work for the versions of these utilities in early proposals due to the way processing of trailing slashes was specified. Consideration was given to leaving processing unspecified if there were trailing slashes, but this cannot be done; the Base Definitions volume of IEEE Std 1003.1-2001, Section 3.266, Pathname allows trailing slashes. The *basename* and *dirname* utilities have to specify consistent handling for all valid pathnames.

FUTURE DIRECTIONS

None.

SEE ALSO

basename() , *Parameters and Variables*

COPYRIGHT

Portions of this text are reprinted and reproduced in electronic form from IEEE Std 1003.1, 2003 Edition, Standard for Information Technology -- Portable Operating System Interface (POSIX), The Open Group Base Specifications Issue 6, Copyright (C) 2001-2003 by the Institute of Electrical and Electronics Engineers, Inc and The Open Group. In the event of any discrepancy between this version and the original IEEE and The Open Group Standard, the original IEEE and The Open Group Standard is the referee document. The original Standard can be obtained online at <http://www.open-group.org/unix/online.html> .

NAME

du - estimate file space usage

SYNOPSIS

du [-a | -s][**-kx**][**-H** | **-L**][*file ...*]

DESCRIPTION

By default, the *du* utility shall write to standard output the size of the file space allocated to, and the size of the file space allocated to each subdirectory of, the file hierarchy rooted in each of the specified files. By default, when a symbolic link is encountered on the command line or in the file hierarchy, *du* shall count the size of the symbolic link (rather than the file referenced by the link), and shall not follow the link to another portion of the file hierarchy. The size of the file space allocated to a file of type directory shall be defined as the sum total of space allocated to all files in the file hierarchy rooted in the directory plus the space allocated to the directory itself.

When *du* cannot *stat()* files or *stat()* or read directories, it shall report an error condition and the final exit status is affected. Files with multiple links shall be counted and written for only one entry. The directory entry that is selected in the report is unspecified. By default, file sizes shall be written in 512-byte units, rounded up to the next 512-byte unit.

OPTIONS

The *du* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section 12.2, Utility Syntax Guidelines.

The following options shall be supported:

- a** In addition to the default output, report the size of each file not of type directory in the file hierarchy rooted in the specified file. Regardless of the presence of the **-a** option, non-directories given as *file* operands shall always be listed.
- H** If a symbolic link is specified on the command line, *du* shall count the size of the file or file hierarchy referenced by the link.
- k** Write the files sizes in units of 1024 bytes, rather than the default 512-byte units.
- L** If a symbolic link is specified on the command line or encountered during the traversal of a file hierarchy, *du* shall count the size of the file or file hierarchy referenced by the link.
- s** Instead of the default output, report only the total sum for each of the specified files.
- x** When evaluating file sizes, evaluate only those files that have the same device as the file specified by the *file* operand.

Specifying more than one of the mutually-exclusive options **-H** and **-L** shall not be considered an error. The last option specified shall determine the behavior of the utility.

OPERANDS

The following operand shall be supported:

- file* The pathname of a file whose size is to be written. If no *file* is specified, the current directory shall be used.

STDIN

Not used.

INPUT FILES

None.

ENVIRONMENT VARIABLES

The following environment variables shall affect the execution of *du*:

- LANG** Provide a default value for the internationalization variables that are unset or null. (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2, Internationalization Variables for the precedence of internationalization variables used to determine the values of locale categories.)

LC_ALL

If set to a non-empty string value, override the values of all the other internationalization variables.

LC_CTYPE

Determine the locale for the interpretation of sequences of bytes of text data as characters (for example, single-byte as opposed to multi-byte characters in arguments).

LC_MESSAGES

Determine the locale that should be used to affect the format and contents of diagnostic messages written to standard error.

NLSPATH

Determine the location of message catalogs for the processing of *LC_MESSAGES*.

ASYNCHRONOUS EVENTS

Default.

STDOUT

The output from *du* shall consist of the amount of space allocated to a file and the name of the file, in the following format:

```
"%d %s\n", <size>, <pathname>
```

STDERR

The standard error shall be used only for diagnostic messages.

OUTPUT FILES

None.

EXTENDED DESCRIPTION

None.

EXIT STATUS

The following exit values shall be returned:

- 0 Successful completion.
- >0 An error occurred.

CONSEQUENCES OF ERRORS

Default.

The following sections are informative.

APPLICATION USAGE

None.

EXAMPLES

None.

RATIONALE

The use of 512-byte units is historical practice and maintains compatibility with *ls* and other utilities in this volume of IEEE Std 1003.1-2001. This does not mandate that the file system itself be based on 512-byte blocks. The **-k** option was added as a compromise measure. It was agreed by the standard developers that 512 bytes was the best default unit because of its complete historical consistency on System V (*versus* the mixed 512/1024-byte usage on BSD systems), and that a **-k** option to switch to 1024-byte units was a good compromise. Users who prefer the 1024-byte quantity can easily alias *du* to *du -k* without breaking the many historical scripts relying on the 512-byte units.

The **-b** option was added to an early proposal to provide a resolution to the situation where System V and BSD systems give figures for file sizes in *blocks*, which is an implementation-defined concept. (In common usage, the block size is 512 bytes for System V and 1024 bytes for BSD systems.) However, **-b** was later deleted, since the default was eventually decided as 512-byte units.

Historical file systems provided no way to obtain exact figures for the space allocation given to files.

There are two known areas of inaccuracies in historical file systems: cases of *indirect blocks* being used by the file system or *sparse* files yielding incorrectly high values. An indirect block is space used by the file system in the storage of the file, but that need not be counted in the space allocated to the file. A *sparse* file is one in which an *lseek()* call has been made to a position beyond the end of the file and data has subsequently been written at that point. A file system need not allocate all the intervening zero-filled blocks to such a file. It is up to the implementation to define exactly how accurate its methods are.

The **-a** and **-s** options were mutually-exclusive in the original version of *du*. The POSIX Shell and Utilities description is implied by the language in the SVID where **-s** is described as causing "only the grand total" to be reported. Some systems may produce output for **-sa**, but a Strictly Conforming POSIX Shell and Utilities Application cannot use that combination.

The **-a** and **-s** options were adopted from the SVID except that the System V behavior of not listing non-directories explicitly given as operands, unless the **-a** option is specified, was considered a bug; the BSD-based behavior (report for all operands) is mandated. The default behavior of *du* in the SVID with regard to reporting the failure to read files (it produces no messages) was considered counter-intuitive, and thus it was specified that the POSIX Shell and Utilities default behavior shall be to produce such messages. These messages can be turned off with shell redirection to achieve the System V behavior.

The **-x** option is historical practice on recent BSD systems. It has been adopted by this volume of IEEE Std 1003.1-2001 because there was no other historical method of limiting the *du* search to a single file hierarchy. This limitation of the search is necessary to make it possible to obtain file space usage information about a file system on which other file systems are mounted, without having to resort to a lengthy *find* and *awk* script.

FUTURE DIRECTIONS

None.

SEE ALSO

ls, the System Interfaces volume of IEEE Std 1003.1-2001, *stat()*

COPYRIGHT

Portions of this text are reprinted and reproduced in electronic form from IEEE Std 1003.1, 2003 Edition, Standard for Information Technology -- Portable Operating System Interface (POSIX), The Open Group Base Specifications Issue 6, Copyright (C) 2001-2003 by the Institute of Electrical and Electronics Engineers, Inc and The Open Group. In the event of any discrepancy between this version and the original IEEE and The Open Group Standard, the original IEEE and The Open Group Standard is the referee document. The original Standard can be obtained online at <http://www.open-group.org/unix/online.html>.

NAME

echo - write arguments to standard output

SYNOPSIS

echo [*string* ...]

DESCRIPTION

The *echo* utility writes its arguments to standard output, followed by a <newline>. If there are no arguments, only the <newline> is written.

OPTIONS

The *echo* utility shall not recognize the "--" argument in the manner specified by Guideline 10 of the Base Definitions volume of IEEE Std 1003.1-2001, Section 12.2, Utility Syntax Guidelines; "--" shall be recognized as a string operand.

Implementations shall not support any options.

OPERANDS

The following operands shall be supported:

string A string to be written to standard output. If the first operand is **-n**, or if any of the operands contain a backslash ('\ ') character, the results are implementation-defined.

On XSI-conformant systems, if the first operand is **-n**, it shall be treated as a string, not an option. The following character sequences shall be recognized on XSI-conformant systems within any of the arguments:

- \a** Write an <alert>.
- \b** Write a <backspace>.
- \c** Suppress the <newline> that otherwise follows the final argument in the output. All characters following the '\c' in the arguments shall be ignored.
- \f** Write a <form-feed>.
- \n** Write a <newline>.
- \r** Write a <carriage-return>.
- \t** Write a <tab>.
- \v** Write a <vertical-tab>.
- ** Write a backslash character.
- \0num** Write an 8-bit value that is the zero, one, two, or three-digit octal number *num*.

STDIN

Not used.

INPUT FILES

None.

ENVIRONMENT VARIABLES

The following environment variables shall affect the execution of *echo*:

LANG Provide a default value for the internationalization variables that are unset or null. (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2, Internationalization Variables for the precedence of internationalization variables used to determine the values of locale categories.)

LC_ALL

If set to a non-empty string value, override the values of all the other internationalization variables.

LC_CTYPE

Determine the locale for the interpretation of sequences of bytes of text data as characters (for example, single-byte as opposed to multi-byte characters in arguments).

LC_MESSAGES

Determine the locale that should be used to affect the format and contents of diagnostic messages written to standard error.


NLSPATH

Determine the location of message catalogs for the processing of *LC_MESSAGES*.

ASYNCHRONOUS EVENTS

Default.

STDOUT

The *echo* utility arguments shall be separated by single <space>s and a <newline> shall follow the last argument. Output transformations shall occur based on the escape sequences in the input. See the OPERANDS section.  **Option End**

STDERR

The standard error shall be used only for diagnostic messages.

OUTPUT FILES

None.

EXTENDED DESCRIPTION

None.

EXIT STATUS

The following exit values shall be returned:

- 0 Successful completion.
- >0 An error occurred.

CONSEQUENCES OF ERRORS

Default.

The following sections are informative.

APPLICATION USAGE

It is not possible to use *echo* portably across all POSIX systems unless both **-n** (as the first argument) and escape sequences are omitted.

The *printf* utility can be used portably to emulate any of the traditional behaviors of the *echo* utility as follows (assuming that *IFS* has its standard value or is unset):

The historic System V *echo* and the requirements on XSI implementations in this volume of IEEE Std 1003.1-2001 are equivalent to:

```
printf "%b\n" "$*
```

The BSD *echo* is equivalent to:

```
if [ "X$1" = "X-n" ]
then
  shift
  printf "%s" "$*"
else
  printf "%s\n" "$*"
fi
```

New applications are encouraged to use *printf* instead of *echo*.

EXAMPLES

None.

RATIONALE

The *echo* utility has not been made obsolescent because of its extremely widespread use in historical applications. Conforming applications that wish to do prompting without <newline>s or that could possibly be expecting to echo a **-n**, should use the *printf* utility derived from the Ninth Edition system.

As specified, *echo* writes its arguments in the simplest of ways. The two different historical versions of *echo* vary in fatally incompatible ways.

The BSD *echo* checks the first argument for the string **-n** which causes it to suppress the <newline> that would otherwise follow the final argument in the output.

The System V *echo* does not support any options, but allows escape sequences within its operands, as described for XSI implementations in the OPERANDS section.

The *echo* utility does not support Utility Syntax Guideline 10 because historical applications depend on *echo* to echo *all* of its arguments, except for the **-n** option in the BSD version.

FUTURE DIRECTIONS

None.

SEE ALSO

printf

COPYRIGHT

Portions of this text are reprinted and reproduced in electronic form from IEEE Std 1003.1, 2003 Edition, Standard for Information Technology -- Portable Operating System Interface (POSIX), The Open Group Base Specifications Issue 6, Copyright (C) 2001-2003 by the Institute of Electrical and Electronics Engineers, Inc and The Open Group. In the event of any discrepancy between this version and the original IEEE and The Open Group Standard, the original IEEE and The Open Group Standard is the referee document. The original Standard can be obtained online at <http://www.open-group.org/unix/online.html>.

NAME

env - set the environment for command invocation

SYNOPSIS

env [-i][*name=value*]... [*utility* [*argument*...]]

DESCRIPTION

The *env* utility shall obtain the current environment, modify it according to its arguments, then invoke the utility named by the *utility* operand with the modified environment.

Optional arguments shall be passed to *utility*.

If no *utility* operand is specified, the resulting environment shall be written to the standard output, with one *name= value* pair per line.

OPTIONS

The *env* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section 12.2, Utility Syntax Guidelines.

The following options shall be supported:

- i** Invoke *utility* with exactly the environment specified by the arguments; the inherited environment shall be ignored completely.

OPERANDS

The following operands shall be supported:

name=value

Arguments of the form *name= value* shall modify the execution environment, and shall be placed into the inherited environment before the *utility* is invoked.

utility The name of the utility to be invoked. If the *utility* operand names any of the special built-in utilities in *Special Built-In Utilities*, the results are undefined.

argument

A string to pass as an argument for the invoked utility.

STDIN

Not used.

INPUT FILES

None.

ENVIRONMENT VARIABLES

The following environment variables shall affect the execution of *env*:

LANG Provide a default value for the internationalization variables that are unset or null. (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2, Internationalization Variables for the precedence of internationalization variables used to determine the values of locale categories.)

LC_ALL

If set to a non-empty string value, override the values of all the other internationalization variables.

LC_CTYPE

Determine the locale for the interpretation of sequences of bytes of text data as characters (for example, single-byte as opposed to multi-byte characters in arguments).

LC_MESSAGES

Determine the locale that should be used to affect the format and contents of diagnostic messages written to standard error.

NLSPATH

Determine the location of message catalogs for the processing of **LC_MESSAGES**.

PATH Determine the location of the *utility*, as described in the Base Definitions volume of IEEE Std 1003.1-2001, Chapter 8, Environment Variables. If *PATH* is specified as a *name=value* operand to *env*, the *value* given shall be used in the search for *utility*.

ASYNCHRONOUS EVENTS

Default.

STDOUT

If no *utility* operand is specified, each *name=value* pair in the resulting environment shall be written in the form:

`"%s=%s\n", <name>, <value>`

If the *utility* operand is specified, the *env* utility shall not write to standard output.

STDERR

The standard error shall be used only for diagnostic messages.

OUTPUT FILES

None.

EXTENDED DESCRIPTION

None.

EXIT STATUS

If *utility* is invoked, the exit status of *env* shall be the exit status of *utility*; otherwise, the *env* utility shall exit with one of the following values:

- 0 The *env* utility completed successfully.
- 1-125 An error occurred in the *env* utility.
- 126 The utility specified by *utility* was found but could not be invoked.
- 127 The utility specified by *utility* could not be found.

CONSEQUENCES OF ERRORS

Default.

The following sections are informative.

APPLICATION USAGE

The *command*, *env*, *nice*, *nohup*, *time*, and *xargs* utilities have been specified to use exit code 127 if an error occurs so that applications can distinguish "failure to find a utility" from "invoked utility exited with an error indication". The value 127 was chosen because it is not commonly used for other meanings; most utilities use small values for "normal error conditions" and the values above 128 can be confused with termination due to receipt of a signal. The value 126 was chosen in a similar manner to indicate that the utility could be found, but not invoked. Some scripts produce meaningful error messages differentiating the 126 and 127 cases. The distinction between exit codes 126 and 127 is based on KornShell practice that uses 127 when all attempts to *exec* the utility fail with [ENOENT], and uses 126 when any attempt to *exec* the utility fails for any other reason.

Historical implementations of the *env* utility use the *execvp()* or *execlp()* functions defined in the System Interfaces volume of IEEE Std 1003.1-2001 to invoke the specified utility; this provides better performance and keeps users from having to escape characters with special meaning to the shell. Therefore, shell functions, special built-ins, and built-ins that are only provided by the shell are not found.

EXAMPLES

The following command:

```
env -i PATH=/mybin mygrep xyz myfile
```

invokes the command *mygrep* with a new *PATH* value as the only entry in its environment. In this case, *PATH* is used to locate *mygrep*, which then must reside in **/mybin**.

RATIONALE

As with all other utilities that invoke other utilities, this volume of IEEE Std 1003.1-2001 only specifies what *env* does with standard input, standard output, standard error, input files, and output files. If a utility is executed, it is not constrained by the specification of input and output by *env*.

The **-i** option was added to allow the functionality of the withdrawn **-** option in a manner compatible with the Utility Syntax Guidelines.

Some have suggested that *env* is redundant since the same effect is achieved by:

name=value ... utility [argument ...]

The example is equivalent to *env* when an environment variable is being added to the environment of the command, but not when the environment is being set to the given value. The *env* utility also writes out the current environment if invoked without arguments. There is sufficient functionality beyond what the example provides to justify inclusion of *env*.

FUTURE DIRECTIONS

None.

SEE ALSO

Parameters and Variables , Special Built-In Utilities

COPYRIGHT

Portions of this text are reprinted and reproduced in electronic form from IEEE Std 1003.1, 2003 Edition, Standard for Information Technology -- Portable Operating System Interface (POSIX), The Open Group Base Specifications Issue 6, Copyright (C) 2001-2003 by the Institute of Electrical and Electronics Engineers, Inc and The Open Group. In the event of any discrepancy between this version and the original IEEE and The Open Group Standard, the original IEEE and The Open Group Standard is the referee document. The original Standard can be obtained online at <http://www.open-group.org/unix/online.html>.

NAME

expand - convert tabs to spaces

SYNOPSIS

expand [-t *tablist*][*file* ...]

DESCRIPTION

The *expand* utility shall write files or the standard input to the standard output with <tab>s replaced with one or more <space>s needed to pad to the next tab stop. Any <backspace>s shall be copied to the output and cause the column position count for tab stop calculations to be decremented; the column position count shall not be decremented below zero.

OPTIONS

The *expand* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section 12.2, Utility Syntax Guidelines.

The following option shall be supported:

-t *tablist*

Specify the tab stops. The application shall ensure that the argument *tablist* consists of either a single positive decimal integer or a list of tabstops. If a single number is given, tabs shall be set that number of column positions apart instead of the default 8.

If a list of tabstops is given, the application shall ensure that it consists of a list of two or more positive decimal integers, separated by <blank>s or commas, in ascending order. The tabs shall be set at those specific column positions. Each tab stop *N* shall be an integer value greater than zero, and the list is in strictly ascending order. This is taken to mean that, from the start of a line of output, tabbing to position *N* shall cause the next character output to be in the (*N*+1)th column position on that line.

In the event of *expand* having to process a <tab> at a position beyond the last of those specified in a multiple tab-stop list, the <tab> shall be replaced by a single <space> in the output.

OPERANDS

The following operand shall be supported:

file The pathname of a text file to be used as input.

STDIN

See the INPUT FILES section.

INPUT FILES

Input files shall be text files.

ENVIRONMENT VARIABLES

The following environment variables shall affect the execution of *expand*:

LANG Provide a default value for the internationalization variables that are unset or null. (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2, Internationalization Variables for the precedence of internationalization variables used to determine the values of locale categories.)

LC_ALL

If set to a non-empty string value, override the values of all the other internationalization variables.

LC_CTYPE

Determine the locale for the interpretation of sequences of bytes of text data as characters (for example, single-byte as opposed to multi-byte characters in arguments and input files), the processing of <tab>s and <space>s, and for the determination of the width in column positions each character would occupy on an output device.

LC_MESSAGES

Determine the locale that should be used to affect the format and contents of diagnostic messages written to standard error.

expand(P)

expand(P)

NLSPATH

Determine the location of message catalogs for the processing of *LC_MESSAGES* .

ASYNCHRONOUS EVENTS

Default.

STDOUT

The standard output shall be equivalent to the input files with <tab>s converted into the appropriate number of <space>s.

STDERR

The standard error shall be used only for diagnostic messages.

OUTPUT FILES

None.

EXTENDED DESCRIPTION

None.

EXIT STATUS

The following exit values shall be returned:

- | | |
|----|-----------------------|
| 0 | Successful completion |
| >0 | An error occurred. |

CONSEQUENCES OF ERRORS

The *expand* utility shall terminate with an error message and non-zero exit status upon encountering difficulties accessing one of the *file* operands.

The following sections are informative.

APPLICATION USAGE

None.

EXAMPLES

None.

RATIONALE

The *expand* utility is useful for preprocessing text files (before sorting, looking at specific columns, and so on) that contain <tab>s.

See the Base Definitions volume of IEEE Std 1003.1-2001, Section 3.103, Column Position.

The *tablist* option-argument consists of integers in ascending order. Utility Syntax Guideline 8 mandates that *expand* shall accept the integers (within the single argument) separated using either commas or <blank>s.

FUTURE DIRECTIONS

None.

SEE ALSO

tabs , *unexpand*

COPYRIGHT

Portions of this text are reprinted and reproduced in electronic form from IEEE Std 1003.1, 2003 Edition, Standard for Information Technology -- Portable Operating System Interface (POSIX), The Open Group Base Specifications Issue 6, Copyright (C) 2001-2003 by the Institute of Electrical and Electronics Engineers, Inc and The Open Group. In the event of any discrepancy between this version and the original IEEE and The Open Group Standard, the original IEEE and The Open Group Standard is the referee document. The original Standard can be obtained online at <http://www.open-group.org/unix/online.html> .

NAME

expr - evaluate arguments as an expression

SYNOPSIS

expr *operand*

DESCRIPTION

The *expr* utility shall evaluate an expression and write the result to standard output.

OPTIONS

None.

OPERANDS

The single expression evaluated by *expr* shall be formed from the operands, as described in the EXTENDED DESCRIPTION section. The application shall ensure that each of the expression operator symbols:

() | & = > >= < <= != + - * / % :

and the symbols *integer* and *string* in the table are provided as separate arguments to *expr*.

STDIN

Not used.

INPUT FILES

None.

ENVIRONMENT VARIABLES

The following environment variables shall affect the execution of *expr*:

LANG Provide a default value for the internationalization variables that are unset or null. (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2, Internationalization Variables for the precedence of internationalization variables used to determine the values of locale categories.)

LC_ALL

If set to a non-empty string value, override the values of all the other internationalization variables.

LC_COLLATE

Determine the locale for the behavior of ranges, equivalence classes, and multi-character collating elements within regular expressions and by the string comparison operators.

LC_CTYPE

Determine the locale for the interpretation of sequences of bytes of text data as characters (for example, single-byte as opposed to multi-byte characters in arguments) and the behavior of character classes within regular expressions.

LC_MESSAGES

Determine the locale that should be used to affect the format and contents of diagnostic messages written to standard error.

NLSPATH

Determine the location of message catalogs for the processing of **LC_MESSAGES**.

ASYNCHRONOUS EVENTS

Default.

STDOUT

The *expr* utility shall evaluate the expression and write the result, followed by a <newline>, to standard output.

STDERR

The standard error shall be used only for diagnostic messages.

OUTPUT FILES

None.

EXTENDED DESCRIPTION

The formation of the expression to be evaluated is shown in the following table. The symbols *expr*, *expr1*, and *expr2* represent expressions formed from *integer* and *string* symbols and the expression operator symbols (all separate arguments) by recursive application of the constructs described in the table. The expressions are listed in order of increasing precedence, with equal-precedence operators grouped between horizontal lines. All of the operators shall be left-associative.

Expression	Description
<i>expr1</i> <i>expr2</i>	Returns the evaluation of <i>expr1</i> if it is neither null nor zero; otherwise, returns the evaluation of <i>expr2</i> if it is not null nor zero.
<i>expr1</i> & <i>expr2</i>	Returns the evaluation of <i>expr1</i> if neither expression evaluates to null or zero; otherwise, returns zero.
<i>expr1</i> = <i>expr2</i>	Returns the result of a decimal integer comparison if both arguments are integers; otherwise, returns the result of a string comparison.
<i>expr1</i> > <i>expr2</i>	Equal.
<i>expr1</i> >= <i>expr2</i>	Greater than.
<i>expr1</i> < <i>expr2</i>	Greater than or equal.
<i>expr1</i> <= <i>expr2</i>	Less than.
<i>expr1</i> != <i>expr2</i>	Less than or equal.
<i>expr1</i> + <i>expr2</i>	Not equal.
<i>expr1</i> - <i>expr2</i>	Addition of decimal integer-valued arguments.
<i>expr1</i> * <i>expr2</i>	Subtraction of decimal integer-valued arguments.
<i>expr1</i> / <i>expr2</i>	Multiplication of decimal integer-valued arguments.
<i>expr1</i> % <i>expr2</i>	Integer division of decimal integer-valued arguments, producing an integer result.
<i>expr1</i> : <i>expr2</i>	Remainder of integer division of decimal integer-valued arguments.
(<i>expr</i>)	Matching expression; see below.
<i>integer</i>	Grouping symbols. Any expression can be placed within parentheses. Parentheses can be nested to a depth of {EXPR_DEPTH}.
<i>string</i>	An argument consisting only of an (optional) unary minus followed by digits.
	A string argument; see below.

Matching Expression

The ':' matching operator shall compare the string resulting from the evaluation of *expr1* with the regular expression pattern resulting from the evaluation of *expr2*. Regular expression syntax shall be that defined in the Base Definitions volume of IEEE Std 1003.1-2001, Section 9.3, Basic Regular Expressions, except that all patterns are anchored to the beginning of the string (that is, only sequences starting at the first character of a string are matched by the regular expression) and, therefore, it is unspecified whether '^' is a special character in that context. Usually, the matching operator shall return a string representing the number of characters matched ('0' on failure). Alternatively, if the pattern contains at least one regular expression subexpression "[(...)]" , the string corresponding to "\1" shall be returned.

String Operand

A string argument is an argument that cannot be identified as an *integer* argument or as one of the expression operator symbols shown in the OPERANDS section.

The use of string arguments **length**, **substr**, **index**, or **match** produces unspecified results.

EXIT STATUS

The following exit values shall be returned:

- 0 The *expression* evaluates to neither null nor zero.
- 1 The *expression* evaluates to null or zero.
- 2 Invalid *expression*.
- >2 An error occurred.

CONSEQUENCES OF ERRORS

Default.

The following sections are informative.

APPLICATION USAGE

After argument processing by the shell, *expr* is not required to be able to tell the difference between an operator and an operand except by the value. If "**\$a**" is '=', the command:

```
expr $a = '='
```

looks like:

```
expr ==
```

as the arguments are passed to *expr* (and they all may be taken as the '=' operator). The following works reliably:

```
expr X$a = X=
```

Also note that this volume of IEEE Std 1003.1-2001 permits implementations to extend utilities. The *expr* utility permits the integer arguments to be preceded with a unary minus. This means that an integer argument could look like an option. Therefore, the conforming application must employ the "--" construct of Guideline 10 of the Base Definitions volume of IEEE Std 1003.1-2001, Section 12.2, Utility Syntax Guidelines to protect its operands if there is any chance the first operand might be a negative integer (or any string with a leading minus).

EXAMPLES

The *expr* utility has a rather difficult syntax:

Many of the operators are also shell control operators or reserved words, so they have to be escaped on the command line.

Each part of the expression is composed of separate arguments, so liberal usage of <blank>s is required. For example:

Invalid	Valid
<i>expr</i> 1+2	<i>expr</i> 1 + 2
<i>expr</i> "1 + 2"	<i>expr</i> 1 + 2
<i>expr</i> 1 + (2 * 3)	<i>expr</i> 1 + \(2 * 3 \)

In many cases, the arithmetic and string features provided as part of the shell command language are easier to use than their equivalents in *expr*. Newly written scripts should avoid *expr* in favor of the new features within the shell; see *Parameters and Variables* and *Arithmetic Expansion*.

The following command:

```
a=$(expr $a + 1)
```

adds 1 to the variable *a*.

The following command, for "**\$a**" equal to either */usr/abc/file* or just *file*:

```
expr $a : '.*\(.*)' \| $a
```

returns the last segment of a pathname (that is, **file**). Applications should avoid the character '/' used alone as an argument; *expr* may interpret it as the division operator.

The following command:

```
expr "$a" : '.*\(.*)'
```

is a better representation of the previous example. The addition of the "\$" characters eliminates any ambiguity about the division operator and simplifies the whole expression. Also note that pathnames may contain characters contained in the *IFS* variable and should be quoted to avoid having "**\$a**" expand into multiple arguments.

The following command:

expr "\$VAR" : '.*'

returns the number of characters in *VAR*.

RATIONALE

In an early proposal, EREs were used in the matching expression syntax. This was changed to BREs to avoid breaking historical applications.

The use of a leading circumflex in the BRE is unspecified because many historical implementations have treated it as a special character, despite their system documentation. For example:

expr foo : ^foo expr ^foo : ^foo

return 3 and 0, respectively, on those systems; their documentation would imply the reverse. Thus, the anchoring condition is left unspecified to avoid breaking historical scripts relying on this undocumented feature.

FUTURE DIRECTIONS

None.

SEE ALSO

Parameters and Variables , Arithmetic Expansion

COPYRIGHT

Portions of this text are reprinted and reproduced in electronic form from IEEE Std 1003.1, 2003 Edition, Standard for Information Technology -- Portable Operating System Interface (POSIX), The Open Group Base Specifications Issue 6, Copyright (C) 2001-2003 by the Institute of Electrical and Electronics Engineers, Inc and The Open Group. In the event of any discrepancy between this version and the original IEEE and The Open Group Standard, the original IEEE and The Open Group Standard is the referee document. The original Standard can be obtained online at <http://www.open-group.org/unix/online.html> .

NAME

false - return false value

SYNOPSIS

false

DESCRIPTION

The *false* utility shall return with a non-zero exit code.

OPTIONS

None.

OPERANDS

None.

STDIN

Not used.

INPUT FILES

None.

ENVIRONMENT VARIABLES

None.

ASYNCHRONOUS EVENTS

Default.

STDOUT

Not used.

STDERR

Not used.

OUTPUT FILES

None.

EXTENDED DESCRIPTION

None.

EXIT STATUS

The *false* utility shall always exit with a value other than zero.

CONSEQUENCES OF ERRORS

Default.

The following sections are informative.

APPLICATION USAGE

None.

EXAMPLES

None.

RATIONALE

None.

FUTURE DIRECTIONS

None.

SEE ALSO

true

COPYRIGHT

Portions of this text are reprinted and reproduced in electronic form from IEEE Std 1003.1, 2003 Edition, Standard for Information Technology -- Portable Operating System Interface (POSIX), The Open Group Base Specifications Issue 6, Copyright (C) 2001-2003 by the Institute of Electrical and Electronics Engineers, Inc and The Open Group. In the event of any discrepancy between this version and the original IEEE and The Open Group Standard, the original IEEE and The Open Group Standard is the referee document. The original Standard can be obtained online at <http://www.open-group.org/unix/online.html>.

NAME

fold - filter for folding lines

SYNOPSIS

fold [-bs][**-w** *width*][*file...*]

DESCRIPTION

The *fold* utility is a filter that shall fold lines from its input files, breaking the lines to have a maximum of *width* column positions (or bytes, if the **-b** option is specified). Lines shall be broken by the insertion of a <newline> such that each output line (referred to later in this section as a *segment*) is the maximum width possible that does not exceed the specified number of column positions (or bytes). A line shall not be broken in the middle of a character. The behavior is undefined if *width* is less than the number of columns any single character in the input would occupy.

If the <carriage-return>s, <backspace>s, or <tab>s are encountered in the input, and the **-b** option is not specified, they shall be treated specially:

<backspace>

The current count of line width shall be decremented by one, although the count never shall become negative. The *fold* utility shall not insert a <newline> immediately before or after any <backspace>.

<carriage-return>

The current count of line width shall be set to zero. The *fold* utility shall not insert a <newline> immediately before or after any <carriage-return>.

<tab> Each <tab> encountered shall advance the column position pointer to the next tab stop. Tab stops shall be at each column position *n* such that *n* modulo 8 equals 1.

OPTIONS

The *fold* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section 12.2, Utility Syntax Guidelines.

The following options shall be supported:

-b Count *width* in bytes rather than column positions.

-s If a segment of a line contains a <blank> within the first *width* column positions (or bytes), break the line after the last such <blank> meeting the width constraints. If there is no <blank> meeting the requirements, the **-s** option shall have no effect for that output segment of the input line.

-w *width*

Specify the maximum line length, in column positions (or bytes if **-b** is specified). The results are unspecified if *width* is not a positive decimal number. The default value shall be 80.

OPERANDS

The following operand shall be supported:

file A pathname of a text file to be folded. If no *file* operands are specified, the standard input shall be used.

STDIN

The standard input shall be used only if no *file* operands are specified. See the INPUT FILES section.

INPUT FILES

If the **-b** option is specified, the input files shall be text files except that the lines are not limited to {LINE_MAX} bytes in length. If the **-b** option is not specified, the input files shall be text files.

ENVIRONMENT VARIABLES

The following environment variables shall affect the execution of *fold*:

LANG Provide a default value for the internationalization variables that are unset or null. (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2, Internationalization Variables for the precedence of internationalization variables used to determine the values of locale categories.)

LC_ALL

If set to a non-empty string value, override the values of all the other internationalization variables.

LC_CTYPE

Determine the locale for the interpretation of sequences of bytes of text data as characters (for example, single-byte as opposed to multi-byte characters in arguments and input files), and for the determination of the width in column positions each character would occupy on a constant-width font output device.

LC_MESSAGES

Determine the locale that should be used to affect the format and contents of diagnostic messages written to standard error.

NLSPATH

Determine the location of message catalogs for the processing of **LC_MESSAGES**.

ASYNCHRONOUS EVENTS

Default.

STDOUT

The standard output shall be a file containing a sequence of characters whose order shall be preserved from the input files, possibly with inserted <newline>s.

STDERR

The standard error shall be used only for diagnostic messages.

OUTPUT FILES

None.

EXTENDED DESCRIPTION

None.

EXIT STATUS

The following exit values shall be returned:

- 0 All input files were processed successfully.
- >0 An error occurred.

CONSEQUENCES OF ERRORS

Default.

The following sections are informative.

APPLICATION USAGE

The *cut* and *fold* utilities can be used to create text files out of files with arbitrary line lengths. The *cut* utility should be used when the number of lines (or records) needs to remain constant. The *fold* utility should be used when the contents of long lines need to be kept contiguous.

The *fold* utility is frequently used to send text files to printers that truncate, rather than fold, lines wider than the printer is able to print (usually 80 or 132 column positions).

EXAMPLES

An example invocation that submits a file of possibly long lines to the printer (under the assumption that the user knows the line width of the printer to be assigned by *lp*):

```
fold -w 132 bigfile | lp
```

RATIONALE

Although terminal input in canonical processing mode requires the erase character (frequently set to <backspace>) to erase the previous character (not byte or column position), terminal output is not buffered and is extremely difficult, if not impossible, to parse correctly; the interpretation depends entirely on the physical device that actually displays/prints/stores the output. In all known internationalized implementations, the utilities producing output for mixed column-width output assume that a <backspace> backs up one column position and outputs enough <backspace>s to return to the start of the character when <backspace> is used to provide local line motions to support underlining and boldening operations. Since *fold* without the **-b** option is dealing with these same constraints, <backspace> is always treated as backing up one column position rather than backing up one character.

Historical versions of the *fold* utility assumed 1 byte was one character and occupied one column position when written out. This is no longer always true. Since the most common usage of *fold* is believed to be folding long lines for output to limited-length output devices, this capability was preserved as the default case. The **-b** option was added so that applications could *fold* files with arbitrary length lines into text files that could then be processed by the standard utilities. Note that although the width for the **-b** option is in bytes, a line is never split in the middle of a character. (It is unspecified what happens if a width is specified that is too small to hold a single character found in the input followed by a <new-line>.)

The tab stops are hardcoded to be every eighth column to meet historical practice. No new method of specifying other tab stops was invented.

FUTURE DIRECTIONS

None.

SEE ALSO

cut

COPYRIGHT

Portions of this text are reprinted and reproduced in electronic form from IEEE Std 1003.1, 2003 Edition, Standard for Information Technology -- Portable Operating System Interface (POSIX), The Open Group Base Specifications Issue 6, Copyright (C) 2001-2003 by the Institute of Electrical and Electronics Engineers, Inc and The Open Group. In the event of any discrepancy between this version and the original IEEE and The Open Group Standard, the original IEEE and The Open Group Standard is the referee document. The original Standard can be obtained online at <http://www.open-group.org/unix/online.html>.

NAME

head - copy the first part of files

SYNOPSIS

head [-n *number*][*file*...]

DESCRIPTION

The *head* utility shall copy its input files to the standard output, ending the output for each file at a designated point.

Copying shall end at the point in each input file indicated by the **-n** *number* option. The option-argument *number* shall be counted in units of lines.

OPTIONS

The *head* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section 12.2, Utility Syntax Guidelines.

The following option shall be supported:

-n *number*

The first *number* lines of each input file shall be copied to standard output. The application shall ensure that the *number* option-argument is a positive decimal integer.

When a file contains less than *number* lines, it shall be copied to standard output in its entirety. This shall not be an error.

If no options are specified, *head* shall act as if **-n 10** had been specified.

OPERANDS

The following operand shall be supported:

file A pathname of an input file. If no *file* operands are specified, the standard input shall be used.

STDIN

The standard input shall be used only if no *file* operands are specified. See the INPUT FILES section.

INPUT FILES

Input files shall be text files, but the line length is not restricted to {LINE_MAX} bytes.

ENVIRONMENT VARIABLES

The following environment variables shall affect the execution of *head*:

LANG Provide a default value for the internationalization variables that are unset or null. (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2, Internationalization Variables for the precedence of internationalization variables used to determine the values of locale categories.)

LC_ALL

If set to a non-empty string value, override the values of all the other internationalization variables.

LC_CTYPE

Determine the locale for the interpretation of sequences of bytes of text data as characters (for example, single-byte as opposed to multi-byte characters in arguments and input files).

LC_MESSAGES

Determine the locale that should be used to affect the format and contents of diagnostic messages written to standard error.

NLSPATH

Determine the location of message catalogs for the processing of *LC_MESSAGES*.

ASYNCHRONOUS EVENTS

Default.

STDOUT

The standard output shall contain designated portions of the input files.

If multiple *file* operands are specified, *head* shall precede the output for each with the header:

```
"\n==> %s <==\n", <pathname>
```

except that the first header written shall not include the initial <newline>.

STDERR

The standard error shall be used only for diagnostic messages.

OUTPUT FILES

None.

EXTENDED DESCRIPTION

None.

EXIT STATUS

The following exit values shall be returned:

- | | |
|----|------------------------|
| 0 | Successful completion. |
| >0 | An error occurred. |

CONSEQUENCES OF ERRORS

Default.

The following sections are informative.

APPLICATION USAGE

The obsolescent *- number* form is withdrawn in this version. Applications should use the **-n number** option.

EXAMPLES

To write the first ten lines of all files (except those with a leading period) in the directory:

```
head *
```

RATIONALE

Although it is possible to simulate *head* with *sed* 10q for a single file, the standard developers decided that the popularity of *head* on historical BSD systems warranted its inclusion alongside *tail*.

This standard version of *head* follows the Utility Syntax Guidelines. The **-n** option was added to this new interface so that *head* and *tail* would be more logically related.

There is no **-c** option (as there is in *tail*) because it is not historical practice and because other utilities in this volume of IEEE Std 1003.1-2001 provide similar functionality.

FUTURE DIRECTIONS

None.

SEE ALSO

sed, *tail*

COPYRIGHT

Portions of this text are reprinted and reproduced in electronic form from IEEE Std 1003.1, 2003 Edition, Standard for Information Technology -- Portable Operating System Interface (POSIX), The Open Group Base Specifications Issue 6, Copyright (C) 2001-2003 by the Institute of Electrical and Electronics Engineers, Inc and The Open Group. In the event of any discrepancy between this version and the original IEEE and The Open Group Standard, the original IEEE and The Open Group Standard is the referee document. The original Standard can be obtained online at <http://www.open-group.org/unix/online.html>.

NAME

id - return user identity

SYNOPSIS

id [*user*]

id -G[-n] [*user*]

id -g[-nr] [*user*]

id -u[-nr] [*user*]

DESCRIPTION

If no *user* operand is provided, the *id* utility shall write the user and group IDs and the corresponding user and group names of the invoking process to standard output. If the effective and real IDs do not match, both shall be written. If multiple groups are supported by the underlying system (see the description of {NGROUPS_MAX} in the System Interfaces volume of IEEE Std 1003.1-2001), the supplementary group affiliations of the invoking process shall also be written.

If a *user* operand is provided and the process has the appropriate privileges, the user and group IDs of the selected user shall be written. In this case, effective IDs shall be assumed to be identical to real IDs. If the selected user has more than one allowable group membership listed in the group database, these shall be written in the same manner as the supplementary groups described in the preceding paragraph.

OPTIONS

The *id* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section 12.2, Utility Syntax Guidelines.

The following options shall be supported:

- G** Output all different group IDs (effective, real, and supplementary) only, using the format "%**u**\n" . If there is more than one distinct group affiliation, output each such affiliation, using the format " %**u**" , before the <newline> is output.
- g** Output only the effective group ID, using the format "%**u**\n" .
- n** Output the name in the format "%**s**" instead of the numeric ID using the format "%**u**" .
- r** Output the real ID instead of the effective ID.
- u** Output only the effective user ID, using the format "%**u**\n" .

OPERANDS

The following operand shall be supported:

user The login name for which information is to be written.

STDIN

Not used.

INPUT FILES

None.

ENVIRONMENT VARIABLES

The following environment variables shall affect the execution of *id*:

LANG Provide a default value for the internationalization variables that are unset or null. (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2, Internationalization Variables for the precedence of internationalization variables used to determine the values of locale categories.)

LC_ALL

If set to a non-empty string value, override the values of all the other internationalization variables.

LC_CTYPE

Determine the locale for the interpretation of sequences of bytes of text data as characters (for example, single-byte as opposed to multi-byte characters in arguments).

LC_MESSAGES

Determine the locale that should be used to affect the format and contents of diagnostic messages written to standard error and informative messages written to standard output.

NLSPATH

Determine the location of message catalogs for the processing of *LC_MESSAGES*.

ASYNCHRONOUS EVENTS

Default.

STDOUT

The following formats shall be used when the *LC_MESSAGES* locale category specifies the POSIX locale. In other locales, the strings *uid*, *gid*, *euid*, *egid*, and *groups* may be replaced with more appropriate strings corresponding to the locale.

```
"uid=%u(%s) gid=%u(%s)\n", <real user ID>, <user-name>,
    <real group ID>, <group-name>
```

If the effective and real user IDs do not match, the following shall be inserted immediately before the '\n' character in the previous format:

```
" euid=%u(%s)"
```

with the following arguments added at the end of the argument list:

```
<effective user ID>, <effective user-name>
```

If the effective and real group IDs do not match, the following shall be inserted directly before the '\n' character in the format string (and after any addition resulting from the effective and real user IDs not matching):

```
" egid=%u(%s)"
```

with the following arguments added at the end of the argument list:

```
<effective group-ID>, <effective group name>
```

If the process has supplementary group affiliations or the selected user is allowed to belong to multiple groups, the first shall be added directly before the <newline> in the format string:

```
" groups=%u(%s)"
```

with the following arguments added at the end of the argument list:

```
<supplementary group ID>, <supplementary group name>
```

and the necessary number of the following added after that for any remaining supplementary group IDs:

```
",%u(%s)"
```

and the necessary number of the following arguments added at the end of the argument list:

<*supplementary group ID*>, <*supplementary group name*>

If any of the user ID, group ID, effective user ID, effective group ID, or supplementary/multiple group IDs cannot be mapped by the system into printable user or group names, the corresponding "(%s)" and *name* argument shall be omitted from the corresponding format string.

When any of the options are specified, the output format shall be as described in the OPTIONS section.

STDERR

The standard error shall be used only for diagnostic messages.

OUTPUT FILES

None.

EXTENDED DESCRIPTION

None.

EXIT STATUS

The following exit values shall be returned:

- 0 Successful completion.
- >0 An error occurred.

CONSEQUENCES OF ERRORS

Default.

The following sections are informative.

APPLICATION USAGE

Output produced by the **-G** option and by the default case could potentially produce very long lines on systems that support large numbers of supplementary groups. (On systems with user and group IDs that are 32-bit integers and with group names with a maximum of 8 bytes per name, 93 supplementary groups plus distinct effective and real group and user IDs could theoretically overflow the 2048-byte {LINE_MAX} text file line limit on the default output case. It would take about 186 supplementary groups to overflow the 2048-byte barrier using *id -G*). This is not expected to be a problem in practice, but in cases where it is a concern, applications should consider using *fold -s* before postprocessing the output of *id*.

EXAMPLES

None.

RATIONALE

The functionality provided by the 4 BSD *groups* utility can be simulated using:

id -Gn [user]

The 4 BSD command *groups* was considered, but it was not included because it did not provide the functionality of the *id* utility of the SVID. Also, it was thought that it would be easier to modify *id* to provide the additional functionality necessary to systems with multiple groups than to invent another command.

The options **-u**, **-g**, **-n**, and **-r** were added to ease the use of *id* with shell commands substitution. Without these options it is necessary to use some preprocessor such as *sed* to select the desired piece of information. Since output such as that produced by:

id -u -n

is frequently wanted, it seemed desirable to add the options.

FUTURE DIRECTIONS

None.

SEE ALSO

fold, *logname*, *who*, the System Interfaces volume of IEEE Std 1003.1-2001, *getgid()*, *getgroups()*, *getuid()*

COPYRIGHT

Portions of this text are reprinted and reproduced in electronic form from IEEE Std 1003.1, 2003 Edition, Standard for Information Technology -- Portable Operating System Interface (POSIX), The Open Group Base Specifications Issue 6, Copyright (C) 2001-2003 by the Institute of Electrical and Electronics Engineers, Inc and The Open Group. In the event of any discrepancy between this version and the original IEEE and The Open Group Standard, the original IEEE and The Open Group Standard is the referee document. The original Standard can be obtained online at <http://www.open-group.org/unix/online.html> .

NAME

join - relational database operator

SYNOPSIS

```
join [-a file_number | -v file_number][-e string][-o list][-t char]
      [-1 field][-2 field] file1 file2
```

DESCRIPTION

The *join* utility shall perform an equality join on the files *file1* and *file2*. The joined files shall be written to the standard output.

The join field is a field in each file on which the files are compared. The *join* utility shall write one line in the output for each pair of lines in *file1* and *file2* that have identical join fields. The output line by default shall consist of the join field, then the remaining fields from *file1*, then the remaining fields from *file2*. This format can be changed by using the **-o** option (see below). The **-a** option can be used to add unmatched lines to the output. The **-v** option can be used to output only unmatched lines.

The files *file1* and *file2* shall be ordered in the collating sequence of *sort -b* on the fields on which they shall be joined, by default the first in each line. All selected output shall be written in the same collating sequence.

The default input field separators shall be <blank>s. In this case, multiple separators shall count as one field separator, and leading separators shall be ignored. The default output field separator shall be a <space>.

The field separator and collating sequence can be changed by using the **-t** option (see below).

If the same key appears more than once in either file, all combinations of the set of remaining fields in *file1* and the set of remaining fields in *file2* are output in the order of the lines encountered.

If the input files are not in the appropriate collating sequence, the results are unspecified.

OPTIONS

The *join* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section 12.2, Utility Syntax Guidelines.

The following options shall be supported:

-a *file_number*

Produce a line for each unpairable line in file *file_number*, where *file_number* is 1 or 2, in addition to the default output. If both **-a1** and **-a2** are specified, all unpairable lines shall be output.

-e *string*

Replace empty output fields in the list selected by **-o** with the string *string*.

-o *list* Construct the output line to comprise the fields specified in *list*, each element of which shall have one of the following two forms:

file_number.field, where *file_number* is a file number and *field* is a decimal integer field number

0 (zero), representing the join field

The elements of *list* shall be either comma-separated or <blank>-separated, as specified in Guideline 8 of the Base Definitions volume of IEEE Std 1003.1-2001, Section 12.2, Utility Syntax Guidelines. The fields specified by *list* shall be written for all selected output lines. Fields selected by *list* that do not appear in the input shall be treated as empty output fields. (See the **-e** option.) Only specifically requested fields shall be written. The application shall ensure that *list* is a single command line argument.

-t *char*

Use character *char* as a separator, for both input and output. Every appearance of *char* in a line shall be significant. When this option is specified, the collating sequence shall be the same as *sort* without the **-b** option.

-v *file_number*

Instead of the default output, produce a line only for each unpairable line in *file_number*, where *file_number* is 1 or 2. If both **-v1** and **-v2** are specified, all unpairable lines shall be output.

-1 *field*

Join on the *fieldth* field of file 1. Fields are decimal integers starting with 1.

-2 *field*

Join on the *fieldth* field of file 2. Fields are decimal integers starting with 1.

OPERANDS

The following operands shall be supported:

file1, *file2*

A pathname of a file to be joined. If either of the *file1* or *file2* operands is '-', the standard input shall be used in its place.

STDIN

The standard input shall be used only if the *file1* or *file2* operand is '-'. See the INPUT FILES section.

INPUT FILES

The input files shall be text files.

ENVIRONMENT VARIABLES

The following environment variables shall affect the execution of *join*:

LANG Provide a default value for the internationalization variables that are unset or null. (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2, Internationalization Variables for the precedence of internationalization variables used to determine the values of locale categories.)

LC_ALL

If set to a non-empty string value, override the values of all the other internationalization variables.

LC_COLLATE

Determine the locale of the collating sequence *join* expects to have been used when the input files were sorted.

LC_CTYPE

Determine the locale for the interpretation of sequences of bytes of text data as characters (for example, single-byte as opposed to multi-byte characters in arguments and input files).

LC_MESSAGES

Determine the locale that should be used to affect the format and contents of diagnostic messages written to standard error.

NLSPATH

Determine the location of message catalogs for the processing of **LC_MESSAGES**.

ASYNCHRONOUS EVENTS

Default.

STDOUT

The *join* utility output shall be a concatenation of selected character fields. When the **-o** option is not specified, the output shall be:

```
"%s%s%s\n", <join field>, <other file1 fields>,
<other file2 fields>
```

If the join field is not the first field in a file, the *<other file fields>* for that file shall be:

join(P)

join(P)

<fields preceding join field>, <fields following join field>

When the **-o** option is specified, the output format shall be:

"%s\n", <concatenation of fields>

where the concatenation of fields is described by the **-o** option, above.

For either format, each field (except the last) shall be written with its trailing separator character. If the separator is the default (**<blank>s**), a single **<space>** shall be written after each field (except the last).

STDERR

The standard error shall be used only for diagnostic messages.

OUTPUT FILES

None.

EXTENDED DESCRIPTION

None.

EXIT STATUS

The following exit values shall be returned:

- 0 All input files were output successfully.
- >0 An error occurred.

CONSEQUENCES OF ERRORS

Default.

The following sections are informative.

APPLICATION USAGE

Pathnames consisting of numeric digits or of the form *string.string* should not be specified directly following the **-o** list.

EXAMPLES

The **-o 0** field essentially selects the union of the join fields. For example, given file **phone**:

!Name	Phone Number
Don	+1 123-456-7890
Hal	+1 234-567-8901
Yasushi	+2 345-678-9012

and file **fax**:

!Name	Fax Number
Don	+1 123-456-7899
Keith	+1 456-789-0122
Yasushi	+2 345-678-9011

(where the large expanses of white space are meant to each represent a single **<tab>**), the command:

join -t "<tab>" -a 1 -a 2 -e '(unknown)' -o 0,1,2,2 phone fax

would produce:

!Name	Phone Number	Fax Number
Don	+1 123-456-7890	+1 123-456-7899
Hal	+1 234-567-8901	(unknown)
Keith	(unknown)	+1 456-789-0122
Yasushi	+2 345-678-9012	+2 345-678-9011

join(P)

join(P)

Multiple instances of the same key will produce combinatorial results. The following:

```
fa:
  a x
  a y
  a z
fb:
  a p
```

will produce:

```
a x p
a y p
a z p
```

And the following:

```
fa:
  a b c
  a d e
fb:
  a w x
  a y z
  a o p
```

will produce:

```
a b c w x
a b c y z
a b c o p
a d e w x
a d e y z
a d e o p
```

RATIONALE

The **-e** option is only effective when used with **-o** because, unless specific fields are identified using **-o**, *join* is not aware of what fields might be empty. The exception to this is the join field, but identifying an empty join field with the **-e** string is not historical practice and some scripts might break if this were changed.

The 0 field in the **-o** list was adopted from the Tenth Edition version of *join* to satisfy international objections that the *join* in the base documents does not support the "full join" or "outer join" described in relational database literature. Although it has been possible to include a join field in the output (by default, or by field number using **-o**), the join field could not be included for an unpaired line selected by **-a**. The **-o 0** field essentially selects the union of the join fields.

This sort of outer join was not possible with the *join* commands in the base documents. The **-o 0** field was chosen because it is an upwards-compatible change for applications. An alternative was considered: have the join field represent the union of the fields in the files (where they are identical for matched lines, and one or both are null for unmatched lines). This was not adopted because it would break some historical applications.

The ability to specify *file2* as **-** is not historical practice; it was added for completeness.

The **-v** option is not historical practice, but was considered necessary because it permitted the writing of *only* those lines that do not match on the join field, as opposed to the **-a** option, which prints both lines that do and do not match. This additional facility is parallel with the **-v** option of *grep*.

Some historical implementations have been encountered where a blank line in one of the input files was considered to be the end of the file; the description in this volume of IEEE Std 1003.1-2001 does not

join(P)

join(P)

cite this as an allowable case.

FUTURE DIRECTIONS

None.

SEE ALSO

awk , comm , sort , uniq

COPYRIGHT

Portions of this text are reprinted and reproduced in electronic form from IEEE Std 1003.1, 2003 Edition, Standard for Information Technology -- Portable Operating System Interface (POSIX), The Open Group Base Specifications Issue 6, Copyright (C) 2001-2003 by the Institute of Electrical and Electronics Engineers, Inc and The Open Group. In the event of any discrepancy between this version and the original IEEE and The Open Group Standard, the original IEEE and The Open Group Standard is the referee document. The original Standard can be obtained online at <http://www.open-group.org/unix/online.html> .

NAME

kill - terminate or signal processes

SYNOPSIS

kill **-s** *signal_name* *pid* ...

kill **-l** [*exit_status*]

kill [**-signal_name**] *pid* ...

kill [**-signal_number**] *pid* ...

DESCRIPTION

The *kill* utility shall send a signal to the process or processes specified by each *pid* operand.

For each *pid* operand, the *kill* utility shall perform actions equivalent to the *kill()* function defined in the System Interfaces volume of IEEE Std 1003.1-2001 called with the following arguments:

The value of the *pid* operand shall be used as the *pid* argument.

The *sig* argument is the value specified by the **-s** option, **-signal_number** option, or the **-signal_name** option, or by SIGTERM, if none of these options is specified.

OPTIONS

The *kill* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section 12.2, Utility Syntax Guidelines, except that in the last two SYNOPSIS forms, the **-signal_number** and **-signal_name** options are usually more than a single character.

The following options shall be supported:

-l (The letter ell.) Write all values of *signal_name* supported by the implementation, if no operand is given. If an *exit_status* operand is given and it is a value of the **??** shell special parameter (see *Special Parameters* and *wait()*) corresponding to a process that was terminated by a signal, the *signal_name* corresponding to the signal that terminated the process shall be written. If an *exit_status* operand is given and it is the unsigned decimal integer value of a signal number, the *signal_name* (the symbolic constant name without the **SIG** prefix defined in the Base Definitions volume of IEEE Std 1003.1-2001) corresponding to that signal shall be written. Otherwise, the results are unspecified.

-s *signal_name*

Specify the signal to send, using one of the symbolic names defined in the *<signal.h>* header. Values of *signal_name* shall be recognized in a case-independent fashion, without the **SIG** prefix. In addition, the symbolic name 0 shall be recognized, representing the signal value zero. The corresponding signal shall be sent instead of SIGTERM.

-signal_name

Equivalent to **-s** *signal_name*.

-signal_number

Specify a non-negative decimal integer, *signal_number*, representing the signal to be used instead of SIGTERM, as the *sig* argument in the effective call to *kill()*. The correspondence between integer values and the *sig* value used is shown in the following table.

The effects of specifying any *signal_number* other than those listed in the table are undefined.

<i>signal_number</i>	<i>sig</i> Value
0	0

<i>1</i>	<i>SIGHUP</i>
<i>2</i>	<i>SIGINT</i>
<i>3</i>	<i>SIGQUIT</i>
<i>6</i>	<i>SIGABRT</i>
<i>9</i>	<i>SIGKILL</i>
<i>14</i>	<i>SIGALRM</i>
<i>15</i>	<i>SIGTERM</i>

If the first argument is a negative integer, it shall be interpreted as a *- signal_number* option, not as a negative *pid* operand specifying a process group.

OPERANDS

The following operands shall be supported:

pid One of the following:

A decimal integer specifying a process or process group to be signaled. The process or processes selected by positive, negative, and zero values of the *pid* operand shall be as described for the *kill()* function. If process number 0 is specified, all processes in the current process group shall be signaled. For the effects of negative *pid* numbers, see the *kill()* function defined in the System Interfaces volume of IEEE Std 1003.1-2001. If the first *pid* operand is negative, it should be preceded by "--" to keep it from being interpreted as an option.

A job control job ID (see the Base Definitions volume of IEEE Std 1003.1-2001, Section 3.203, Job Control Job ID) that identifies a background process group to be signaled. The job control job ID notation is applicable only for invocations of *kill* in the current shell execution environment; see *Shell Execution Environment*.

exit_status

A decimal integer specifying a signal number or the exit status of a process terminated by a signal.

STDIN

Not used.

INPUT FILES

None.

ENVIRONMENT VARIABLES

The following environment variables shall affect the execution of *kill*:

LANG Provide a default value for the internationalization variables that are unset or null. (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2, Internationalization Variables for the precedence of internationalization variables used to determine the values of locale categories.)

LC_ALL

If set to a non-empty string value, override the values of all the other internationalization variables.

LC_CTYPE

Determine the locale for the interpretation of sequences of bytes of text data as characters (for example, single-byte as opposed to multi-byte characters in arguments).

LC_MESSAGES

Determine the locale that should be used to affect the format and contents of diagnostic messages written to standard error.

NLSPATH

Determine the location of message catalogs for the processing of *LC_MESSAGES*.

ASYNCHRONOUS EVENTS

Default.

STDOUT

When the **-l** option is not specified, the standard output shall not be used.

When the **-l** option is specified, the symbolic name of each signal shall be written in the following format:

```
"%s%c", <signal_name>, <separator>
```

where the *<signal_name>* is in uppercase, without the **SIG** prefix, and the *<separator>* shall be either a *<newline>* or a *<space>*. For the last signal written, *<separator>* shall be a *<newline>*.

When both the **-l** option and *exit_status* operand are specified, the symbolic name of the corresponding signal shall be written in the following format:

```
"%s\n", <signal_name>
```

STDERR

The standard error shall be used only for diagnostic messages.

OUTPUT FILES

None.

EXTENDED DESCRIPTION

None.

EXIT STATUS

The following exit values shall be returned:

- 0 At least one matching process was found for each *pid* operand, and the specified signal was successfully processed for at least one matching process.
- >0 An error occurred.

CONSEQUENCES OF ERRORS

Default.

The following sections are informative.

APPLICATION USAGE

Process numbers can be found by using *ps*.

The job control job ID notation is not required to work as expected when *kill* is operating in its own utility execution environment. In either of the following examples:

```
nohup kill %1 &
system("kill %1");
```

the *kill* operates in a different environment and does not share the shell's understanding of job numbers.

EXAMPLES

Any of the commands:

```
kill -9 100 -165
kill -s kill 100 -165
kill -s KILL 100 -165
```

sends the SIGKILL signal to the process whose process ID is 100 and to all processes whose process group ID is 165, assuming the sending process has permission to send that signal to the specified processes, and that they exist.

The System Interfaces volume of IEEE Std 1003.1-2001 and this volume of IEEE Std 1003.1-2001 do not require specific signal numbers for any *signal_names*. Even the *- signal_number* option provides symbolic (although numeric) names for signals. If a process is terminated by a signal, its exit status indicates the signal that killed it, but the exact values are not specified. The *kill -l* option, however, can

be used to map decimal signal numbers and exit status values into the name of a signal. The following example reports the status of a terminated job:

```

job
stat=$?
if [ $stat -eq 0 ]
then
    echo job completed successfully.
elif [ $stat -gt 128 ]
then
    echo job terminated by signal SIG$(kill -l $stat).
else
    echo job terminated with error code $stat.
fi

```

To send the default signal to a process group (say 123), an application should use a command similar to one of the following:

```

kill -TERM -123
kill -- -123

```

RATIONALE

The **-l** option originated from the C shell, and is also implemented in the KornShell. The C shell output can consist of multiple output lines because the signal names do not always fit on a single line on some terminal screens. The KornShell output also included the implementation-defined signal numbers and was considered by the standard developers to be too difficult for scripts to parse conveniently. The specified output format is intended not only to accommodate the historical C shell output, but also to permit an entirely vertical or entirely horizontal listing on systems for which this is appropriate.

An early proposal invented the name *SIGNULL* as a *signal_name* for signal 0 (used by the System Interfaces volume of IEEE Std 1003.1-2001 to test for the existence of a process without sending it a signal). Since the *signal_name* 0 can be used in this case unambiguously, *SIGNULL* has been removed.

An early proposal also required symbolic *signal_names* to be recognized with or without the **SIG** prefix. Historical versions of *kill* have not written the **SIG** prefix for the **-l** option and have not recognized the **SIG** prefix on *signal_names*. Since neither applications portability nor ease-of-use would be improved by requiring this extension, it is no longer required.

To avoid an ambiguity of an initial negative number argument specifying either a signal number or a process group, IEEE Std 1003.1-2001 mandates that it is always considered the former by implementations that support the XSI option. It also requires that conforming applications always use the "--" options terminator argument when specifying a process group, unless an option is also specified.

The **-s** option was added in response to international interest in providing some form of *kill* that meets the Utility Syntax Guidelines.

The job control job ID notation is not required to work as expected when *kill* is operating in its own utility execution environment. In either of the following examples:

```

nohup kill %1 &
system("kill %1");

```

the *kill* operates in a different environment and does not understand how the shell has managed its job numbers.

FUTURE DIRECTIONS

None.

SEE ALSO

Shell Command Language , *ps* , *wait()* , the System Interfaces volume of IEEE Std 1003.1-2001, *kill()*, the Base Definitions volume of IEEE Std 1003.1-2001, <*signal.h*>

COPYRIGHT

Portions of this text are reprinted and reproduced in electronic form from IEEE Std 1003.1, 2003 Edition, Standard for Information Technology -- Portable Operating System Interface (POSIX), The Open Group Base Specifications Issue 6, Copyright (C) 2001-2003 by the Institute of Electrical and Electronics Engineers, Inc and The Open Group. In the event of any discrepancy between this version and the original IEEE and The Open Group Standard, the original IEEE and The Open Group Standard is the referee document. The original Standard can be obtained online at <http://www.open-group.org/unix/online.html> .

link(P)

link(P)

NAME

link - call link function

SYNOPSIS

link *file1 file2*

DESCRIPTION

The *link* utility shall perform the function call:

```
link(file1, file2);
```

A user may need appropriate privilege to invoke the *link* utility.

OPTIONS

None.

OPERANDS

The following operands shall be supported:

file1 The pathname of an existing file.

file2 The pathname of the new directory entry to be created.

STDIN

Not used.

INPUT FILES

Not used.

ENVIRONMENT VARIABLES

The following environment variables shall affect the execution of *link*:

LANG Provide a default value for the internationalization variables that are unset or null. (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2, Internationalization Variables for the precedence of internationalization variables used to determine the values of locale categories.)

LC_ALL If set to a non-empty string value, override the values of all the other internationalization variables.

LC_CTYPE Determine the locale for the interpretation of sequences of bytes of text data as characters (for example, single-byte as opposed to multi-byte characters in arguments).

LC_MESSAGES Determine the locale that should be used to affect the format and contents of diagnostic messages written to standard error.

NLSPATH Determine the location of message catalogs for the processing of *LC_MESSAGES*.

ASYNCHRONOUS EVENTS

Default.

STDOUT

None.

STDERR

The standard error shall be used only for diagnostic messages.

OUTPUT FILES

None.

EXTENDED DESCRIPTION

None.

EXIT STATUS

The following exit values shall be returned:

- | | |
|----|------------------------|
| 0 | Successful completion. |
| >0 | An error occurred. |

CONSEQUENCES OF ERRORS

Default.

The following sections are informative.

APPLICATION USAGE

None.

EXAMPLES

None.

RATIONALE

None.

FUTURE DIRECTIONS

None.

SEE ALSO

ln , *unlink()* , the System Interfaces volume of IEEE Std 1003.1-2001, *link()*

COPYRIGHT

Portions of this text are reprinted and reproduced in electronic form from IEEE Std 1003.1, 2003 Edition, Standard for Information Technology -- Portable Operating System Interface (POSIX), The Open Group Base Specifications Issue 6, Copyright (C) 2001-2003 by the Institute of Electrical and Electronics Engineers, Inc and The Open Group. In the event of any discrepancy between this version and the original IEEE and The Open Group Standard, the original IEEE and The Open Group Standard is the referee document. The original Standard can be obtained online at <http://www.open-group.org/unix/online.html> .

NAME

ln - link files

SYNOPSIS

ln [-fs] *source_file target_file*

ln [-fs] *source_file ... target_dir*

DESCRIPTION

In the first synopsis form, the *ln* utility shall create a new directory entry (link) at the destination path specified by the *target_file* operand. If the **-s** option is specified, a symbolic link shall be created for the file specified by the *source_file* operand. This first synopsis form shall be assumed when the final operand does not name an existing directory; if more than two operands are specified and the final is not an existing directory, an error shall result.

In the second synopsis form, the *ln* utility shall create a new directory entry (link), or if the **-s** option is specified a symbolic link, for each file specified by a *source_file* operand, at a destination path in the existing directory named by *target_dir*.

If the last operand specifies an existing file of a type not specified by the System Interfaces volume of IEEE Std 1003.1-2001, the behavior is implementation-defined.

The corresponding destination path for each *source_file* shall be the concatenation of the target directory pathname, a slash character, and the last pathname component of the *source_file*. The second synopsis form shall be assumed when the final operand names an existing directory.

For each *source_file*:

If the destination path exists: <ol type="a">

If the **-f** option is not specified, *ln* shall write a diagnostic message to standard error, do nothing more with the current *source_file*, and go on to any remaining *source_files*.

Actions shall be performed equivalent to the *unlink()* function defined in the System Interfaces volume of IEEE Std 1003.1-2001, called using *destination* as the *path* argument. If this fails for any reason, *ln* shall write a diagnostic message to standard error, do nothing more with the current *source_file*, and go on to any remaining *source_files*.

If the **-s** option is specified, *ln* shall create a symbolic link named by the destination path and containing as its pathname *source_file*. The *ln* utility shall do nothing more with *source_file* and shall go on to any remaining files.

If *source_file* is a symbolic link, actions shall be performed equivalent to the *link()* function using the object that *source_file* references as the *path1* argument and the destination path as the *path2* argument. The *ln* utility shall do nothing more with *source_file* and shall go on to any remaining files.

Actions shall be performed equivalent to the *link()* function defined in the System Interfaces volume of IEEE Std 1003.1-2001 using *source_file* as the *path1* argument, and the destination path as the *path2* argument.

OPTIONS

The *ln* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section 12.2, Utility Syntax Guidelines.

The following option shall be supported:

- f** Force existing destination pathnames to be removed to allow the link.
- s** Create symbolic links instead of hard links.

OPERANDS

The following operands shall be supported:

source_file

A pathname of a file to be linked. If the **-s** option is specified, no restrictions on the type of file or on its existence shall be made. If the **-s** option is not specified, whether a directory can be

linked is implementation-defined.

target_file

The pathname of the new directory entry to be created.

target_dir

A pathname of an existing directory in which the new directory entries are created.

STDIN

Not used.

INPUT FILES

None.

ENVIRONMENT VARIABLES

The following environment variables shall affect the execution of *ln*:

LANG Provide a default value for the internationalization variables that are unset or null. (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2, Internationalization Variables for the precedence of internationalization variables used to determine the values of locale categories.)

LC_ALL

If set to a non-empty string value, override the values of all the other internationalization variables.

LC_CTYPE

Determine the locale for the interpretation of sequences of bytes of text data as characters (for example, single-byte as opposed to multi-byte characters in arguments).

LC_MESSAGES

Determine the locale that should be used to affect the format and contents of diagnostic messages written to standard error.

NLSPATH

Determine the location of message catalogs for the processing of *LC_MESSAGES*.

ASYNCHRONOUS EVENTS

Default.

STDOUT

Not used.

STDERR

The standard error shall be used only for diagnostic messages.

OUTPUT FILES

None.

EXTENDED DESCRIPTION

None.

EXIT STATUS

The following exit values shall be returned:

- 0 All the specified files were linked successfully.
- >0 An error occurred.

CONSEQUENCES OF ERRORS

Default.

The following sections are informative.

APPLICATION USAGE

None.

EXAMPLES

None.

RATIONALE

Some historic versions of *ln* (including the one specified by the SVID) unlink the destination file, if it exists, by default. If the mode does not permit writing, these versions prompt for confirmation before attempting the unlink. In these versions the **-f** option causes *ln* not to attempt to prompt for confirmation.

This allows *ln* to succeed in creating links when the target file already exists, even if the file itself is not writable (although the directory must be). Early proposals specified this functionality.

This volume of IEEE Std 1003.1-2001 does not allow the *ln* utility to unlink existing destination paths by default for the following reasons:

The *ln* utility has historically been used to provide locking for shell applications, a usage that is incompatible with *ln* unlinking the destination path by default. There was no corresponding technical advantage to adding this functionality.

This functionality gave *ln* the ability to destroy the link structure of files, which changes the historical behavior of *ln*.

This functionality is easily replicated with a combination of *rm* and *ln*.

It is not historical practice in many systems; BSD and BSD-derived systems do not support this behavior. Unfortunately, whichever behavior is selected can cause scripts written expecting the other behavior to fail.

It is preferable that *ln* perform in the same manner as the *link()* function, which does not permit the target to exist already.

This volume of IEEE Std 1003.1-2001 retains the **-f** option to provide support for shell scripts depending on the SVID semantics. It seems likely that shell scripts would not be written to handle prompting by *ln* and would therefore have specified the **-f** option.

The **-f** option is an undocumented feature of many historical versions of the *ln* utility, allowing linking to directories. These versions require modification.

Early proposals of this volume of IEEE Std 1003.1-2001 also required a **-i** option, which behaved like the **-i** options in *cp* and *mv*, prompting for confirmation before unlinking existing files. This was not historical practice for the *ln* utility and has been omitted.

FUTURE DIRECTIONS

None.

SEE ALSO

chmod(), *find*, *pax*, *rm*, the System Interfaces volume of IEEE Std 1003.1-2001, *link()*, *unlink()*

COPYRIGHT

Portions of this text are reprinted and reproduced in electronic form from IEEE Std 1003.1, 2003 Edition, Standard for Information Technology -- Portable Operating System Interface (POSIX), The Open Group Base Specifications Issue 6, Copyright (C) 2001-2003 by the Institute of Electrical and Electronics Engineers, Inc and The Open Group. In the event of any discrepancy between this version and the original IEEE and The Open Group Standard, the original IEEE and The Open Group Standard is the referee document. The original Standard can be obtained online at <http://www.open-group.org/unix/online.html>.

NAME

logname - return the user's login name

SYNOPSIS

logname

DESCRIPTION

The *logname* utility shall write the user's login name to standard output. The login name shall be the string that would be returned by the *getlogin()* function defined in the System Interfaces volume of IEEE Std 1003.1-2001. Under the conditions where the *getlogin()* function would fail, the *logname* utility shall write a diagnostic message to standard error and exit with a non-zero exit status.

OPTIONS

None.

OPERANDS

None.

STDIN

Not used.

INPUT FILES

None.

ENVIRONMENT VARIABLES

The following environment variables shall affect the execution of *logname*:

LANG Provide a default value for the internationalization variables that are unset or null. (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2, Internationalization Variables for the precedence of internationalization variables used to determine the values of locale categories.)

LC_ALL

If set to a non-empty string value, override the values of all the other internationalization variables.

LC_CTYPE

Determine the locale for the interpretation of sequences of bytes of text data as characters (for example, single-byte as opposed to multi-byte characters in arguments).

LC_MESSAGES

Determine the locale that should be used to affect the format and contents of diagnostic messages written to standard error.

NLSPATH

Determine the location of message catalogs for the processing of *LC_MESSAGES*.

ASYNCHRONOUS EVENTS

Default.

STDOUT

The *logname* utility output shall be a single line consisting of the user's login name:

`"%s\n", <login name>`

STDERR

The standard error shall be used only for diagnostic messages.

OUTPUT FILES

None.

EXTENDED DESCRIPTION

None.

EXIT STATUS

The following exit values shall be returned:

- 0 Successful completion.
- >0 An error occurred.

CONSEQUENCES OF ERRORS

Default.

The following sections are informative.

APPLICATION USAGE

The *logname* utility explicitly ignores the *LOGNAME* environment variable because environment changes could produce erroneous results.

EXAMPLES

None.

RATIONALE

The **passwd** file is not listed as required because the implementation may have other means of mapping login names.

FUTURE DIRECTIONS

None.

SEE ALSO

id , *who* , the System Interfaces volume of IEEE Std 1003.1-2001, *getlogin()*

COPYRIGHT

Portions of this text are reprinted and reproduced in electronic form from IEEE Std 1003.1, 2003 Edition, Standard for Information Technology -- Portable Operating System Interface (POSIX), The Open Group Base Specifications Issue 6, Copyright (C) 2001-2003 by the Institute of Electrical and Electronics Engineers, Inc and The Open Group. In the event of any discrepancy between this version and the original IEEE and The Open Group Standard, the original IEEE and The Open Group Standard is the referee document. The original Standard can be obtained online at <http://www.open-group.org/unix/online.html> .

NAME

ls - list directory contents

SYNOPSIS

ls [-CFRacdilqrtu1][-H | -L][-fgmnopsx][file...]

DESCRIPTION

For each operand that names a file of a type other than directory or symbolic link to a directory, *ls* shall write the name of the file as well as any requested, associated information. For each operand that names a file of type directory, *ls* shall write the names of files contained within the directory as well as any requested, associated information. If one of the **-d**, **-F**, or **-l** options are specified, and one of the **-H** or **-L** options are not specified, for each operand that names a file of type symbolic link to a directory, *ls* shall write the name of the file as well as any requested, associated information. If none of the **-d**, **-F**, or **-l** options are specified, or the **-H** or **-L** options are specified, for each operand that names a file of type symbolic link to a directory, *ls* shall write the names of files contained within the directory as well as any requested, associated information.

If no operands are specified, *ls* shall write the contents of the current directory. If more than one operand is specified, *ls* shall write non-directory operands first; it shall sort directory and non-directory operands separately according to the collating sequence in the current locale.

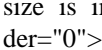
The *ls* utility shall detect infinite loops; that is, entering a previously visited directory that is an ancestor of the last file encountered. When it detects an infinite loop, *ls* shall write a diagnostic message to standard error and shall either recover its position in the hierarchy or terminate.

OPTIONS

The *ls* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section 12.2, Utility Syntax Guidelines.

The following options shall be supported:

- C** Write multi-text-column output with entries sorted down the columns, according to the collating sequence. The number of text columns and the column separator characters are unspecified, but should be adapted to the nature of the output device.
- F** Do not follow symbolic links named as operands unless the **-H** or **-L** options are specified. Write a slash (**'/'**) immediately after each pathname that is a directory, an asterisk (**'*'**) after each that is executable, a vertical bar (**'|'**) after each that is a FIFO, and an at sign (**'@'**) after each that is a symbolic link. For other file types, other symbols may be written.
- H** If a symbolic link referencing a file of type directory is specified on the command line, *ls* shall evaluate the file information and file type to be those of the file referenced by the link, and not the link itself; however, *ls* shall write the name of the link itself and not the file referenced by the link.
- L** Evaluate the file information and file type for all symbolic links (whether named on the command line or encountered in a file hierarchy) to be those of the file referenced by the link, and not the link itself; however, *ls* shall write the name of the link itself and not the file referenced by the link. When **-L** is used with **-l**, write the contents of symbolic links in the long format (see the STDOUT section).
- R** Recursively list subdirectories encountered.
- a** Write out all directory entries, including those whose names begin with a period (**'.'**). Entries beginning with a period shall not be written out unless explicitly referenced, the **-a** option is supplied, or an implementation-defined condition shall cause them to be written.
- c** Use time of last modification of the file status information (see *<sys/stat.h>* in the System Interfaces volume of IEEE Std 1003.1-2001) instead of last modification of the file itself for sorting (**-t**) or writing (**-l**).
- d** Do not follow symbolic links named as operands unless the **-H** or **-L** options are specified. Do not treat directories differently than other types of files. The use of **-d** with **-R** produces unspecified results.

- f** Force each argument to be interpreted as a directory and list the name found in each slot. This option shall turn off **-l**, **-t**, **-s**, and **-r**, and shall turn on **-a**; the order is the order in which entries appear in the directory.
- g** The same as **-l**, except that the owner shall not be written.
- i** For each file, write the file's file serial number (see *stat()* in the System Interfaces volume of IEEE Std 1003.1-2001).
- l** (The letter ell.) Do not follow symbolic links named as operands unless the **-H** or **-L** options are specified. Write out in long format (see the STDOUT section). When **-l** (ell) is specified, **-l** (one) shall be assumed.
- m** Stream output format; list files across the page, separated by commas.
- n** The same as **-l**, except that the owner's UID and GID numbers shall be written, rather than the associated character strings.
- o** The same as **-l**, except that the group shall not be written.
- p** Write a slash (`'/'`) after each filename if that file is a directory.
- q** Force each instance of non-printable filename characters and `<tab>`s to be written as the question-mark (`'?'`) character. Implementations may provide this option by default if the output is to a terminal device.
- r** Reverse the order of the sort to get reverse collating sequence or oldest first.
- s** Indicate the total number of file system blocks consumed by each file displayed. The block size is implementation-defined.  ``
- t** Sort with the primary key being time modified (most recently modified first) and the secondary key being filename in the collating sequence.
- u** Use time of last access (see *<sys/stat.h>*) instead of last modification of the file for sorting (**-t**) or writing (**-l**).
- x** The same as **-C**, except that the multi-text-column output is produced with entries sorted across, rather than down, the columns.
- 1** (The numeric digit one.) Force output to be one entry per line.

Specifying more than one of the options in the following mutually-exclusive pairs shall not be considered an error: **-C** and **-l** (ell), **-m** and **-l** (ell), **-x** and **-l** (ell), **-C** and **-l** (one), **-H** and **-L**, **-c** and **-u**. The last option specified in each pair shall determine the output format.

OPERANDS

The following operand shall be supported:

- file* A pathname of a file to be written. If the file specified is not found, a diagnostic message shall be output on standard error.

STDIN

Not used.

INPUT FILES

None.

ENVIRONMENT VARIABLES

The following environment variables shall affect the execution of *ls*:

COLUMNS

Determine the user's preferred column position width for writing multiple text-column output. If this variable contains a string representing a decimal integer, the *ls* utility shall calculate how many pathname text columns to write (see **-C**) based on the width provided. If *COLUMNS* is not set or invalid, an implementation-defined number of column positions shall be assumed, based on the implementation's knowledge of the output device. The column width chosen to write the names of files in any given directory shall be constant. Filenames shall not

be truncated to fit into the multiple text-column output.

LANG Provide a default value for the internationalization variables that are unset or null. (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2, Internationalization Variables for the precedence of internationalization variables used to determine the values of locale categories.)

LC_ALL

If set to a non-empty string value, override the values of all the other internationalization variables.

LC_COLLATE

Determine the locale for character collation information in determining the pathname collation sequence.

LC_CTYPE

Determine the locale for the interpretation of sequences of bytes of text data as characters (for example, single-byte as opposed to multi-byte characters in arguments) and which characters are defined as printable (character class **print**).

LC_MESSAGES

Determine the locale that should be used to affect the format and contents of diagnostic messages written to standard error.

LC_TIME

Determine the format and contents for date and time strings written by *ls*.

NLSPATH

Determine the location of message catalogs for the processing of *LC_MESSAGES*.

TZ

Determine the timezone for date and time strings written by *ls*. If *TZ* is unset or null, an unspecified default timezone shall be used.

ASYNCHRONOUS EVENTS

Default.

STDOUT

The default format shall be to list one entry per line to standard output; the exceptions are to terminals or when one of the **-C**, **-m**, or **-x** options is specified. If the output is to a terminal, the format is implementation-defined.

When **-m** is specified, the format used shall be:

```
"%s, %s, ...\n", <filename1>, <filename2>
```

where the largest number of filenames shall be written without exceeding the length of the line.

If the **-i** option is specified, the file's file serial number (see <sys/stat.h>) shall be written in the following format before any other output for the corresponding entry:

```
%u ", <file serial number>
```

If the **-l** option is specified without **-L**, the following information shall be written:

```
"%s %u %s %s %u %s %s\n", <file mode>, <number of links>,
    <owner name>, <group name>, <number of bytes in the file>,
    <date and time>, <pathname>
```

If the file is a symbolic link, this information shall be about the link itself and the <pathname> field shall be of the form:

```
"%s -> %s", <pathname of link>, <contents of link>
```

If both **-l** and **-L** are specified, the following information shall be written:

```
"%s %u %s %s %u %s %s\n", <file mode>, <number of links>,
    <owner name>, <group name>, <number of bytes in the file>,
    <date and time>, <pathname of link>
```

where all fields except *<pathname of link>* shall be for the file resolved from the symbolic link.

The **-g**, **-n**, and **-o** options use the same format as **-l**, but with omitted items and their associated *<blank>*s. See the OPTIONS section.

In both the preceding **-l** forms, if *<owner name>* or *<group name>* cannot be determined, or if **-n** is given, they shall be replaced with their associated numeric values using the format **%u**.

The *<date and time>* field shall contain the appropriate date and timestamp of when the file was last modified. In the POSIX locale, the field shall be the equivalent of the output of the following *date* command:

```
date "+%b %e %H:%M"
```

if the file has been modified in the last six months, or:

```
date "+%b %e %Y"
```

(where two *<space>*s are used between **%e** and **%Y**) if the file has not been modified in the last six months or if the modification date is in the future, except that, in both cases, the final *<newline>* produced by *date* shall not be included and the output shall be as if the *date* command were executed at the time of the last modification date of the file rather than the current time. When the *LC_TIME* locale category is not set to the POSIX locale, a different format and order of presentation of this field may be used.

If the file is a character special or block special file, the size of the file may be replaced with implementation-defined information associated with the device in question.

If the pathname was specified as a *file* operand, it shall be written as specified.

The file mode written under the **-l**, **-g**, **-n**, and **-o** options shall consist of the following format:

```
"%c%s%s%s%c", <entry type>, <owner permissions>,
    <group permissions>, <other permissions>,
    <optional alternate access method flag>
```

The *<optional alternate access method flag>* shall be a single *<space>* if there is no alternate or additional access control method associated with the file; otherwise, a printable character shall be used.

The *<entry type>* character shall describe the type of file, as follows:

- d** Directory.
- b** Block special file.
- c** Character special file.
- l** (ell) Symbolic link.
- p** FIFO.
- Regular file.

Implementations may add other characters to this list to represent other implementation-defined file types.

The next three fields shall be three characters each:

<owner permissions>

Permissions for the file owner class (see the Base Definitions volume of IEEE Std 1003.1-2001, Section 4.4, File Access Permissions).

<group permissions>

Permissions for the file group class.

<other permissions>

Permissions for the file other class.

Each field shall have three character positions:

If 'r', the file is readable; if '-', the file is not readable.

If 'w', the file is writable; if '-', the file is not writable.

The first of the following that applies:

- S** If in *<owner permissions>*, the file is not executable and set-user-ID mode is set. If in *<group permissions>*, the file is not executable and set-group-ID mode is set.
- s** If in *<owner permissions>*, the file is executable and set-user-ID mode is set. If in *<group permissions>*, the file is executable and set-group-ID mode is set.
- T** If in *<other permissions>* and the file is a directory, search permission is not granted to others, and the restricted deletion flag is set.
- t** If in *<other permissions>* and the file is a directory, search permission is granted to others, and the restricted deletion flag is set.
- x** The file is executable or the directory is searchable.
- None of the attributes of 'S', 's', 'T', 't', or 'x' applies.

Implementations may add other characters to this list for the third character position. Such additions shall, however, be written in lowercase if the file is executable or searchable, and in uppercase if it is not.

If any of the **-l**, **-g**, **-n**, **-o**, or **-s** options is specified, each list of files within the directory shall be preceded by a status line indicating the number of file system blocks occupied by files in the directory in 512-byte units, rounded up to the next integral number of units, if necessary. In the POSIX locale, the format shall be:

"total %u\n", <number of units in the directory>

If more than one directory, or a combination of non-directory files and directories are written, either as a result of specifying multiple operands, or the **-R** option, each list of files within a directory shall be preceded by:

"\n%s:\n", <directory name>

If this string is the first thing to be written, the first <newline> shall not be written. This output shall precede the number of units in the directory.

If the **-s** option is given, each file shall be written with the number of blocks used by the file. Along with **-C**, **-l**, **-m**, or **-x**, the number and a <space> shall precede the filename; with **-g**, **-l**, **-n**, or **-o**, they shall precede each line describing a file.

STDERR

The standard error shall be used only for diagnostic messages.

OUTPUT FILES

None.

EXTENDED DESCRIPTION

None.

EXIT STATUS

The following exit values shall be returned:

- 0 Successful completion.
- >0 An error occurred.

CONSEQUENCES OF ERRORS

Default.

The following sections are informative.

APPLICATION USAGE

Many implementations use the equal sign ('=') to denote sockets bound to the file system for the **-F** option. Similarly, many historical implementations use the 's' character to denote sockets as the entry type characters for the **-l** option.

It is difficult for an application to use every part of the file modes field of *ls -l* in a portable manner. Certain file types and executable bits are not guaranteed to be exactly as shown, as implementations may have extensions. Applications can use this field to pass directly to a user printout or prompt, but actions based on its contents should generally be deferred, instead, to the *test* utility.

The output of *ls* (with the **-l** and related options) contains information that logically could be used by utilities such as *chmod* and *touch* to restore files to a known state. However, this information is presented in a format that cannot be used directly by those utilities or be easily translated into a format that can be used. A character has been added to the end of the permissions string so that applications at least have an indication that they may be working in an area they do not understand instead of assuming that they can translate the permissions string into something that can be used. Future issues or related documents may define one or more specific characters to be used based on different standard additional or alternative access control mechanisms.

As with many of the utilities that deal with filenames, the output of *ls* for multiple files or in one of the long listing formats must be used carefully on systems where filenames can contain embedded white space. Systems and system administrators should institute policies and user training to limit the use of such filenames.

The number of disk blocks occupied by the file that it reports varies depending on underlying file system type, block size units reported, and the method of calculating the number of blocks. On some file system types, the number is the actual number of blocks occupied by the file (counting indirect blocks and ignoring holes in the file); on others it is calculated based on the file size (usually making an allowance for indirect blocks, but ignoring holes).

EXAMPLES

An example of a small directory tree being fully listed with *ls -laRF a* in the POSIX locale:

```
total 11
drwxr-xr-x 3 hlj  prog    64 Jul  4 12:07 ./
drwxrwxrwx 4 hlj  prog   3264 Jul  4 12:09 ../
drwxr-xr-x 2 hlj  prog    48 Jul  4 12:07 b/
-rwxr--r-- 1 hlj  prog   572 Jul  4 12:07 foo*
```

```
a/b:
total 4
drwxr-xr-x 2 hlj  prog    48 Jul  4 12:07 ./
drwxr-xr-x 3 hlj  prog    64 Jul  4 12:07 ../
-rw-r--r-- 1 hlj  prog   700 Jul  4 12:07 bar
```

RATIONALE

Some historical implementations of the *ls* utility show all entries in a directory except dot and dot-dot when a superuser invokes *ls* without specifying the **-a** option. When "normal" users invoke *ls* without specifying **-a**, they should not see information about any files with names beginning with a period unless they were named as *file* operands.

Implementations are expected to traverse arbitrary depths when processing the **-R** option. The only limitation on depth should be based on running out of physical storage for keeping track of untraversed directories.

The **-1** (one) option was historically found in BSD and BSD-derived implementations only. It is required in this volume of IEEE Std 1003.1-2001 so that conforming applications might ensure that output is one entry per line, even if the output is to a terminal.

Generally, this volume of IEEE Std 1003.1-2001 is silent about what happens when options are given multiple times. In the cases of **-C**, **-l**, and **-1**, however, it does specify the results of these overlapping options. Since *ls* is one of the most aliased commands, it is important that the implementation perform intuitively. For example, if the alias were:

```
alias ls="ls -C"
```

and the user typed *ls -1*, single-text-column output should result, not an error.

The BSD *ls* provides a **-A** option (like **-a**, but dot and dot-dot are not written out). The small difference from **-a** did not seem important enough to require both.

Implementations may make **-q** the default for terminals to prevent trojan horse attacks on terminals with special escape sequences. This is not required because:

Some control characters may be useful on some terminals; for example, a system might write them as "\001" or "^A".

Special behavior for terminals is not relevant to applications portability.

An early proposal specified that the optional alternate access method flag had to be '+' if there was an alternate access method used on the file or <space> if there was not. This was changed to be <space> if there is not and a single printable character if there is. This was done for three reasons:

There are historical implementations using characters other than '+'.

There are implementations that vary this character used in that position to distinguish between various alternate access methods in use.

The standard developers did not want to preclude future specifications that might need a way to specify more than one alternate access method.

Nonetheless, implementations providing a single alternate access method are encouraged to use '+'.

In an early proposal, the units used to specify the number of blocks occupied by files in a directory in an *ls -l* listing were implementation-defined. This was because BSD systems have historically used 1024-byte units and System V systems have historically used 512-byte units. It was pointed out by BSD developers that their system has used 512-byte units in some places and 1024-byte units in other places. (System V has consistently used 512.) Therefore, this volume of IEEE Std 1003.1-2001 usually specifies 512. Future releases of BSD are expected to consistently provide 512 bytes as a default with a way of specifying 1024-byte units where appropriate.

The <date and time> field in the **-l** format is specified only for the POSIX locale. As noted, the format can be different in other locales. No mechanism for defining this is present in this volume of IEEE Std 1003.1-2001, as the appropriate vehicle is a messaging system; that is, the format should be specified as a "message".

FUTURE DIRECTIONS

The **-s** uses implementation-defined units and cannot be used portably; it may be withdrawn in a future version.

SEE ALSO

chmod(), *find*, the System Interfaces volume of IEEE Std 1003.1-2001, *stat()*, the Base Definitions volume of IEEE Std 1003.1-2001, <sys/stat.h>

COPYRIGHT

Portions of this text are reprinted and reproduced in electronic form from IEEE Std 1003.1, 2003 Edition, Standard for Information Technology -- Portable Operating System Interface (POSIX), The Open Group Base Specifications Issue 6, Copyright (C) 2001-2003 by the Institute of Electrical and Electronics Engineers, Inc and The Open Group. In the event of any discrepancy between this version and the original IEEE and The Open Group Standard, the original IEEE and The Open Group Standard is the referee document. The original Standard can be obtained online at <http://www.open-group.org/unix/online.html> .

NAME

mkdir - make directories

SYNOPSIS

mkdir [-p][-m *mode*] *dir*...

DESCRIPTION

The *mkdir* utility shall create the directories specified by the operands, in the order specified.

For each *dir* operand, the *mkdir* utility shall perform actions equivalent to the *mkdir*() function defined in the System Interfaces volume of IEEE Std 1003.1-2001, called with the following arguments:

The *dir* operand is used as the *path* argument.

The value of the bitwise-inclusive OR of S_IRWXU, S_IRWXG, and S_IRWXO is used as the *mode* argument. (If the -m option is specified, the *mode* option-argument overrides this default.)

OPTIONS

The *mkdir* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section 12.2, Utility Syntax Guidelines.

The following options shall be supported:

-m *mode*

Set the file permission bits of the newly-created directory to the specified *mode* value. The *mode* option-argument shall be the same as the *mode* operand defined for the *chmod* utility. In the *symbolic_mode* strings, the *op* characters '+' and '-' shall be interpreted relative to an assumed initial mode of *a= rwx*; '+' shall add permissions to the default mode, '-' shall delete permissions from the default mode.

-p Create any missing intermediate pathname components.

For each *dir* operand that does not name an existing directory, effects equivalent to those caused by the following command shall occur:

```
mkdir -p -m $(umask -S),u+wx $(dirname dir) &&
mkdir [-m mode] dir
```

where the -m *mode* option represents that option supplied to the original invocation of *mkdir*, if any.

Each *dir* operand that names an existing directory shall be ignored without error.

OPERANDS

The following operand shall be supported:

dir A pathname of a directory to be created.

STDIN

Not used.

INPUT FILES

None.

ENVIRONMENT VARIABLES

The following environment variables shall affect the execution of *mkdir*:

LANG Provide a default value for the internationalization variables that are unset or null. (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2, Internationalization Variables for the precedence of internationalization variables used to determine the values of locale categories.)

LC_ALL

If set to a non-empty string value, override the values of all the other internationalization variables.

LC_CTYPE

Determine the locale for the interpretation of sequences of bytes of text data as characters (for example, single-byte as opposed to multi-byte characters in arguments).

LC_MESSAGES

Determine the locale that should be used to affect the format and contents of diagnostic messages written to standard error.

NLSPATH

Determine the location of message catalogs for the processing of *LC_MESSAGES*.

ASYNCHRONOUS EVENTS

Default.

STDOUT

Not used.

STDERR

The standard error shall be used only for diagnostic messages.

OUTPUT FILES

None.

EXTENDED DESCRIPTION

None.

EXIT STATUS

The following exit values shall be returned:

- 0 All the specified directories were created successfully or the **-p** option was specified and all the specified directories now exist.
- >0 An error occurred.

CONSEQUENCES OF ERRORS

Default.

The following sections are informative.

APPLICATION USAGE

The default file mode for directories is *a= rwx* (777 on most systems) with selected permissions removed in accordance with the file mode creation mask. For intermediate pathname components created by *mkdir*, the mode is the default modified by *u+ wx* so that the subdirectories can always be created regardless of the file mode creation mask; if different ultimate permissions are desired for the intermediate directories, they can be changed afterwards with *chmod*.

Note that some of the requested directories may have been created even if an error occurs.

EXAMPLES

None.

RATIONALE

The System V **-m** option was included to control the file mode.

The System V **-p** option was included to create any needed intermediate directories and to complement the functionality provided by *rmdir* for removing directories in the path prefix as they become empty. Because no error is produced if any path component already exists, the **-p** option is also useful to ensure that a particular directory exists.

The functionality of *mkdir* is described substantially through a reference to the *mkdir()* function in the System Interfaces volume of IEEE Std 1003.1-2001. For example, by default, the mode of the directory is affected by the file mode creation mask in accordance with the specified behavior of the *mkdir()* function. In this way, there is less duplication of effort required for describing details of the directory creation.

FUTURE DIRECTIONS

None.

SEE ALSO

chmod() , *rm* , *rmdir()* , *umask()* , the System Interfaces volume of IEEE Std 1003.1-2001, *mkdir()*

COPYRIGHT

Portions of this text are reprinted and reproduced in electronic form from IEEE Std 1003.1, 2003 Edition, Standard for Information Technology -- Portable Operating System Interface (POSIX), The Open Group Base Specifications Issue 6, Copyright (C) 2001-2003 by the Institute of Electrical and Electronics Engineers, Inc and The Open Group. In the event of any discrepancy between this version and the original IEEE and The Open Group Standard, the original IEEE and The Open Group Standard is the referee document. The original Standard can be obtained online at <http://www.open-group.org/unix/online.html> .

NAME

mkfifo - make FIFO special files

SYNOPSIS

mkfifo [-m *mode*] *file*...

DESCRIPTION

The *mkfifo* utility shall create the FIFO special files specified by the operands, in the order specified.

For each *file* operand, the *mkfifo* utility shall perform actions equivalent to the *mkfifo*() function defined in the System Interfaces volume of IEEE Std 1003.1-2001, called with the following arguments:

The *file* operand is used as the *path* argument.

The value of the bitwise-inclusive OR of S_IRUSR, S_IWUSR, S_IRGRP, S_IWGRP, S_IROTH, and S_IWOTH is used as the *mode* argument. (If the -m option is specified, the value of the *mkfifo*() *mode* argument is unspecified, but the FIFO shall at no time have permissions less restrictive than the -m *mode* option-argument.)

OPTIONS

The *mkfifo* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section 12.2, Utility Syntax Guidelines.

The following option shall be supported:

-m *mode*

Set the file permission bits of the newly-created FIFO to the specified *mode* value. The *mode* option-argument shall be the same as the *mode* operand defined for the *chmod* utility. In the *symbolic_mode* strings, the *op* characters '+' and '-' shall be interpreted relative to an assumed initial mode of *a= rw*.

OPERANDS

The following operand shall be supported:

file A pathname of the FIFO special file to be created.

STDIN

Not used.

INPUT FILES

None.

ENVIRONMENT VARIABLES

The following environment variables shall affect the execution of *mkfifo*:

LANG Provide a default value for the internationalization variables that are unset or null. (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2, Internationalization Variables for the precedence of internationalization variables used to determine the values of locale categories.)

LC_ALL

If set to a non-empty string value, override the values of all the other internationalization variables.

LC_CTYPE

Determine the locale for the interpretation of sequences of bytes of text data as characters (for example, single-byte as opposed to multi-byte characters in arguments).

LC_MESSAGES

Determine the locale that should be used to affect the format and contents of diagnostic messages written to standard error.

NLSPATH

Determine the location of message catalogs for the processing of *LC_MESSAGES*.

ASYNCHRONOUS EVENTS

Default.

STDOUT

Not used.

STDERR

The standard error shall be used only for diagnostic messages.

OUTPUT FILES

None.

EXTENDED DESCRIPTION

None.

EXIT STATUS

The following exit values shall be returned:

- 0 All the specified FIFO special files were created successfully.
- >0 An error occurred.

CONSEQUENCES OF ERRORS

Default.

The following sections are informative.

APPLICATION USAGE

None.

EXAMPLES

None.

RATIONALE

This utility was added to permit shell applications to create FIFO special files.

The **-m** option was added to control the file mode, for consistency with the similar functionality provided by the *mkdir* utility.

Early proposals included a **-p** option similar to the *mkdir -p* option that created intermediate directories leading up to the FIFO specified by the final component. This was removed because it is not commonly needed and is not common practice with similar utilities.

The functionality of *mkfifo* is described substantially through a reference to the *mkfifo()* function in the System Interfaces volume of IEEE Std 1003.1-2001. For example, by default, the mode of the FIFO file is affected by the file mode creation mask in accordance with the specified behavior of the *mkfifo()* function. In this way, there is less duplication of effort required for describing details of the file creation.

FUTURE DIRECTIONS

None.

SEE ALSO

chmod() , *umask()* , the System Interfaces volume of IEEE Std 1003.1-2001, *mkfifo()*

COPYRIGHT

Portions of this text are reprinted and reproduced in electronic form from IEEE Std 1003.1, 2003 Edition, Standard for Information Technology -- Portable Operating System Interface (POSIX), The Open Group Base Specifications Issue 6, Copyright (C) 2001-2003 by the Institute of Electrical and Electronics Engineers, Inc and The Open Group. In the event of any discrepancy between this version and the original IEEE and The Open Group Standard, the original IEEE and The Open Group Standard is the referee document. The original Standard can be obtained online at <http://www.open-group.org/unix/online.html> .

NAME

mv - move files

SYNOPSIS

mv [-fi] *source_file target_file*

mv [-fi] *source_file... target_file*

DESCRIPTION

In the first synopsis form, the *mv* utility shall move the file named by the *source_file* operand to the destination specified by the *target_file*. This first synopsis form is assumed when the final operand does not name an existing directory and is not a symbolic link referring to an existing directory.

In the second synopsis form, *mv* shall move each file named by a *source_file* operand to a destination file in the existing directory named by the *target_dir* operand, or referenced if *target_dir* is a symbolic link referring to an existing directory. The destination path for each *source_file* shall be the concatenation of the target directory, a single slash character, and the last pathname component of the *source_file*. This second form is assumed when the final operand names an existing directory.

If any operand specifies an existing file of a type not specified by the System Interfaces volume of IEEE Std 1003.1-2001, the behavior is implementation-defined.

For each *source_file* the following steps shall be taken:

If the destination path exists, the **-f** option is not specified, and either of the following conditions is true: <ol type="a">

The permissions of the destination path do not permit writing and the standard input is a terminal.

The **-i** option is specified.

the *mv* utility shall write a prompt to standard error and read a line from standard input. If the response is not affirmative, *mv* shall do nothing more with the current *source_file* and go on to any remaining *source_files*.

The *mv* utility shall perform actions equivalent to the *rename()* function defined in the System Interfaces volume of IEEE Std 1003.1-2001, called with the following arguments: <ol type="a">

The *source_file* operand is used as the *old* argument.

The destination path is used as the *new* argument.

If this succeeds, *mv* shall do nothing more with the current *source_file* and go on to any remaining *source_files*. If this fails for any reasons other than those described for the *errno* [EXDEV] in the System Interfaces volume of IEEE Std 1003.1-2001, *mv* shall write a diagnostic message to standard error, do nothing more with the current *source_file*, and go on to any remaining *source_files*.

If the destination path exists, and it is a file of type directory and *source_file* is not a file of type directory, or it is a file not of type directory and *source_file* is a file of type directory, *mv* shall write a diagnostic message to standard error, do nothing more with the current *source_file*, and go on to any remaining *source_files*.

If the destination path exists, *mv* shall attempt to remove it. If this fails for any reason, *mv* shall write a diagnostic message to standard error, do nothing more with the current *source_file*, and go on to any remaining *source_files*.

The file hierarchy rooted in *source_file* shall be duplicated as a file hierarchy rooted in the destination path. If *source_file* or any of the files below it in the hierarchy are symbolic links, the links themselves shall be duplicated, including their contents, rather than any files to which they refer. The following characteristics of each file in the file hierarchy shall be duplicated:

The time of last data modification and time of last access

The user ID and group ID

The file mode

If the user ID, group ID, or file mode of a regular file cannot be duplicated, the file mode bits S_ISUID

and `S_ISGID` shall not be duplicated.

When files are duplicated to another file system, the implementation may require that the process invoking *mv* has read access to each file being duplicated.

If the duplication of the file hierarchy fails for any reason, *mv* shall write a diagnostic message to standard error, do nothing more with the current *source_file*, and go on to any remaining *source_files*.

If the duplication of the file characteristics fails for any reason, *mv* shall write a diagnostic message to standard error, but this failure shall not cause *mv* to modify its exit status.

The file hierarchy rooted in *source_file* shall be removed. If this fails for any reason, *mv* shall write a diagnostic message to the standard error, do nothing more with the current *source_file*, and go on to any remaining *source_files*.

OPTIONS

The *mv* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section 12.2, Utility Syntax Guidelines.

The following options shall be supported:

- f** Do not prompt for confirmation if the destination path exists. Any previous occurrence of the **-i** option is ignored.
- i** Prompt for confirmation if the destination path exists. Any previous occurrence of the **-f** option is ignored.

Specifying more than one of the **-f** or **-i** options shall not be considered an error. The last option specified shall determine the behavior of *mv*.

OPERANDS

The following operands shall be supported:

source_file

A pathname of a file or directory to be moved.

target_file

A new pathname for the file or directory being moved.

target_dir

A pathname of an existing directory into which to move the input files.

STDIN

The standard input shall be used to read an input line in response to each prompt specified in the STDERR section. Otherwise, the standard input shall not be used.

INPUT FILES

The input files specified by each *source_file* operand can be of any file type.

ENVIRONMENT VARIABLES

The following environment variables shall affect the execution of *mv*:

LANG Provide a default value for the internationalization variables that are unset or null. (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2, Internationalization Variables for the precedence of internationalization variables used to determine the values of locale categories.)

LC_ALL

If set to a non-empty string value, override the values of all the other internationalization variables.

LC_COLLATE

Determine the locale for the behavior of ranges, equivalence classes, and multi-character collating elements used in the extended regular expression defined for the **yesexpr** locale keyword in the *LC_MESSAGES* category.

LC_CTYPE

Determine the locale for the interpretation of sequences of bytes of text data as characters (for example, single-byte as opposed to multi-byte characters in arguments and input files), the behavior of character classes used in the extended regular expression defined for the **yesexpr** locale keyword in the **LC_MESSAGES** category.

LC_MESSAGES

Determine the locale for the processing of affirmative responses that should be used to affect the format and contents of diagnostic messages written to standard error.

NLSPATH

Determine the location of message catalogs for the processing of **LC_MESSAGES**.

ASYNCHRONOUS EVENTS

Default.

STDOUT

Not used.

STDERR

Prompts shall be written to the standard error under the conditions specified in the DESCRIPTION section. The prompts shall contain the destination pathname, but their format is otherwise unspecified. Otherwise, the standard error shall be used only for diagnostic messages.

OUTPUT FILES

The output files may be of any file type.

EXTENDED DESCRIPTION

None.

EXIT STATUS

The following exit values shall be returned:

- 0 All input files were moved successfully.
- >0 An error occurred.

CONSEQUENCES OF ERRORS

If the copying or removal of *source_file* is prematurely terminated by a signal or error, *mv* may leave a partial copy of *source_file* at the source or destination. The *mv* utility shall not modify both *source_file* and the destination path simultaneously; termination at any point shall leave either *source_file* or the destination path complete.

The following sections are informative.

APPLICATION USAGE

Some implementations mark for update the *st_ctime* field of renamed files and some do not. Applications which make use of the *st_ctime* field may behave differently with respect to renamed files unless they are designed to allow for either behavior.

EXAMPLES

If the current directory contains only files **a** (of any type defined by the System Interfaces volume of IEEE Std 1003.1-2001), **b** (also of any type), and a directory **c**:

```
mv a b c
mv c d
```

results with the original files **a** and **b** residing in the directory **d** in the current directory.

RATIONALE

Early proposals diverged from the SVID and BSD historical practice in that they required that when the destination path exists, the **-f** option is not specified, and input is not a terminal, *mv* fails. This was done for compatibility with *cp*. The current text returns to historical practice. It should be noted that this is consistent with the *rename()* function defined in the System Interfaces volume of IEEE Std 1003.1-2001, which does not require write permission on the target.

For absolute clarity, paragraph (1), describing the behavior of *mv* when prompting for confirmation, should be interpreted in the following manner:

**if (exists AND (NOT f_option) AND
(not_writable AND input_is_terminal) OR i_option))**

The **-i** option exists on BSD systems, giving applications and users a way to avoid accidentally unlinking files when moving others. When the standard input is not a terminal, the 4.3 BSD *mv* deletes all existing destination paths without prompting, even when **-i** is specified; this is inconsistent with the behavior of the 4.3 BSD *cp* utility, which always generates an error when the file is unwritable and the standard input is not a terminal. The standard developers decided that use of **-i** is a request for interaction, so when the destination path exists, the utility takes instructions from whatever responds to standard input.

The *rename()* function is able to move directories within the same file system. Some historical versions of *mv* have been able to move directories, but not to a different file system. The standard developers considered that this was an annoying inconsistency, so this volume of IEEE Std 1003.1-2001 requires directories to be able to be moved even across file systems. There is no **-R** option to confirm that moving a directory is actually intended, since such an option was not required for moving directories in historical practice. Requiring the application to specify it sometimes, depending on the destination, seemed just as inconsistent. The semantics of the *rename()* function were preserved as much as possible. For example, *mv* is not permitted to "rename" files to or from directories, even though they might be empty and removable.

Historic implementations of *mv* did not exit with a non-zero exit status if they were unable to duplicate any file characteristics when moving a file across file systems, nor did they write a diagnostic message for the user. The former behavior has been preserved to prevent scripts from breaking; a diagnostic message is now required, however, so that users are alerted that the file characteristics have changed.

The exact format of the interactive prompts is unspecified. Only the general nature of the contents of prompts are specified because implementations may desire more descriptive prompts than those used on historical implementations. Therefore, an application not using the **-f** option or using the **-i** option relies on the system to provide the most suitable dialog directly with the user, based on the behavior specified.

When *mv* is dealing with a single file system and *source_file* is a symbolic link, the link itself is moved as a consequence of the dependence on the *rename()* functionality, per the DESCRIPTION. Across file systems, this has to be made explicit.

FUTURE DIRECTIONS

None.

SEE ALSO

cp, *ln*, the System Interfaces volume of IEEE Std 1003.1-2001, *rename()*

COPYRIGHT

Portions of this text are reprinted and reproduced in electronic form from IEEE Std 1003.1, 2003 Edition, Standard for Information Technology -- Portable Operating System Interface (POSIX), The Open Group Base Specifications Issue 6, Copyright (C) 2001-2003 by the Institute of Electrical and Electronics Engineers, Inc and The Open Group. In the event of any discrepancy between this version and the original IEEE and The Open Group Standard, the original IEEE and The Open Group Standard is the referee document. The original Standard can be obtained online at <http://www.open-group.org/unix/online.html>.

NAME

nice - invoke a utility with an altered nice value

SYNOPSIS

nice [-n *increment*] *utility* [*argument...*]

DESCRIPTION

The *nice* utility shall invoke a utility, requesting that it be run with a different nice value (see the Base Definitions volume of IEEE Std 1003.1-2001, Section 3.239, Nice Value). With no options and only if the user has appropriate privileges, the executed utility shall be run with a nice value that is some implementation-defined quantity less than or equal to the nice value of the current process. If the user lacks appropriate privileges to affect the nice value in the requested manner, the *nice* utility shall not affect the nice value; in this case, a warning message may be written to standard error, but this shall not prevent the invocation of *utility* or affect the exit status.

OPTIONS

The *nice* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section 12.2, Utility Syntax Guidelines.

The following option is supported:

-n *increment*

A positive or negative decimal integer which shall have the same effect on the execution of the utility as if the utility had called the *nice*() function with the numeric value of the *increment* option-argument.

OPERANDS

The following operands shall be supported:

utility The name of a utility that is to be invoked. If the *utility* operand names any of the special built-in utilities in *Special Built-In Utilities*, the results are undefined.

argument

Any string to be supplied as an argument when invoking the utility named by the *utility* operand.

STDIN

Not used.

INPUT FILES

None.

ENVIRONMENT VARIABLES

The following environment variables shall affect the execution of *nice*:

LANG Provide a default value for the internationalization variables that are unset or null. (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2, Internationalization Variables for the precedence of internationalization variables used to determine the values of locale categories.)

LC_ALL

If set to a non-empty string value, override the values of all the other internationalization variables.

LC_CTYPE

Determine the locale for the interpretation of sequences of bytes of text data as characters (for example, single-byte as opposed to multi-byte characters in arguments).

LC_MESSAGES

Determine the locale that should be used to affect the format and contents of diagnostic messages written to standard error.

NLSPATH

Determine the location of message catalogs for the processing of **LC_MESSAGES**.

PATH Determine the search path used to locate the utility to be invoked. See the Base Definitions volume of IEEE Std 1003.1-2001, Chapter 8, Environment Variables.

ASYNCHRONOUS EVENTS

Default.

STDOUT

Not used.

STDERR

The standard error shall be used only for diagnostic messages.

OUTPUT FILES

None.

EXTENDED DESCRIPTION

None.

EXIT STATUS

If *utility* is invoked, the exit status of *nice* shall be the exit status of *utility*; otherwise, the *nice* utility shall exit with one of the following values:

- 1-125 An error occurred in the *nice* utility.
- 126 The utility specified by *utility* was found but could not be invoked.
- 127 The utility specified by *utility* could not be found.

CONSEQUENCES OF ERRORS

Default.

The following sections are informative.

APPLICATION USAGE

The only guaranteed portable uses of this utility are:

nice utility

Run *utility* with the default lower nice value.

nice -n <positive integer> utility

Run *utility* with a lower nice value.

On some implementations they have no discernible effect on the invoked utility and on some others they are exactly equivalent.

Historical systems have frequently supported the *<positive integer>* up to 20. Since there is no error penalty associated with guessing a number that is too high, users without access to the system conformance document (to see what limits are actually in place) could use the historical 1 to 20 range or attempt to use very large numbers if the job should be truly low priority.

The nice value of a process can be displayed using the command:

ps -o nice

The *command*, *env*, *nice*, *nohup*, *time*, and *xargs* utilities have been specified to use exit code 127 if an error occurs so that applications can distinguish "failure to find a utility" from "invoked utility exited with an error indication". The value 127 was chosen because it is not commonly used for other meanings; most utilities use small values for "normal error conditions" and the values above 128 can be confused with termination due to receipt of a signal. The value 126 was chosen in a similar manner to indicate that the utility could be found, but not invoked. Some scripts produce meaningful error messages differentiating the 126 and 127 cases. The distinction between exit codes 126 and 127 is based on KornShell practice that uses 127 when all attempts to *exec* the utility fail with [ENOENT], and uses 126 when any attempt to *exec* the utility fails for any other reason.

EXAMPLES

None.

RATIONALE

Due to the text about the limits of the nice value being implementation-defined, *nice* is not actually required to change the nice value of the executed command; the limits could be zero differences from the system default, although the implementor is required to document this fact in the conformance document.

The 4.3 BSD version of *nice* does not check whether *increment* is a valid decimal integer. The command *nice -x utility*, for example, would be treated the same as the command *nice --1 utility*. If the user does not have appropriate privileges, this results in a "permission denied" error. This is considered a bug.

When a user without appropriate privileges gives a negative *increment*, System V treats it like the command *nice -0 utility*, while 4.3 BSD writes a "permission denied" message and does not run the utility. Neither was considered clearly superior, so the behavior was left unspecified.

The C shell has a built-in version of *nice* that has a different interface from the one described in this volume of IEEE Std 1003.1-2001.

The term "utility" is used, rather than "command", to highlight the fact that shell compound commands, pipelines, and so on, cannot be used. Special built-ins also cannot be used. However, "utility" includes user application programs and shell scripts, not just utilities defined in this volume of IEEE Std 1003.1-2001.

Historical implementations of *nice* provide a nice value range of 40 or 41 discrete steps, with the default nice value being the midpoint of that range. By default, they lower the nice value of the executed utility by 10.

Some historical documentation states that the *increment* value must be within a fixed range. This is misleading; the valid *increment* values on any invocation are determined by the current process nice value, which is not always the default.

The definition of nice value is not intended to suggest that all processes in a system have priorities that are comparable. Scheduling policy extensions such as the realtime priorities in the System Interfaces volume of IEEE Std 1003.1-2001 make the notion of a single underlying priority for all scheduling policies problematic. Some implementations may implement the *nice*-related features to affect all processes on the system, others to affect just the general time-sharing activities implied by this volume of IEEE Std 1003.1-2001, and others may have no effect at all. Because of the use of "implementation-defined" in *nice* and *renice*, a wide range of implementation strategies are possible.

FUTURE DIRECTIONS

None.

SEE ALSO

Shell Command Language, *renice*, the System Interfaces volume of IEEE Std 1003.1-2001, *nice()*

COPYRIGHT

Portions of this text are reprinted and reproduced in electronic form from IEEE Std 1003.1, 2003 Edition, Standard for Information Technology -- Portable Operating System Interface (POSIX), The Open Group Base Specifications Issue 6, Copyright (C) 2001-2003 by the Institute of Electrical and Electronics Engineers, Inc and The Open Group. In the event of any discrepancy between this version and the original IEEE and The Open Group Standard, the original IEEE and The Open Group Standard is the referee document. The original Standard can be obtained online at <http://www.open-group.org/unix/online.html>.

NAME

nl - line numbering filter

SYNOPSIS

nl [-p][-b *type*][-d *delim*][-f *type*][-h *type*][-i *incr*][-l *num*][-n *format*]
 [-s *sep*][-v *startnum*][-w *width*][*file*]

DESCRIPTION

The *nl* utility shall read lines from the named *file* or the standard input if no *file* is named and shall reproduce the lines to standard output. Lines shall be numbered on the left. Additional functionality may be provided in accordance with the command options in effect.

The *nl* utility views the text it reads in terms of logical pages. Line numbering shall be reset at the start of each logical page. A logical page consists of a header, a body, and a footer section. Empty sections are valid. Different line numbering options are independently available for header, body, and footer (for example, no numbering of header and footer lines while numbering blank lines only in the body).

The starts of logical page sections shall be signaled by input lines containing nothing but the following delimiter characters:

Line	Start of
\:.\:	Header
\:.	Body
\:	Footer

Unless otherwise specified, *nl* shall assume the text being read is in a single logical page body.

OPTIONS

The *nl* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section 12.2, Utility Syntax Guidelines. Only one file can be named.

The following options shall be supported:

- b** *type*
Specify which logical page body lines shall be numbered. Recognized *types* and their meaning are:
 - a** Number all lines.
 - t** Number only non-empty lines.
 - n** No line numbering.
- pstring** Number only lines that contain the basic regular expression specified in *string*.

The default *type* for logical page body shall be **t** (text lines numbered).

- d** *delim*
Specify the delimiter characters that indicate the start of a logical page section. These can be changed from the default characters "\:." to two user-specified characters. If only one character is entered, the second character shall remain the default character ':\:'.
 - f** *type* Specify the same as **b** *type* except for footer. The default for logical page footer shall be **n** (no lines numbered).
 - h** *type* Specify the same as **b** *type* except for header. The default *type* for logical page header shall be **n** (no lines numbered).
 - i** *incr* Specify the increment value used to number logical page lines. The default shall be 1.
 - l** *num* Specify the number of blank lines to be considered as one. For example, **-l 2** results in only the second adjacent blank line being numbered (if the appropriate **-h a**, **-b a**, or **-f a** option is set). The default shall be 1.
 - n** *format*
Specify the line numbering format. Recognized values are: **ln**, left justified, leading zeros suppressed; **rn**, right justified, leading zeros suppressed; **rz**, right justified, leading zeros kept. The default *format* shall be **rn** (right justified).

- p** Specify that numbering should not be restarted at logical page delimiters.
- s *sep*** Specify the characters used in separating the line number and the corresponding text line. The default *sep* shall be a <tab>.
- v *startnum***
Specify the initial value used to number logical page lines. The default shall be 1.
- w *width***
Specify the number of characters to be used for the line number. The default *width* shall be 6.

OPERANDS

The following operand shall be supported:

file A pathname of a text file to be line-numbered.

STDIN

The standard input is a text file that is used if no *file* operand is given.

INPUT FILES

The input file named by the *file* operand is a text file.

ENVIRONMENT VARIABLES

The following environment variables shall affect the execution of *nl*:

LANG Provide a default value for the internationalization variables that are unset or null. (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2, Internationalization Variables for the precedence of internationalization variables used to determine the values of locale categories.)

LC_ALL

If set to a non-empty string value, override the values of all the other internationalization variables.

LC_COLLATE

Determine the locale for the behavior of ranges, equivalence classes, and multi-character collating elements within regular expressions.

LC_CTYPE

Determine the locale for the interpretation of sequences of bytes of text data as characters (for example, single-byte as opposed to multi-byte characters in arguments and input files), the behavior of character classes within regular expressions, and for deciding which characters are in character class **graph** (for the **-b t**, **-f t**, and **-h t** options).

LC_MESSAGES

Determine the locale that should be used to affect the format and contents of diagnostic messages written to standard error.

NLSPATH

Determine the location of message catalogs for the processing of **LC_MESSAGES**.

ASYNCHRONOUS EVENTS

Default.

STDOUT

The standard output shall be a text file in the following format:

```
"%s%s%s", <line number>, <separator>, <input line>
```

where <line number> is one of the following numeric formats:

%6d When the **rn** format is used (the default; see **-n**).

%06d When the **rz** format is used.

%-6d When the **ln** format is used.

<empty>

When line numbers are suppressed for a portion of the page; the *<separator>* is also suppressed.

In the preceding list, the number 6 is the default width; the **-w** option can change this value.

STDERR

The standard error shall be used only for diagnostic messages.

OUTPUT FILES

None.

EXTENDED DESCRIPTION

None.

EXIT STATUS

The following exit values shall be returned:

- 0 Successful completion.
- >0 An error occurred.

CONSEQUENCES OF ERRORS

Default.

The following sections are informative.

APPLICATION USAGE

In using the **-d delim** option, care should be taken to escape characters that have special meaning to the command interpreter.

EXAMPLES

The command:

nl -v 10 -i 10 -d \!+ file1

numbers *file1* starting at line number 10 with an increment of 10. The logical page delimiter is **"!+"**. Note that the **"!"** has to be escaped when using *csh* as a command interpreter because of its history substitution syntax. For *ksh* and *sh* the escape is not necessary, but does not do any harm.

RATIONALE

None.

FUTURE DIRECTIONS

None.

SEE ALSO

pr

COPYRIGHT

Portions of this text are reprinted and reproduced in electronic form from IEEE Std 1003.1, 2003 Edition, Standard for Information Technology -- Portable Operating System Interface (POSIX), The Open Group Base Specifications Issue 6, Copyright (C) 2001-2003 by the Institute of Electrical and Electronics Engineers, Inc and The Open Group. In the event of any discrepancy between this version and the original IEEE and The Open Group Standard, the original IEEE and The Open Group Standard is the referee document. The original Standard can be obtained online at <http://www.open-group.org/unix/online.html>.

NAME

nohup - invoke a utility immune to hangups

SYNOPSIS

nohup *utility* [*argument...*]

DESCRIPTION

The *nohup* utility shall invoke the utility named by the *utility* operand with arguments supplied as the *argument* operands. At the time the named *utility* is invoked, the SIGHUP signal shall be set to be ignored.

If the standard output is a terminal, all output written by the named *utility* to its standard output shall be appended to the end of the file **nohup.out** in the current directory. If **nohup.out** cannot be created or opened for appending, the output shall be appended to the end of the file **nohup.out** in the directory specified by the *HOME* environment variable. If neither file can be created or opened for appending, *utility* shall not be invoked. If a file is created, the file's permission bits shall be set to S_IRUSR | S_IWUSR.

If the standard error is a terminal, all output written by the named *utility* to its standard error shall be redirected to the same file descriptor as the standard output.

OPTIONS

None.

OPERANDS

The following operands shall be supported:

utility The name of a utility that is to be invoked. If the *utility* operand names any of the special built-in utilities in *Special Built-In Utilities*, the results are undefined.

argument Any string to be supplied as an argument when invoking the utility named by the *utility* operand.

STDIN

Not used.

INPUT FILES

None.

ENVIRONMENT VARIABLES

The following environment variables shall affect the execution of *nohup*:

HOME Determine the pathname of the user's home directory: if the output file **nohup.out** cannot be created in the current directory, the *nohup* utility shall use the directory named by *HOME* to create the file.

LANG Provide a default value for the internationalization variables that are unset or null. (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2, Internationalization Variables for the precedence of internationalization variables used to determine the values of locale categories.)

LC_ALL

If set to a non-empty string value, override the values of all the other internationalization variables.

LC_CTYPE

Determine the locale for the interpretation of sequences of bytes of text data as characters (for example, single-byte as opposed to multi-byte characters in arguments).

LC_MESSAGES

Determine the locale that should be used to affect the format and contents of diagnostic messages written to standard error.

NLSPATH

Determine the location of message catalogs for the processing of *LC_MESSAGES*.

PATH Determine the search path that is used to locate the utility to be invoked. See the Base Definitions volume of IEEE Std 1003.1-2001, Chapter 8, Environment Variables.

ASYNCHRONOUS EVENTS

The *nohup* utility shall take the standard action for all signals except that SIGHUP shall be ignored.

STDOUT

If the standard output is not a terminal, the standard output of *nohup* shall be the standard output generated by the execution of the *utility* specified by the operands. Otherwise, nothing shall be written to the standard output.

STDERR

If the standard output is a terminal, a message shall be written to the standard error, indicating the name of the file to which the output is being appended. The name of the file shall be either **nohup.out** or **\$HOME/nohup.out**.

OUTPUT FILES

If the standard output is a terminal, all output written by the named *utility* to the standard output and standard error is appended to the file **nohup.out**, which is created if it does not already exist.

EXTENDED DESCRIPTION

None.

EXIT STATUS

The following exit values shall be returned:

- 126 The utility specified by *utility* was found but could not be invoked.
- 127 An error occurred in the *nohup* utility or the utility specified by *utility* could not be found.

Otherwise, the exit status of *nohup* shall be that of the utility specified by the *utility* operand.

CONSEQUENCES OF ERRORS

Default.

The following sections are informative.

APPLICATION USAGE

The *command*, *env*, *nice*, *nohup*, *time*, and *xargs* utilities have been specified to use exit code 127 if an error occurs so that applications can distinguish "failure to find a utility" from "invoked utility exited with an error indication". The value 127 was chosen because it is not commonly used for other meanings; most utilities use small values for "normal error conditions" and the values above 128 can be confused with termination due to receipt of a signal. The value 126 was chosen in a similar manner to indicate that the utility could be found, but not invoked. Some scripts produce meaningful error messages differentiating the 126 and 127 cases. The distinction between exit codes 126 and 127 is based on KornShell practice that uses 127 when all attempts to *exec* the utility fail with [ENOENT], and uses 126 when any attempt to *exec* the utility fails for any other reason.

EXAMPLES

It is frequently desirable to apply *nohup* to pipelines or lists of commands. This can be done by placing pipelines and command lists in a single file; this file can then be invoked as a utility, and the *nohup* applies to everything in the file.

Alternatively, the following command can be used to apply *nohup* to a complex command:

```
nohup sh -c 'complex-command-line'
```

RATIONALE

The 4.3 BSD version ignores SIGTERM and SIGHUP, and if **./nohup.out** cannot be used, it fails instead of trying to use **\$HOME/nohup.out**.

The *cs* utility has a built-in version of *nohup* that acts differently from the *nohup* defined in this volume of IEEE Std 1003.1-2001.

The term *utility* is used, rather than *command*, to highlight the fact that shell compound commands, pipelines, special built-ins, and so on, cannot be used directly. However, *utility* includes user application programs and shell scripts, not just the standard utilities.

Historical versions of the *nohup* utility use default file creation semantics. Some more recent versions use the permissions specified here as an added security precaution.

Some historical implementations ignore SIGQUIT in addition to SIGHUP; others ignore SIGTERM. An early proposal allowed, but did not require, SIGQUIT to be ignored. Several reviewers objected that *nohup* should only modify the handling of SIGHUP as required by this volume of IEEE Std 1003.1-2001.

FUTURE DIRECTIONS

None.

SEE ALSO

Shell Command Language , *sh* , the System Interfaces volume of IEEE Std 1003.1-2001, *signal()*

COPYRIGHT

Portions of this text are reprinted and reproduced in electronic form from IEEE Std 1003.1, 2003 Edition, Standard for Information Technology -- Portable Operating System Interface (POSIX), The Open Group Base Specifications Issue 6, Copyright (C) 2001-2003 by the Institute of Electrical and Electronics Engineers, Inc and The Open Group. In the event of any discrepancy between this version and the original IEEE and The Open Group Standard, the original IEEE and The Open Group Standard is the referee document. The original Standard can be obtained online at <http://www.open-group.org/unix/online.html> .

NAME

od - dump files in various formats

SYNOPSIS

```
od [-v][-A address_base][-j skip][-N count][-t type_string]...
    [file...]
```

```
od [-bcdosx][file] [[+]offset[.][b]]
```

DESCRIPTION

The *od* utility shall write the contents of its input files to standard output in a user-specified format.

OPTIONS

The *od* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section 12.2, Utility Syntax Guidelines, except that the order of presentation of the **-t** options and the **-bcdosx** options is significant.

The following options shall be supported:

-A *address_base*

Specify the input offset base. See the EXTENDED DESCRIPTION section. The application shall ensure that the *address_base* option-argument is a character. The characters **'d'**, **'o'**, and **'x'** specify that the offset base shall be written in decimal, octal, or hexadecimal, respectively. The character **'n'** specifies that the offset shall not be written.

-b Interpret bytes in octal. This shall be equivalent to **-t o1**.

-c Interpret bytes as characters specified by the current setting of the *LC_CTYPE* category. Certain non-graphic characters appear as C escapes: **"NUL=\0"**, **"BS=\b"**, **"FF=\f"**, **"NL=\n"**, **"CR=\r"**, **"HT=\t"**; others appear as 3-digit octal numbers.

-d Interpret *words* (two-byte units) in unsigned decimal. This shall be equivalent to **-t u2**.

-j *skip* Jump over *skip* bytes from the beginning of the input. The *od* utility shall read or seek past the first *skip* bytes in the concatenated input files. If the combined input is not at least *skip* bytes long, the *od* utility shall write a diagnostic message to standard error and exit with a non-zero exit status.

By default, the *skip* option-argument shall be interpreted as a decimal number. With a leading 0x or 0X, the offset shall be interpreted as a hexadecimal number; otherwise, with a leading **'0'**, the offset shall be interpreted as an octal number. Appending the character **'b'**, **'k'**, or **'m'** to offset shall cause it to be interpreted as a multiple of 512, 1024, or 1048576 bytes, respectively. If the *skip* number is hexadecimal, any appended **'b'** shall be considered to be the final hexadecimal digit.

-N *count*

Format no more than *count* bytes of input. By default, *count* shall be interpreted as a decimal number. With a leading 0x or 0X, *count* shall be interpreted as a hexadecimal number; otherwise, with a leading **'0'**, it shall be interpreted as an octal number. If *count* bytes of input (after successfully skipping, if **-j** *skip* is specified) are not available, it shall not be considered an error; the *od* utility shall format the input that is available.

-o Interpret *words* (two-byte units) in octal. This shall be equivalent to **-t o2**.

-s Interpret *words* (two-byte units) in signed decimal. This shall be equivalent to **-t d2**.

-t *type_string*

Specify one or more output types. See the EXTENDED DESCRIPTION section. The application shall ensure that the *type_string* option-argument is a string specifying the types to be used when writing the input data. The string shall consist of the type specification characters **a**, **c**, **d**, **f**, **o**, **u**, and **x**, specifying named character, character, signed decimal, floating point, octal, unsigned decimal, and hexadecimal, respectively. The type specification characters **d**, **f**

, **o**, **u**, and **x** can be followed by an optional unsigned decimal integer that specifies the number of bytes to be transformed by each instance of the output type. The type specification character **f** can be followed by an optional **F**, **D**, or **L** indicating that the conversion should be applied to an item of type **float**, **double**, or **long double**, respectively. The type specification characters **d**, **o**, **u**, and **x** can be followed by an optional **C**, **S**, **I**, or **L** indicating that the conversion should be applied to an item of type **char**, **short**, **int**, or **long**, respectively. Multiple types can be concatenated within the same *type_string* and multiple **-t** options can be specified. Output lines shall be written for each type specified in the order in which the type specification characters are specified.

- v** Write all input data. Without the **-v** option, any number of groups of output lines, which would be identical to the immediately preceding group of output lines (except for the byte offsets), shall be replaced with a line containing only an asterisk (**'*'**).
- x** Interpret *words* (two-byte units) in hexadecimal. This shall be equivalent to **-t x2**.

Multiple types can be specified by using multiple **-bcdostx** options. Output lines are written for each type specified in the order in which the types are specified.

OPERANDS

The following operands shall be supported:

file A pathname of a file to be read. If no *file* operands are specified, the standard input shall be used.

If there are no more than two operands, none of the **-A**, **-j**, **-N**, or **-t** options is specified, and either of the following is true: the first character of the last operand is a plus sign (**'+'**), or there are two operands and the first character of the last operand is numeric; the last operand shall be interpreted as an offset operand on XSI-conformant systems. Under these conditions, the results are unspecified on systems that are not XSI-conformant systems.

[+]*offset***[.]****[b]**

The *offset* operand specifies the offset in the file where dumping is to commence. This operand is normally interpreted as octal bytes. If **'.'** is appended, the offset shall be interpreted in decimal. If **'b'** is appended, the offset shall be interpreted in units of 512 bytes.

STDIN

The standard input shall be used only if no *file* operands are specified. See the INPUT FILES section.

INPUT FILES

The input files can be any file type.

ENVIRONMENT VARIABLES

The following environment variables shall affect the execution of *od*:

LANG Provide a default value for the internationalization variables that are unset or null. (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2, Internationalization Variables for the precedence of internationalization variables used to determine the values of locale categories.)

LC_ALL

If set to a non-empty string value, override the values of all the other internationalization variables.

LC_CTYPE

Determine the locale for the interpretation of sequences of bytes of text data as characters (for example, single-byte as opposed to multi-byte characters in arguments and input files).

LC_MESSAGES

Determine the locale that should be used to affect the format and contents of diagnostic messages written to standard error.

LC_NUMERIC

Determine the locale for selecting the radix character used when writing floating-point formatted output.

NLSPATH

Determine the location of message catalogs for the processing of *LC_MESSAGES*.

ASYNCHRONOUS EVENTS

Default.

STDOUT

See the EXTENDED DESCRIPTION section.

STDERR

The standard error shall be used only for diagnostic messages.

OUTPUT FILES

None.

EXTENDED DESCRIPTION

The *od* utility shall copy sequentially each input file to standard output, transforming the input data according to the output types specified by the **-t** option or the **-bcdox** options. If no output type is specified, the default output shall be as if **-t oS** had been specified.

The number of bytes transformed by the output type specifier **c** may be variable depending on the *LC_CTYPE* category.

The default number of bytes transformed by output type specifiers **d**, **f**, **o**, **u**, and **x** corresponds to the various C-language types as follows. If the *c99* compiler is present on the system, these specifiers shall correspond to the sizes used by default in that compiler. Otherwise, these sizes may vary among systems that conform to IEEE Std 1003.1-2001.

For the type specifier characters **d**, **o**, **u**, and **x**, the default number of bytes shall correspond to the size of the underlying implementation's basic integer type. For these specifier characters, the implementation shall support values of the optional number of bytes to be converted corresponding to the number of bytes in the C-language types **char**, **short**, **int**, and **long**. These numbers can also be specified by an application as the characters **'C'**, **'S'**, **'I'**, and **'L'**, respectively. The implementation shall also support the values 1, 2, 4, and 8, even if it provides no C-Language types of those sizes. The implementation shall support the decimal value corresponding to the C-language type **long long**. The byte order used when interpreting numeric values is implementation-defined, but shall correspond to the order in which a constant of the corresponding type is stored in memory on the system.

For the type specifier character **f**, the default number of bytes shall correspond to the number of bytes in the underlying implementation's basic double precision floating-point data type. The implementation shall support values of the optional number of bytes to be converted corresponding to the number of bytes in the C-language types **float**, **double**, and **long double**. These numbers can also be specified by an application as the characters **'F'**, **'D'**, and **'L'**, respectively.

The type specifier character **a** specifies that bytes shall be interpreted as named characters from the International Reference Version (IRV) of the ISO/IEC 646:1991 standard. Only the least significant seven bits of each byte shall be used for this type specification. Bytes with the values listed in the following table shall be written using the corresponding names for those characters.

Table: Named Characters in *od*

Value	Name	Value	Name	Value	Name	Value	Name
\000	nul	\001	soh	\002	stx	\003	etx
\004	eot	\005	enq	\006	ack	\007	bel
\010	bs	\011	ht	\012	lf or nl	\013	vt
\014	ff	\015	cr	\016	so	\017	si
\020	dle	\021	dc1	\022	dc2	\023	dc3
\024	dc4	\025	nak	\026	syn	\027	etb
\030	can	\031	em	\032	sub	\033	esc
\034	fs	\035	gs	\036	rs	\037	us
\040	sp	\177	del				

Note: The "\012" value may be written either as **lf** or **nl**.

The type specifier character **c** specifies that bytes shall be interpreted as characters specified by the current setting of the *LC_CTYPE* locale category. Characters listed in the table in the Base Definitions volume of IEEE Std 1003.1-2001, Chapter 5, File Format Notation (`'\'`, `'\a'`, `'\b'`, `'\f'`, `'\n'`, `'\r'`, `'\t'`, `'\v'`) shall be written as the corresponding escape sequences, except that backslash shall be written as a single backslash and a NUL shall be written as `'\0'`. Other non-printable characters shall be written as one three-digit octal number for each byte in the character. If the size of a byte on the system is greater than nine bits, the format used for non-printable characters is implementation-defined. Printable multi-byte characters shall be written in the area corresponding to the first byte of the character; the two-character sequence `"**"` shall be written in the area corresponding to each remaining byte in the character, as an indication that the character is continued. When either the `-j skip` or `-N count` option is specified along with the **c** type specifier, and this results in an attempt to start or finish in the middle of a multi-byte character, the result is implementation-defined.

The input data shall be manipulated in blocks, where a block is defined as a multiple of the least common multiple of the number of bytes transformed by the specified output types. If the least common multiple is greater than 16, the results are unspecified. Each input block shall be written as transformed by each output type, one per written line, in the order that the output types were specified. If the input block size is larger than the number of bytes transformed by the output type, the output type shall sequentially transform the parts of the input block, and the output from each of the transformations shall be separated by one or more `<blank>`s.

If, as a result of the specification of the `-N` option or end-of-file being reached on the last input file, input data only partially satisfies an output type, the input shall be extended sufficiently with null bytes to write the last byte of the input.

Unless `-A n` is specified, the first output line produced for each input block shall be preceded by the input offset, cumulative across input files, of the next byte to be written. The format of the input offset is unspecified; however, it shall not contain any `<blank>`s, shall start at the first character of the output line, and shall be followed by one or more `<blank>`s. In addition, the offset of the byte following the last byte written shall be written after all the input data has been processed, but shall not be followed by any `<blank>`s.

If no `-A` option is specified, the input offset base is unspecified.

EXIT STATUS

The following exit values shall be returned:

- 0 All input files were processed successfully.
- >0 An error occurred.

CONSEQUENCES OF ERRORS

Default.

The following sections are informative.

APPLICATION USAGE

XSI-conformant applications are warned not to use filenames starting with `'+'` or a first operand starting with a numeric character so that the old functionality can be maintained by implementations, unless they specify one of the `-A`, `-j`, or `-N` options. To guarantee that one of these filenames is always interpreted as a filename, an application could always specify the address base format with the `-A` option.

EXAMPLES

If a file containing 128 bytes with decimal values zero to 127, in increasing order, is supplied as standard input to the command:

```
od -A d -t a
```

on an implementation using an input block size of 16 bytes, the standard output, independent of the current locale setting, would be similar to:

```
0000000 nul soh stx etx eot enq ack bel bs ht nl vt ff cr so si
0000016 dle dc1 dc2 dc3 dc4 nak syn etb can em sub esc fs gs rs us
```

```

0000032 sp ! " # $ % & ' ( ) * + , - . /
0000048 0 1 2 3 4 5 6 7 8 9 : ; < = > ?
0000064 @ A B C D E F G H I J K L M N O
0000080 P Q R S T U V W X Y Z [ \ ] ^ _
0000096 ` a b c d e f g h i j k l m n o
0000112 p q r s t u v w x y z { | } ~ del
0000128

```

Note that this volume of IEEE Std 1003.1-2001 allows **nl** or **lf** to be used as the name for the ISO/IEC 646:1991 standard IRV character with decimal value 10. The IRV names this character **lf** (line feed), but traditional implementations have referred to this character as newline (**nl**) and the POSIX locale character set symbolic name for the corresponding character is a <newline>.

The command:

```
od -A o -t o2x2x -N 18
```

on a system with 32-bit words and an implementation using an input block size of 16 bytes could write 18 bytes in approximately the following format:

```

0000000 032056 031440 041123 042040 052516 044530 020043 031464
      342e 3320 4253 4420 554e 4958 2023 3334
      342e3320 42534420 554e4958 20233334
0000020 032472
      353a
      353a0000
0000022

```

The command:

```
od -A d -t f -t o4 -t x4 -N 24 -j 0x15
```

on a system with 64-bit doubles (for example, IEEE Std 754-1985 double precision floating-point format) would skip 21 bytes of input data and then write 24 bytes in approximately the following format:

```

0000000 1.00000000000000e+00 1.57350000000000e+01
      0777400000 0000000000 10013674121 35341217270
      3ff00000 00000000 402f3851 eb851eb8
0000016 1.40668230000000e+02
      10030312542 04370303230
      40619562 23e18698
0000024

```

RATIONALE

The *od* utility went through several names in early proposals, including *hd*, *xd*, and most recently *hexdump*. There were several objections to all of these based on the following reasons:

The *hd* and *xd* names conflicted with historical utilities that behaved differently.

The *hexdump* description was much more complex than needed for a simple dump utility.

The *od* utility has been available on all historical implementations and there was no need to create a new name for a utility so similar to the historical *od* utility.

The original reasons for not standardizing historical *od* were also fairly widespread. Those reasons are given below along with rationale explaining why the standard developers believe that this version does not suffer from the indicated problem:

The BSD and System V versions of *od* have diverged, and the intersection of features provided by both does not meet the needs of the user community. In fact, the System V version only provides a mechanism for dumping octal bytes and **shorts**, signed and unsigned decimal

shorts, hexadecimal **shorts**, and ASCII characters. BSD added the ability to dump **floats**, **doubles**, named ASCII characters, and octal, signed decimal, unsigned decimal, and hexadecimal **longs**. The version presented here provides more normalized forms for dumping bytes, **shorts**, **ints**, and **longs** in octal, signed decimal, unsigned decimal, and hexadecimal; **float**, **double**, and **long double**; and named ASCII as well as current locale characters.

It would not be possible to come up with a compatible superset of the BSD and System V flags that met the requirements of the standard developers. The historical default *od* output is the specified default output of this utility. None of the option letters chosen for this version of *od* conflict with any of the options to historical versions of *od*.

On systems with different sizes for **short**, **int**, and **long**, there was no way to ask for dumps of **ints**, even in the BSD version. Because of the way options are named, the name space could not be extended to solve these problems. This is why the **-t** option was added (with type specifiers more closely matched to the *printf()* formats used in the rest of this volume of IEEE Std 1003.1-2001) and the optional field sizes were added to the **d**, **f**, **o**, **u**, and **x** type specifiers. It is also one of the reasons why the historical practice was not mandated as a required obsolescent form of *od*. (Although the old versions of *od* are not listed as an obsolescent form, implementations are urged to continue to recognize the older forms for several more years.) The **a**, **c**, **f**, **o**, and **x** types match the meaning of the corresponding format characters in the historical implementations of *od* except for the default sizes of the fields converted. The **d** format is signed in this volume of IEEE Std 1003.1-2001 to match the *printf()* notation. (Historical versions of *od* used **d** as a synonym for **u** in this version. The System V implementation uses **s** for signed decimal; BSD uses **i** for signed decimal and **s** for null-terminated strings.) Other than **d** and **u**, all of the type specifiers match format characters in the historical BSD version of *od*.

The sizes of the C-language types **char**, **short**, **int**, **long**, **float**, **double**, and **long double** are used even though it is recognized that there may be zero or more than one compiler for the C language on an implementation and that they may use different sizes for some of these types. (For example, one compiler might use 2 bytes **shorts**, 2 bytes **ints**, and 4 bytes **longs**, while another compiler (or an option to the same compiler) uses 2 bytes **shorts**, 4 bytes **ints**, and 4 bytes **longs**.) Nonetheless, there has to be a basic size known by the implementation for these types, corresponding to the values reported by invocations of the *getconf* utility when called with *system_var* operands {UCHAR_MAX}, {USHORT_MAX}, {UINT_MAX}, and {ULONG_MAX} for the types **char**, **short**, **int**, and **long**, respectively. There are similar constants required by the ISO C standard, but not required by the System Interfaces volume of IEEE Std 1003.1-2001 or this volume of IEEE Std 1003.1-2001. They are {FLT_MANT_DIG}, {DBL_MANT_DIG}, and {LDBL_MANT_DIG} for the types **float**, **double**, and **long double**, respectively. If the optional *c99* utility is provided by the implementation and used as specified by this volume of IEEE Std 1003.1-2001, these are the sizes that would be provided. If an option is used that specifies different sizes for these types, there is no guarantee that the *od* utility is able to interpret binary data output by such a program correctly.

This volume of IEEE Std 1003.1-2001 requires that the numeric values of these lengths be recognized by the *od* utility and that symbolic forms also be recognized. Thus, a conforming application can always look at an array of **unsigned long** data elements using *od -t uL*.

The method of specifying the format for the address field based on specifying a starting offset in a file unnecessarily tied the two together. The **-A** option now specifies the address base and the **-S** option specifies a starting offset.

It would be difficult to break the dependence on U.S. ASCII to achieve an internationalized utility. It does not seem to be any harder for *od* to dump characters in the current locale than it is for the *ed* or *sed* **I** commands. The **c** type specifier does this without difficulty and is completely compatible with the historical implementations of the **c** format character when the current locale uses a superset of the ISO/IEC 646:1991 standard as a codeset. The **a** type specifier (from the BSD **a** format character) was left as a portable means to dump ASCII (or more correctly ISO/IEC 646:1991 standard (IRV)) so that headers produced by *pax* could be deciphered even on systems that do not use the ISO/IEC 646:1991 standard as a subset of their base codeset.

The use of "***" as an indication of continuation of a multi-byte character in **c** specifier output was

chosen based on seeing an implementation that uses this method. The continuation bytes have to be marked in a way that is not ambiguous with another single-byte or multi-byte character.

An early proposal used **-S** and **-n**, respectively, for the **-j** and **-N** options eventually selected. These were changed to avoid conflicts with historical implementations.

The original standard specified **-t o2** as the default when no output type was given. This was changed to **-t oS** (the length of a **short**) to accommodate a supercomputer implementation that historically used 64 bits as its default (and that defined shorts as 64 bits). This change should not affect conforming applications. The requirement to support lengths of 1, 2, and 4 was added at the same time to address an historical implementation that had no two-byte data types in its C compiler.

The use of a basic integer data type is intended to allow the implementation to choose a word size commonly used by applications on that architecture.

FUTURE DIRECTIONS

All option and operand interfaces marked as extensions may be withdrawn in a future version.

SEE ALSO

c99, *sed*

COPYRIGHT

Portions of this text are reprinted and reproduced in electronic form from IEEE Std 1003.1, 2003 Edition, Standard for Information Technology -- Portable Operating System Interface (POSIX), The Open Group Base Specifications Issue 6, Copyright (C) 2001-2003 by the Institute of Electrical and Electronics Engineers, Inc and The Open Group. In the event of any discrepancy between this version and the original IEEE and The Open Group Standard, the original IEEE and The Open Group Standard is the referee document. The original Standard can be obtained online at <http://www.open-group.org/unix/online.html>.

NAME

paste - merge corresponding or subsequent lines of files

SYNOPSIS

paste [-s][-d *list*] *file*...

DESCRIPTION

The *paste* utility shall concatenate the corresponding lines of the given input files, and write the resulting lines to standard output.

The default operation of *paste* shall concatenate the corresponding lines of the input files. The <newline> of every line except the line from the last input file shall be replaced with a <tab>.

If an end-of-file condition is detected on one or more input files, but not all input files, *paste* shall behave as though empty lines were read from the files on which end-of-file was detected, unless the -s option is specified.

OPTIONS

The *paste* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section 12.2, Utility Syntax Guidelines.

The following options shall be supported:

-d *list* Unless a backslash character appears in *list*, each character in *list* is an element specifying a delimiter character. If a backslash character appears in *list*, the backslash character and one or more characters following it are an element specifying a delimiter character as described below. These elements specify one or more delimiters to use, instead of the default <tab>, to replace the <newline> of the input lines. The elements in *list* shall be used circularly; that is, when the list is exhausted the first element from the list is reused. When the -s option is specified:

The last <newline> in a file shall not be modified.

The delimiter shall be reset to the first element of *list* after each *file* operand is processed.

When the -s option is not specified:

The <newline>s in the file specified by the last *file* operand shall not be modified.

The delimiter shall be reset to the first element of *list* each time a line is processed from each file.

If a backslash character appears in *list*, it and the character following it shall be used to represent the following delimiter characters:

\n <newline>.

\t <tab>.

**** Backslash character.

\0 Empty string (not a null character). If '\0' is immediately followed by the character 'x', the character 'X', or any character defined by the *LC_CTYPE digit* keyword (see the Base Definitions volume of IEEE Std 1003.1-2001, Chapter 7, Locale), the results are unspecified.

If any other characters follow the backslash, the results are unspecified.

-s Concatenate all of the lines of each separate input file in command line order. The <newline> of every line except the last line in each input file shall be replaced with the <tab>, unless otherwise specified by the -d option.

OPERANDS

The following operand shall be supported:

file A pathname of an input file. If '-' is specified for one or more of the *files*, the standard input shall be used; the standard input shall be read one line at a time, circularly, for each instance of '-'. Implementations shall support pasting of at least 12 *file* operands.

STDIN

The standard input shall be used only if one or more *file* operands is *'-'*. See the INPUT FILES section.

INPUT FILES

The input files shall be text files, except that line lengths shall be unlimited.

ENVIRONMENT VARIABLES

The following environment variables shall affect the execution of *paste*:

LANG Provide a default value for the internationalization variables that are unset or null. (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2, Internationalization Variables for the precedence of internationalization variables used to determine the values of locale categories.)

LC_ALL

If set to a non-empty string value, override the values of all the other internationalization variables.

LC_CTYPE

Determine the locale for the interpretation of sequences of bytes of text data as characters (for example, single-byte as opposed to multi-byte characters in arguments and input files).

LC_MESSAGES

Determine the locale that should be used to affect the format and contents of diagnostic messages written to standard error.

NLSPATH

Determine the location of message catalogs for the processing of **LC_MESSAGES**.

ASYNCHRONOUS EVENTS

Default.

STDOUT

Concatenated lines of input files shall be separated by the <tab> (or other characters under the control of the **-d** option) and terminated by a <newline>.

STDERR

The standard error shall be used only for diagnostic messages.

OUTPUT FILES

None.

EXTENDED DESCRIPTION

None.

EXIT STATUS

The following exit values shall be returned:

- 0 Successful completion.
- >0 An error occurred.

CONSEQUENCES OF ERRORS

If one or more input files cannot be opened when the **-s** option is not specified, a diagnostic message shall be written to standard error, but no output is written to standard output. If the **-s** option is specified, the *paste* utility shall provide the default behavior described in *Utility Description Defaults*.

The following sections are informative.

APPLICATION USAGE

When the escape sequences of the *list* option-argument are used in a shell script, they must be quoted; otherwise, the shell treats the *'\'* as a special character.

Conforming applications should only use the specific backslash escaped delimiters presented in this volume of IEEE Std 1003.1-2001. Historical implementations treat *'\x'*, where *'x'* is not in this list, as *'x'*, but future implementations are free to expand this list to recognize other common escapes similar to those accepted by *printf* and other standard utilities.

Most of the standard utilities work on text files. The *cut* utility can be used to turn files with arbitrary

line lengths into a set of text files containing the same data. The *paste* utility can be used to create (or recreate) files with arbitrary line lengths. For example, if *file* contains long lines:

```
cut -b 1-500 -n file > file1
cut -b 501- -n file > file2
```

creates **file1** (a text file) with lines no longer than 500 bytes (plus the <newline>) and **file2** that contains the remainder of the data from *file*. Note that **file2** is not a text file if there are lines in *file* that are longer than 500 + {LINE_MAX} bytes. The original file can be recreated from **file1** and **file2** using the command:

```
paste -d "\0" file1 file2 > file
```

The commands:

```
paste -d "\0" ...
paste -d "" ...
```

are not necessarily equivalent; the latter is not specified by this volume of IEEE Std 1003.1-2001 and may result in an error. The construct '\0' is used to mean "no separator" because historical versions of *paste* did not follow the syntax guidelines, and the command:

```
paste -d "" ...
```

could not be handled properly by *getopt()*.

EXAMPLES

Write out a directory in four columns:

```
ls | paste - - - -
```

Combine pairs of lines from a file into single lines:

```
paste -s -d "\t\n" file
```

RATIONALE

None.

FUTURE DIRECTIONS

None.

SEE ALSO

Utility Description Defaults, *cut*, *grep*, *pr*

COPYRIGHT

Portions of this text are reprinted and reproduced in electronic form from IEEE Std 1003.1, 2003 Edition, Standard for Information Technology -- Portable Operating System Interface (POSIX), The Open Group Base Specifications Issue 6, Copyright (C) 2001-2003 by the Institute of Electrical and Electronics Engineers, Inc and The Open Group. In the event of any discrepancy between this version and the original IEEE and The Open Group Standard, the original IEEE and The Open Group Standard is the referee document. The original Standard can be obtained online at <http://www.open-group.org/unix/online.html>.

NAME

pathchk - check pathnames

SYNOPSIS

pathchk [-p] *pathname*...

DESCRIPTION

The *pathchk* utility shall check that one or more pathnames are valid (that is, they could be used to access or create a file without causing syntax errors) and portable (that is, no filename truncation results). More extensive portability checks are provided by the **-p** option.

By default, the *pathchk* utility shall check each component of each *pathname* operand based on the underlying file system. A diagnostic shall be written for each *pathname* operand that:

Is longer than {PATH_MAX} bytes (see **Pathname Variable Values** in the Base Definitions volume of IEEE Std 1003.1-2001, Chapter 13, Headers, <limits.h>)

Contains any component longer than {NAME_MAX} bytes in its containing directory

Contains any component in a directory that is not searchable

Contains any character in any component that is not valid in its containing directory

The format of the diagnostic message is not specified, but shall indicate the error detected and the corresponding *pathname* operand.

It shall not be considered an error if one or more components of a *pathname* operand do not exist as long as a file matching the pathname specified by the missing components could be created that does not violate any of the checks specified above.

OPTIONS

The *pathchk* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section 12.2, Utility Syntax Guidelines.

The following option shall be supported:

-p Instead of performing checks based on the underlying file system, write a diagnostic for each *pathname* operand that:

Is longer than {_POSIX_PATH_MAX} bytes (see **Minimum Values** in the Base Definitions volume of IEEE Std 1003.1-2001, Chapter 13, Headers, <limits.h>)

Contains any component longer than {_POSIX_NAME_MAX} bytes

Contains any character in any component that is not in the portable filename character set

OPERANDS

The following operand shall be supported:

pathname

A pathname to be checked.

STDIN

Not used.

INPUT FILES

None.

ENVIRONMENT VARIABLES

The following environment variables shall affect the execution of *pathchk*:

LANG Provide a default value for the internationalization variables that are unset or null. (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2, Internationalization Variables for the precedence of internationalization variables used to determine the values of locale categories.)

LC_ALL

If set to a non-empty string value, override the values of all the other internationalization variables.

LC_CTYPE

Determine the locale for the interpretation of sequences of bytes of text data as characters (for example, single-byte as opposed to multi-byte characters in arguments).

LC_MESSAGES

Determine the locale that should be used to affect the format and contents of diagnostic messages written to standard error.

NLSPATH

Determine the location of message catalogs for the processing of *LC_MESSAGES*.

ASYNCHRONOUS EVENTS

Default.

STDOUT

Not used.

STDERR

The standard error shall be used only for diagnostic messages.

OUTPUT FILES

None.

EXTENDED DESCRIPTION

None.

EXIT STATUS

The following exit values shall be returned:

- 0 All *pathname* operands passed all of the checks.
- >0 An error occurred.

CONSEQUENCES OF ERRORS

Default.

The following sections are informative.

APPLICATION USAGE

The *test* utility can be used to determine whether a given pathname names an existing file; it does not, however, give any indication of whether or not any component of the pathname was truncated in a directory where the `_POSIX_NO_TRUNC` feature is not in effect. The *pathchk* utility does not check for file existence; it performs checks to determine whether a pathname does exist or could be created with no pathname component truncation.

The *noclobber* option in the shell (see the *set* special built-in) can be used to atomically create a file. As with all file creation semantics in the System Interfaces volume of IEEE Std 1003.1-2001, it guarantees atomic creation, but still depends on applications to agree on conventions and cooperate on the use of files after they have been created.

EXAMPLES

To verify that all pathnames in an imported data interchange archive are legitimate and unambiguous on the current system:

```
pax -f archive | sed -e ' /= .*s///' | xargs pathchk
if [ $? -eq 0 ]
then
    pax -r -f archive
else
    echo Investigate problems before importing files.
    exit 1
```

fi

To verify that all files in the current directory hierarchy could be moved to any system conforming to the System Interfaces volume of IEEE Std 1003.1-2001 that also supports the *pax* utility:

```
find . -print | xargs pathchk -p
if [ $? -eq 0 ]
then
    pax -w -f archive .
else
    echo Portable archive cannot be created.
    exit 1
fi
```

To verify that a user-supplied pathname names a readable file and that the application can create a file extending the given path without truncation and without overwriting any existing file:

```
case $- in
    *C*)  reset="";;
    *)    reset="set +C"
          set -C;;
esac
test -r "$path" && pathchk "$path.out" &&
rm "$path.out" > "$path.out"
if [ $? -ne 0 ]; then
    printf "%s: %s not found or %s.out fails \
creation checks.\n" $0 "$path" "$path"
    $reset # Reset the noclobber option in case a trap
           # on EXIT depends on it.
    exit 1
fi
$reset
PROCESSING < "$path" > "$path.out"
```

The following assumptions are made in this example:

PROCESSING represents the code that is used by the application to use **\$path** once it is verified that **\$path.out** works as intended.

The state of the *noclobber* option is unknown when this code is invoked and should be set on exit to the state it was in when this code was invoked. (The **reset** variable is used in this example to restore the initial state.)

Note the usage of:

```
rm "$path.out" > "$path.out"
<ol type="a">
```

The *pathchk* command has already verified, at this point, that **\$path.out** is not truncated.

With the *noclobber* option set, the shell verifies that **\$path.out** does not already exist before invoking *rm*.

If the shell succeeded in creating **\$path.out**, *rm* removes it so that the application can create the file again in the **PROCESSING** step.

If the **PROCESSING** step wants the file to exist already when it is invoked, the:

```
rm "$path.out" > "$path.out"
```

should be replaced with:


```
> "$path.out"
```

which verifies that the file did not already exist, but leaves **\$path.out** in place for use by **PROCESSING**.

RATIONALE

The *pathchk* utility was new for the ISO POSIX-2:1993 standard. It, along with the *set -C(noclobber)* option added to the shell, replaces the *mktemp*, *validfnam*, and *create* utilities that appeared in early proposals. All of these utilities were attempts to solve several common problems:

Verify the validity (for several different definitions of "valid") of a pathname supplied by a user, generated by an application, or imported from an external source.

Atomically create a file.

Perform various string handling functions to generate a temporary filename.

The *create* utility, included in an early proposal, provided checking and atomic creation in a single invocation of the utility; these are orthogonal issues and need not be grouped into a single utility. Note that the *noclobber* option also provides a way of creating a lock for process synchronization; since it provides an atomic *create*, there is no race between a test for existence and the following creation if it did not exist.

Having a function like *tmpnam()* in the ISO C standard is important in many high-level languages. The shell programming language, however, has built-in string manipulation facilities, making it very easy to construct temporary filenames. The names needed obviously depend on the application, but are frequently of a form similar to:

```
$TMPDIR/application_abbreviation$$.suffix
```

In cases where there is likely to be contention for a given suffix, a simple shell **for** or **while** loop can be used with the shell *noclobber* option to create a file without risk of collisions, as long as applications trying to use the same filename name space are cooperating on the use of files after they have been created.

FUTURE DIRECTIONS

None.

SEE ALSO

Redirection , *set* , *test*

COPYRIGHT

Portions of this text are reprinted and reproduced in electronic form from IEEE Std 1003.1, 2003 Edition, Standard for Information Technology -- Portable Operating System Interface (POSIX), The Open Group Base Specifications Issue 6, Copyright (C) 2001-2003 by the Institute of Electrical and Electronics Engineers, Inc and The Open Group. In the event of any discrepancy between this version and the original IEEE and The Open Group Standard, the original IEEE and The Open Group Standard is the referee document. The original Standard can be obtained online at <http://www.open-group.org/unix/online.html> .

NAME

pr - print files

SYNOPSIS

pr [+*page*][*-column*][*-adFmrt*][*-e*[*char*][*gap*]][*-h header*][*-i*[*char*][*gap*]]

[*-l lines*][*-n*[*char*][*width*]][*-o offset*][*-s*[*char*]][*-w width*][*-fp*]
[*file...*]

DESCRIPTION

The *pr* utility is a printing and pagination filter. If multiple input files are specified, each shall be read, formatted, and written to standard output. By default, the input shall be separated into 66-line pages, each with:

A 5-line header that includes the page number, date, time, and the pathname of the file

A 5-line trailer consisting of blank lines

If standard output is associated with a terminal, diagnostic messages shall be deferred until the *pr* utility has completed processing.

When options specifying multi-column output are specified, output text columns shall be of equal width; input lines that do not fit into a text column shall be truncated. By default, text columns shall be separated with at least one <blank>.

OPTIONS

The *pr* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section 12.2, Utility Syntax Guidelines, except that: the *page* option has a '+' delimiter; *page* and *column* can be multi-digit numbers; some of the option-arguments are optional; and some of the option-arguments cannot be specified as separate arguments from the preceding option letter. In particular, the *-s* option does not allow the option letter to be separated from its argument, and the options *-e*, *-i*, and *-n* require that both arguments, if present, not be separated from the option letter.

The following options shall be supported. In the following option descriptions, *column*, *lines*, *offset*, *page*, and *width* are positive decimal integers; *gap* is a non-negative decimal integer.

+page Begin output at page number *page* of the formatted input.

-column

Produce multi-column output that is arranged in *column* columns (the default shall be 1) and is written down each column in the order in which the text is received from the input file. This option should not be used with *-m*. The options *-e* and *-i* shall be assumed for multiple text-column output. Whether or not text columns are produced with identical vertical lengths is unspecified, but a text column shall never exceed the length of the page (see the *-l* option). When used with *-t*, use the minimum number of lines to write the output.

-a Modify the effect of the *-column* option so that the columns are filled across the page in a round-robin order (for example, when *column* is 2, the first input line heads column 1, the second heads column 2, the third is the second line in column 1, and so on).

-d Produce output that is double-spaced; append an extra <newline> following every <newline> found in the input.

-e[*char*][*gap*]

Expand each input <tab> to the next greater column position specified by the formula $n * gap + 1$, where *n* is an integer > 0. If *gap* is zero or is omitted, it shall default to 8. All <tab>s in the input shall be expanded into the appropriate number of <space>s. If any non-digit character, *char*, is specified, it shall be used as the input <tab>.

-f Use a <form-feed> for new pages, instead of the default behavior that uses a sequence of <newline>s. Pause before beginning the first page if the standard output is associated with a terminal.

-F Use a <form-feed> for new pages, instead of the default behavior that uses a sequence of <newline>s.

-h *header*

Use the string *header* to replace the contents of the *file* operand in the page header.

-i[*char*][*gap*]

In output, replace multiple <space>s with <tab>s wherever two or more adjacent <space>s reach column positions *gap*+1, 2* *gap*+1, 3* *gap*+1, and so on. If *gap* is zero or is omitted, default tab settings at every eighth column position shall be assumed. If any non-digit character, *char*, is specified, it shall be used as the output <tab>.

-l *lines*

Override the 66-line default and reset the page length to *lines*. If *lines* is not greater than the sum of both the header and trailer depths (in lines), the *pr* utility shall suppress both the header and trailer, as if the **-t** option were in effect.

-m

Merge files. Standard output shall be formatted so the *pr* utility writes one line from each file specified by a *file* operand, side by side into text columns of equal fixed widths, in terms of the number of column positions. Implementations shall support merging of at least nine *file* operands.

-n[*char*][*width*]

Provide *width*-digit line numbering (default for *width* shall be 5). The number shall occupy the first *width* column positions of each text column of default output or each line of **-m** output. If *char* (any non-digit character) is given, it shall be appended to the line number to separate it from whatever follows (default for *char* is a <tab>).

-o *offset*

Each line of output shall be preceded by offset <space>s. If the **-o** option is not specified, the default offset shall be zero. The space taken is in addition to the output line width (see the **-w** option below).

-p

Pause before beginning each page if the standard output is directed to a terminal (*pr* shall write an <alert> to standard error and wait for a <carriage-return> to be read on **/dev/tty**).

-r

Write no diagnostic reports on failure to open files.

-s[*char*]

Separate text columns by the single character *char* instead of by the appropriate number of <space>s (default for *char* shall be <tab>).

-t

Write neither the five-line identifying header nor the five-line trailer usually supplied for each page. Quit writing after the last line of each file without spacing to the end of the page.

-w *width*

Set the width of the line to *width* column positions for multiple text-column output only. If the **-w** option is not specified and the **-s** option is not specified, the default width shall be 72. If the **-w** option is not specified and the **-s** option is specified, the default width shall be 512.

For single column output, input lines shall not be truncated.

OPERANDS

The following operand shall be supported:

file A pathname of a file to be written. If no *file* operands are specified, or if a *file* operand is '**-**', the standard input shall be used.

STDIN

The standard input shall be used only if no *file* operands are specified, or if a *file* operand is '**-**'. See the INPUT FILES section.

INPUT FILES

The input files shall be text files.

The file **/dev/tty** shall be used to read responses required by the **-p** option.

ENVIRONMENT VARIABLES

The following environment variables shall affect the execution of *pr*:

LANG Provide a default value for the internationalization variables that are unset or null. (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2, Internationalization Variables for the precedence of internationalization variables used to determine the values of locale categories.)

LC_ALL

If set to a non-empty string value, override the values of all the other internationalization variables.

LC_CTYPE

Determine the locale for the interpretation of sequences of bytes of text data as characters (for example, single-byte as opposed to multi-byte characters in arguments and input files) and which characters are defined as printable (character class **print**). Non-printable characters are still written to standard output, but are not counted for the purpose for column-width and line-length calculations.

LC_MESSAGES

Determine the locale that should be used to affect the format and contents of diagnostic messages written to standard error.

LC_TIME

Determine the format of the date and time for use in writing header lines.

NLSPATH

Determine the location of message catalogs for the processing of **LC_MESSAGES**.

TZ

Determine the timezone used to calculate date and time strings written in header lines. If **TZ** is unset or null, an unspecified default timezone shall be used.

ASYNCHRONOUS EVENTS

If *pr* receives an interrupt while writing to a terminal, it shall flush all accumulated error messages to the screen before terminating.

STDOUT

The *pr* utility output shall be a paginated version of the original file (or files). This pagination shall be accomplished using either <form-feed>s or a sequence of <newline>s, as controlled by the **-F** or **-f** option. Page headers shall be generated unless the **-t** option is specified. The page headers shall be of the form:

```
"\n\n%s %s Page %d\n\n", <output of date>, <file>, <page number>
```

In the POSIX locale, the <output of date> field, representing the date and time of last modification of the input file (or the current date and time if the input file is standard input), shall be equivalent to the output of the following command as it would appear if executed at the given time:

```
date "+%b %e %H:%M %Y"
```

without the trailing <newline>, if the page being written is from standard input. If the page being written is not from standard input, in the POSIX locale, the same format shall be used, but the time used shall be the modification time of the file corresponding to *file* instead of the current time. When the **LC_TIME** locale category is not set to the POSIX locale, a different format and order of presentation of this field may be used.

If the standard input is used instead of a *file* operand, the <file> field shall be replaced by a null string.

If the **-h** option is specified, the <file> field shall be replaced by the *header* argument.

STDERR

The standard error shall be used for diagnostic messages and for alerting the terminal when **-p** is specified.

OUTPUT FILES

None.

EXTENDED DESCRIPTION

None.

EXIT STATUS

The following exit values shall be returned:

- 0 Successful completion.
- >0 An error occurred.

CONSEQUENCES OF ERRORS

Default.

The following sections are informative.

APPLICATION USAGE

None.

EXAMPLES

Print a numbered list of all files in the current directory:

```
ls -a | pr -n -h "Files in $(pwd)."
```

Print **file1** and **file2** as a double-spaced, three-column listing headed by "file list":

```
pr -3d -h "file list" file1 file2
```

Write **file1** on **file2**, expanding tabs to columns 10, 19, 28, ...:

```
pr -e9 -t <file1 >file2
```

RATIONALE

This utility is one of those that does not follow the Utility Syntax Guidelines because of its historical origins. The standard developers could have added new options that obeyed the guidelines (and marked the old options obsolescent) or devised an entirely new utility; there are examples of both actions in this volume of IEEE Std 1003.1-2001. Because of its widespread use by historical applications, the standard developers decided to exempt this version of *pr* from many of the guidelines.

Implementations are required to accept option-arguments to the **-h**, **-l**, **-o**, and **-w** options whether presented as part of the same argument or as a separate argument to *pr*, as suggested by the Utility Syntax Guidelines. The **-n** and **-s** options, however, are specified as in historical practice because they are frequently specified without their optional arguments. If a <blank> were allowed before the option-argument in these cases, a *file* operand could mistakenly be interpreted as an option-argument in historical applications.

The text about the minimum number of lines in multi-column output was included to ensure that a best effort is made in balancing the length of the columns. There are known historical implementations in which, for example, 60-line files are listed by *pr -2* as one column of 56 lines and a second of 4. Although this is not a problem when a full page with headers and trailers is produced, it would be relatively useless when used with **-t**.

Historical implementations of the *pr* utility have differed in the action taken for the **-f** option. BSD uses it as described here for the **-F** option; System V uses it to change trailing <newline>s on each page to a <form-feed> and, if standard output is a TTY device, sends an <alert> to standard error and reads a line from **/dev/tty** before the first page. There were strong arguments from both sides of this issue concerning historical practice and as a result the **-F** option was added. XSI-conformant systems support the System V historical actions for the **-f** option.

The <output of date> field in the **-l** format is specified only for the POSIX locale. As noted, the format can be different in other locales. No mechanism for defining this is present in this volume of IEEE Std 1003.1-2001, as the appropriate vehicle is a message catalog; that is, the format should be

specified as a "message".

FUTURE DIRECTIONS

None.

SEE ALSO

expand , *lp*

COPYRIGHT

Portions of this text are reprinted and reproduced in electronic form from IEEE Std 1003.1, 2003 Edition, Standard for Information Technology -- Portable Operating System Interface (POSIX), The Open Group Base Specifications Issue 6, Copyright (C) 2001-2003 by the Institute of Electrical and Electronics Engineers, Inc and The Open Group. In the event of any discrepancy between this version and the original IEEE and The Open Group Standard, the original IEEE and The Open Group Standard is the referee document. The original Standard can be obtained online at <http://www.open-group.org/unix/online.html> .

printf(P)

printf(P)

NAME

printf - write formatted output

SYNOPSIS

printf *format*[*argument...*]

DESCRIPTION

The *printf* utility shall write formatted operands to the standard output. The *argument* operands shall be formatted under control of the *format* operand.

OPTIONS

None.

OPERANDS

The following operands shall be supported:

format A string describing the format to use to write the remaining operands. See the EXTENDED DESCRIPTION section.

argument

The strings to be written to standard output, under the control of *format*. See the EXTENDED DESCRIPTION section.

STDIN

Not used.

INPUT FILES

None.

ENVIRONMENT VARIABLES

The following environment variables shall affect the execution of *printf*:

LANG Provide a default value for the internationalization variables that are unset or null. (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2, Internationalization Variables for the precedence of internationalization variables used to determine the values of locale categories.)

LC_ALL

If set to a non-empty string value, override the values of all the other internationalization variables.

LC_CTYPE

Determine the locale for the interpretation of sequences of bytes of text data as characters (for example, single-byte as opposed to multi-byte characters in arguments).

LC_MESSAGES

Determine the locale that should be used to affect the format and contents of diagnostic messages written to standard error.

LC_NUMERIC

Determine the locale for numeric formatting. It shall affect the format of numbers written using the **e**, **E**, **f**, **g**, and **G** conversion specifier characters (if supported).

NLSPATH

Determine the location of message catalogs for the processing of *LC_MESSAGES*.

ASYNCHRONOUS EVENTS

Default.

STDOUT

See the EXTENDED DESCRIPTION section.

STDERR

The standard error shall be used only for diagnostic messages.

OUTPUT FILES

None.

EXTENDED DESCRIPTION

The *format* operand shall be used as the *format* string described in the Base Definitions volume of IEEE Std 1003.1-2001, Chapter 5, File Format Notation with the following exceptions:

A <space> in the format string, in any context other than a flag of a conversion specification, shall be treated as an ordinary character that is copied to the output.

A " character in the format string shall be treated as a " character, not as a <space>.

In addition to the escape sequences shown in the Base Definitions volume of IEEE Std 1003.1-2001, Chapter 5, File Format Notation ('\', '\a', '\b', '\f', '\n', '\r', '\t', '\v'), "\ddd", where *ddd* is a one, two, or three-digit octal number, shall be written as a byte with the numeric value specified by the octal number.

The implementation shall not precede or follow output from the **d** or **u** conversion specifiers with <blank>s not specified by the *format* operand.

The implementation shall not precede output from the **o** conversion specifier with zeros not specified by the *format* operand.

The **e**, **E**, **f**, **g**, and **G** conversion specifiers need not be supported.

An additional conversion specifier character, **b**, shall be supported as follows. The argument shall be taken to be a string that may contain backslash-escape sequences. The following backslash-escape sequences shall be supported:

The escape sequences listed in the Base Definitions volume of IEEE Std 1003.1-2001, Chapter 5, File Format Notation ('\', '\a', '\b', '\f', '\n', '\r', '\t', '\v'), which shall be converted to the characters they represent

"\0ddd", where *ddd* is a zero, one, two, or three-digit octal number that shall be converted to a byte with the numeric value specified by the octal number

'\c', which shall not be written and shall cause *printf* to ignore any remaining characters in the string operand containing it, any remaining string operands, and any additional characters in the *format* operand

The interpretation of a backslash followed by any other sequence of characters is unspecified.

Bytes from the converted string shall be written until the end of the string or the number of bytes indicated by the precision specification is reached. If the precision is omitted, it shall be taken to be infinite, so all bytes up to the end of the converted string shall be written.

For each conversion specification that consumes an argument, the next argument operand shall be evaluated and converted to the appropriate type for the conversion as specified below.

The *format* operand shall be reused as often as necessary to satisfy the argument operands. Any extra **c** or **s** conversion specifiers shall be evaluated as if a null string argument were supplied; other extra conversion specifications shall be evaluated as if a zero argument were supplied. If the *format* operand contains no conversion specifications and *argument* operands are present, the results are unspecified.

If a character sequence in the *format* operand begins with a '%' character, but does not form a valid conversion specification, the behavior is unspecified.

The *argument* operands shall be treated as strings if the corresponding conversion specifier is **b**, **c**, or **s**; otherwise, it shall be evaluated as a C constant, as described by the ISO C standard, with the following extensions:

A leading plus or minus sign shall be allowed.

If the leading character is a single-quote or double-quote, the value shall be the numeric value in the underlying codeset of the character following the single-quote or double-quote.

If an argument operand cannot be completely converted into an internal value appropriate to the corresponding conversion specification, a diagnostic message shall be written to standard error and the utility shall not exit with a zero exit status, but shall continue processing any remaining operands and shall write the value accumulated at the time the error was detected to standard output.

It is not considered an error if an argument operand is not completely used for a **c** or **s** conversion or if a string operand's first or second character is used to get the numeric value of a character.

EXIT STATUS

The following exit values shall be returned:

- 0 Successful completion.
- >0 An error occurred.

CONSEQUENCES OF ERRORS

Default.

The following sections are informative.

APPLICATION USAGE

The floating-point formatting conversion specifications of *printf()* are not required because all arithmetic in the shell is integer arithmetic. The *awk* utility performs floating-point calculations and provides its own **printf** function. The *bc* utility can perform arbitrary-precision floating-point arithmetic, but does not provide extensive formatting capabilities. (This *printf* utility cannot really be used to format *bc* output; it does not support arbitrary precision.) Implementations are encouraged to support the floating-point conversions as an extension.

Note that this *printf* utility, like the *printf()* function defined in the System Interfaces volume of IEEE Std 1003.1-2001 on which it is based, makes no special provision for dealing with multi-byte characters when using the **%c** conversion specification or when a precision is specified in a **%b** or **%s** conversion specification. Applications should be extremely cautious using either of these features when there are multi-byte characters in the character set.

No provision is made in this volume of IEEE Std 1003.1-2001 which allows field widths and precisions to be specified as ****** since the ****** can be replaced directly in the *format* operand using shell variable substitution. Implementations can also provide this feature as an extension if they so choose.

Hexadecimal character constants as defined in the ISO C standard are not recognized in the *format* operand because there is no consistent way to detect the end of the constant. Octal character constants are limited to, at most, three octal digits, but hexadecimal character constants are only terminated by a non-hex-digit character. In the ISO C standard, the **###** concatenation operator can be used to terminate a constant and follow it with a hexadecimal character to be written. In the shell, concatenation occurs before the *printf* utility has a chance to parse the end of the hexadecimal constant.

The **%b** conversion specification is not part of the ISO C standard; it has been added here as a portable way to process backslash escapes expanded in string operands as provided by the *echo* utility. See also the APPLICATION USAGE section of *echo* for ways to use *printf* as a replacement for all of the traditional versions of the *echo* utility.

If an argument cannot be parsed correctly for the corresponding conversion specification, the *printf* utility is required to report an error. Thus, overflow and extraneous characters at the end of an argument being used for a numeric conversion shall be reported as errors.

EXAMPLES

To alert the user and then print and read a series of prompts:

```
printf "\aPlease fill in the following: \nName: "
read name
printf "Phone number: "
read phone
```

To read out a list of right and wrong answers from a file, calculate the percentage correctly, and print them out. The numbers are right-justified and separated by a single <tab>. The percentage is written to one decimal place of accuracy:

```
while read right wrong ; do
    percent=$(echo "scale=1;($right*100)/($right+$wrong)" | bc)
```

```
printf "%2d right\t%2d wrong\t(%%s%%)\n" \
    $right $wrong $percent
done < database_file
```

The command:

```
printf "%5d%4d\n" 1 21 321 4321 54321
```

produces:

```
1 21
3214321
54321 0
```

Note that the *format* operand is used three times to print all of the given strings and that a **'0'** was supplied by *printf* to satisfy the last **%4d** conversion specification.

The *printf* utility is required to notify the user when conversion errors are detected while producing numeric output; thus, the following results would be expected on an implementation with 32-bit two's-complement integers when **%d** is specified as the *format* operand:

Standard		
Argument	Output	Diagnostic Output
5a	5	printf: "5a" not completely converted
9999999999	2147483647	printf: "9999999999" arithmetic overflow
-9999999999	-2147483648	printf: "-9999999999" arithmetic overflow
ABC	0	printf: "ABC" expected numeric value

The diagnostic message format is not specified, but these examples convey the type of information that should be reported. Note that the value shown on standard output is what would be expected as the return value from the *strtol()* function as defined in the System Interfaces volume of IEEE Std 1003.1-2001. A similar correspondence exists between **%u** and *strtoul()* and **%e**, **%f**, and **%g** (if the implementation supports floating-point conversions) and *strtod()*.

In a locale using the ISO/IEC 646:1991 standard as the underlying codeset, the command:

```
printf "%d\n" 3 +3 -3 \'3 \'"+3 "'-3"
```

produces:

```
3      Numeric value of constant 3
3      Numeric value of constant 3
-3     Numeric value of constant -3
51     Numeric value of the character '3' in the ISO/IEC 646:1991 standard codeset
43     Numeric value of the character '+' in the ISO/IEC 646:1991 standard codeset
45     Numeric value of the character '-' in the ISO/IEC 646:1991 standard codeset
```

Note that in a locale with multi-byte characters, the value of a character is intended to be the value of the equivalent of the **wchar_t** representation of the character as described in the System Interfaces volume of IEEE Std 1003.1-2001.

RATIONALE

The *printf* utility was added to provide functionality that has historically been provided by *echo*. However, due to irreconcilable differences in the various versions of *echo* extant, the version has few special features, leaving those to this new *printf* utility, which is based on one in the Ninth Edition system.

The EXTENDED DESCRIPTION section almost exactly matches the *printf()* function in the ISO C standard, although it is described in terms of the file format notation in the Base Definitions volume of IEEE Std 1003.1-2001, Chapter 5, File Format Notation.

printf(P)

printf(P)

FUTURE DIRECTIONS

None.

SEE ALSO

awk , *bc* , *echo* , the System Interfaces volume of IEEE Std 1003.1-2001, *printf()*

COPYRIGHT

Portions of this text are reprinted and reproduced in electronic form from IEEE Std 1003.1, 2003 Edition, Standard for Information Technology -- Portable Operating System Interface (POSIX), The Open Group Base Specifications Issue 6, Copyright (C) 2001-2003 by the Institute of Electrical and Electronics Engineers, Inc and The Open Group. In the event of any discrepancy between this version and the original IEEE and The Open Group Standard, the original IEEE and The Open Group Standard is the referee document. The original Standard can be obtained online at <http://www.open-group.org/unix/online.html> .

NAME

pwd - return working directory name

SYNOPSIS

pwd [-L | -P]

DESCRIPTION

The *pwd* utility shall write to standard output an absolute pathname of the current working directory, which does not contain the filenames dot or dot-dot.

OPTIONS

The *pwd* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section 12.2, Utility Syntax Guidelines.

The following options shall be supported by the implementation:

- L** If the *PWD* environment variable contains an absolute pathname of the current directory that does not contain the filenames dot or dot-dot, *pwd* shall write this pathname to standard output. Otherwise, the **-L** option shall behave as the **-P** option.
- P** The absolute pathname written shall not contain filenames that, in the context of the pathname, refer to files of type symbolic link.

If both **-L** and **-P** are specified, the last one shall apply. If neither **-L** nor **-P** is specified, the *pwd* utility shall behave as if **-L** had been specified.

OPERANDS

None.

STDIN

Not used.

INPUT FILES

None.

ENVIRONMENT VARIABLES

The following environment variables shall affect the execution of *pwd*:

LANG Provide a default value for the internationalization variables that are unset or null. (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2, Internationalization Variables for the precedence of internationalization variables used to determine the values of locale categories.)

LC_ALL

If set to a non-empty string value, override the values of all the other internationalization variables.

LC_MESSAGES

Determine the locale that should be used to affect the format and contents of diagnostic messages written to standard error.

NLSPATH

Determine the location of message catalogs for the processing of *LC_MESSAGES*.

PWD

If the **-P** option is in effect, this variable shall be set to an absolute pathname of the current working directory that does not contain any components that specify symbolic links, does not contain any components that are dot, and does not contain any components that are dot-dot. If an application sets or unsets the value of *PWD*, the behavior of *pwd* is unspecified.

ASYNCHRONOUS EVENTS

Default.

STDOUT

The *pwd* utility output is an absolute pathname of the current working directory:

`"%s\n", <directory pathname>`

STDERR

The standard error shall be used only for diagnostic messages.

OUTPUT FILES

None.

EXTENDED DESCRIPTION

None.

EXIT STATUS

The following exit values shall be returned:

- | | |
|----|------------------------|
| 0 | Successful completion. |
| >0 | An error occurred. |

CONSEQUENCES OF ERRORS

If an error is detected, output shall not be written to standard output, a diagnostic message shall be written to standard error, and the exit status is not zero.

The following sections are informative.

APPLICATION USAGE

None.

EXAMPLES

None.

RATIONALE

Some implementations have historically provided *pwd* as a shell special built-in command.

In most utilities, if an error occurs, partial output may be written to standard output. This does not happen in historical implementations of *pwd*. Because *pwd* is frequently used in historical shell scripts without checking the exit status, it is important that the historical behavior is required here; therefore, the CONSEQUENCES OF ERRORS section specifically disallows any partial output being written to standard output.

FUTURE DIRECTIONS

None.

SEE ALSO

cd, the System Interfaces volume of IEEE Std 1003.1-2001, *getcwd()*

COPYRIGHT

Portions of this text are reprinted and reproduced in electronic form from IEEE Std 1003.1, 2003 Edition, Standard for Information Technology -- Portable Operating System Interface (POSIX), The Open Group Base Specifications Issue 6, Copyright (C) 2001-2003 by the Institute of Electrical and Electronics Engineers, Inc and The Open Group. In the event of any discrepancy between this version and the original IEEE and The Open Group Standard, the original IEEE and The Open Group Standard is the referee document. The original Standard can be obtained online at <http://www.open-group.org/unix/online.html>.

NAME

rm - remove directory entries

SYNOPSIS

rm [-fiRr] *file*...

DESCRIPTION

The *rm* utility shall remove the directory entry specified by each *file* argument.

If either of the files dot or dot-dot are specified as the basename portion of an operand (that is, the final pathname component), *rm* shall write a diagnostic message to standard error and do nothing more with such operands.

For each *file* the following steps shall be taken:

If the *file* does not exist: <ol type="a">

If the **-f** option is not specified, *rm* shall write a diagnostic message to standard error.

Go on to any remaining *files*.

If *file* is of type directory, the following steps shall be taken: <ol type="a">

If neither the **-R** option nor the **-r** option is specified, *rm* shall write a diagnostic message to standard error, do nothing more with *file*, and go on to any remaining files.

If the **-f** option is not specified, and either the permissions of *file* do not permit writing and the standard input is a terminal or the **-i** option is specified, *rm* shall write a prompt to standard error and read a line from the standard input. If the response is not affirmative, *rm* shall do nothing more with the current file and go on to any remaining files.

For each entry contained in *file*, other than dot or dot-dot, the four steps listed here (1 to 4) shall be taken with the entry as if it were a *file* operand. The *rm* utility shall not traverse directories by following symbolic links into other parts of the hierarchy, but shall remove the links themselves.

If the **-i** option is specified, *rm* shall write a prompt to standard error and read a line from the standard input. If the response is not affirmative, *rm* shall do nothing more with the current file, and go on to any remaining files.

If *file* is not of type directory, the **-f** option is not specified, and either the permissions of *file* do not permit writing and the standard input is a terminal or the **-i** option is specified, *rm* shall write a prompt to the standard error and read a line from the standard input. If the response is not affirmative, *rm* shall do nothing more with the current file and go on to any remaining files.

If the current file is a directory, *rm* shall perform actions equivalent to the *rmdir()* function defined in the System Interfaces volume of IEEE Std 1003.1-2001 called with a pathname of the current file used as the *path* argument. If the current file is not a directory, *rm* shall perform actions equivalent to the *unlink()* function defined in the System Interfaces volume of IEEE Std 1003.1-2001 called with a pathname of the current file used as the *path* argument.

If this fails for any reason, *rm* shall write a diagnostic message to standard error, do nothing more with the current file, and go on to any remaining files.

The *rm* utility shall be able to descend to arbitrary depths in a file hierarchy, and shall not fail due to path length limitations (unless an operand specified by the user exceeds system limitations).

OPTIONS

The *rm* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section 12.2, Utility Syntax Guidelines.

The following options shall be supported:

- f** Do not prompt for confirmation. Do not write diagnostic messages or modify the exit status in the case of nonexistent operands. Any previous occurrences of the **-i** option shall be ignored.
- i** Prompt for confirmation as described previously. Any previous occurrences of the **-f** option shall be ignored.
- R** Remove file hierarchies. See the DESCRIPTION.

-r Equivalent to **-R**.

OPERANDS

The following operand shall be supported:

file A pathname of a directory entry to be removed.

STDIN

The standard input shall be used to read an input line in response to each prompt specified in the STD-OUT section. Otherwise, the standard input shall not be used.

INPUT FILES

None.

ENVIRONMENT VARIABLES

The following environment variables shall affect the execution of *rm*:

LANG Provide a default value for the internationalization variables that are unset or null. (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2, Internationalization Variables for the precedence of internationalization variables used to determine the values of locale categories.)

LC_ALL If set to a non-empty string value, override the values of all the other internationalization variables.

LC_COLLATE

Determine the locale for the behavior of ranges, equivalence classes, and multi-character collating elements used in the extended regular expression defined for the **yesexpr** locale keyword in the **LC_MESSAGES** category.

LC_CTYPE

Determine the locale for the interpretation of sequences of bytes of text data as characters (for example, single-byte as opposed to multi-byte characters in arguments) and the behavior of character classes within regular expressions used in the extended regular expression defined for the **yesexpr** locale keyword in the **LC_MESSAGES** category.

LC_MESSAGES

Determine the locale for the processing of affirmative responses that should be used to affect the format and contents of diagnostic messages written to standard error.

NLSPATH

Determine the location of message catalogs for the processing of **LC_MESSAGES**.

ASYNCHRONOUS EVENTS

Default.

STDOUT

Not used.

STDERR

Prompts shall be written to standard error under the conditions specified in the DESCRIPTION and OPTIONS sections. The prompts shall contain the *file* pathname, but their format is otherwise unspecified. The standard error also shall be used for diagnostic messages.

OUTPUT FILES

None.

EXTENDED DESCRIPTION

None.

EXIT STATUS

The following exit values shall be returned:

- 0 All of the named directory entries for which *rm* performed actions equivalent to the *rmdir()* or *unlink()* functions were removed.
- >0 An error occurred.

CONSEQUENCES OF ERRORS

Default.

The following sections are informative.

APPLICATION USAGE

The *rm* utility is forbidden to remove the names dot and dot-dot in order to avoid the consequences of inadvertently doing something like:

rm -r .*

Some implementations do not permit the removal of the last link to an executable binary file that is being executed; see the [EBUSY] error in the *unlink()* function defined in the System Interfaces volume of IEEE Std 1003.1-2001. Thus, the *rm* utility can fail to remove such files.

The **-i** option causes *rm* to prompt and read the standard input even if the standard input is not a terminal, but in the absence of **-i** the mode prompting is not done when the standard input is not a terminal.

EXAMPLES

The following command:

rm a.out core

removes the directory entries: **a.out** and **core**.

The following command:

rm -Rf junk

removes the directory **junk** and all its contents, without prompting.

RATIONALE

For absolute clarity, paragraphs (2b) and (3) in the DESCRIPTION of *rm* describing the behavior when prompting for confirmation, should be interpreted in the following manner:

**if ((NOT f_option) AND
((not_writable AND input_is_terminal) OR i_option))**

The exact format of the interactive prompts is unspecified. Only the general nature of the contents of prompts are specified because implementations may desire more descriptive prompts than those used on historical implementations. Therefore, an application not using the **-f** option, or using the **-i** option, relies on the system to provide the most suitable dialog directly with the user, based on the behavior specified.

The **-r** option is historical practice on all known systems. The synonym **-R** option is provided for consistency with the other utilities in this volume of IEEE Std 1003.1-2001 that provide options requesting recursive descent through the file hierarchy.

The behavior of the **-f** option in historical versions of *rm* is inconsistent. In general, along with "forcing" the unlink without prompting for permission, it always causes diagnostic messages to be suppressed and the exit status to be unmodified for nonexistent operands and files that cannot be unlinked. In some versions, however, the **-f** option suppresses usage messages and system errors as well. Suppressing such messages is not a service to either shell scripts or users.

It is less clear that error messages regarding files that cannot be unlinked (removed) should be suppressed. Although this is historical practice, this volume of IEEE Std 1003.1-2001 does not permit the **-f** option to suppress such messages.

When given the **-r** and **-i** options, historical versions of *rm* prompt the user twice for each directory, once before removing its contents and once before actually attempting to delete the directory entry that names it. This allows the user to "prune" the file hierarchy walk. Historical versions of *rm* were inconsistent in that some did not do the former prompt for directories named on the command line and others had obscure prompting behavior when the **-i** option was specified and the permissions of the file did not permit writing. The POSIX Shell and Utilities *rm* differs little from historic practice, but does require that prompts be consistent. Historical versions of *rm* were also inconsistent in that prompts were done to both standard output and standard error. This volume of IEEE Std 1003.1-2001 requires that prompts be done to standard error, for consistency with *cp* and *mv*, and to allow historical extensions to *rm* that provide an option to list deleted files on standard output.

The *rm* utility is required to descend to arbitrary depths so that any file hierarchy may be deleted. This means, for example, that the *rm* utility cannot run out of file descriptors during its descent (that is, if the number of file descriptors is limited, *rm* cannot be implemented in the historical fashion where one file descriptor is used per directory level). Also, *rm* is not permitted to fail because of path length restrictions, unless an operand specified by the user is longer than {PATH_MAX}.

The *rm* utility removes symbolic links themselves, not the files they refer to, as a consequence of the dependence on the *unlink()* functionality, per the DESCRIPTION. When removing hierarchies with **-r** or **-R**, the prohibition on following symbolic links has to be made explicit.

FUTURE DIRECTIONS

None.

SEE ALSO

rmdir() , the System Interfaces volume of IEEE Std 1003.1-2001, *remove()*, *rmdir()*, *unlink()*

COPYRIGHT

Portions of this text are reprinted and reproduced in electronic form from IEEE Std 1003.1, 2003 Edition, Standard for Information Technology -- Portable Operating System Interface (POSIX), The Open Group Base Specifications Issue 6, Copyright (C) 2001-2003 by the Institute of Electrical and Electronics Engineers, Inc and The Open Group. In the event of any discrepancy between this version and the original IEEE and The Open Group Standard, the original IEEE and The Open Group Standard is the referee document. The original Standard can be obtained online at <http://www.open-group.org/unix/online.html> .

NAME

rmdir - remove directories

SYNOPSIS

rmdir [-p] *dir*...

DESCRIPTION

The *rmdir* utility shall remove the directory entry specified by each *dir* operand.

For each *dir* operand, the *rmdir* utility shall perform actions equivalent to the *rmdir()* function called with the *dir* operand as its only argument.

Directories shall be processed in the order specified. If a directory and a subdirectory of that directory are specified in a single invocation of the *rmdir* utility, the application shall specify the subdirectory before the parent directory so that the parent directory will be empty when the *rmdir* utility tries to remove it.

OPTIONS

The *rmdir* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section 12.2, Utility Syntax Guidelines.

The following option shall be supported:

-p Remove all directories in a pathname. For each *dir* operand:

The directory entry it names shall be removed.

If the *dir* operand includes more than one pathname component, effects equivalent to the following command shall occur:

rmdir -p \$(dirname *dir*)

OPERANDS

The following operand shall be supported:

dir A pathname of an empty directory to be removed.

STDIN

Not used.

INPUT FILES

None.

ENVIRONMENT VARIABLES

The following environment variables shall affect the execution of *rmdir*:

LANG Provide a default value for the internationalization variables that are unset or null. (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2, Internationalization Variables for the precedence of internationalization variables used to determine the values of locale categories.)

LC_ALL

If set to a non-empty string value, override the values of all the other internationalization variables.

LC_CTYPE

Determine the locale for the interpretation of sequences of bytes of text data as characters (for example, single-byte as opposed to multi-byte characters in arguments).

LC_MESSAGES

Determine the locale that should be used to affect the format and contents of diagnostic messages written to standard error.

NLSPATH

Determine the location of message catalogs for the processing of **LC_MESSAGES**.

ASYNCHRONOUS EVENTS

Default.

STDOUT

Not used.

STDERR

The standard error shall be used only for diagnostic messages.

OUTPUT FILES

None.

EXTENDED DESCRIPTION

None.

EXIT STATUS

The following exit values shall be returned:

- 0 Each directory entry specified by a *dir* operand was removed successfully.
- >0 An error occurred.

CONSEQUENCES OF ERRORS

Default.

The following sections are informative.

APPLICATION USAGE

The definition of an empty directory is one that contains, at most, directory entries for dot and dot-dot.

EXAMPLES

If a directory **a** in the current directory is empty except it contains a directory **b** and **a/b** is empty except it contains a directory **c**:

```
rmdir -p a/b/c
```

removes all three directories.

RATIONALE

On historical System V systems, the **-p** option also caused a message to be written to the standard output. The message indicated whether the whole path was removed or whether part of the path remained for some reason. The STDERR section requires this diagnostic when the entire path specified by a *dir* operand is not removed, but does not allow the status message reporting success to be written as a diagnostic.

The *rmdir* utility on System V also included a **-s** option that suppressed the informational message output by the **-p** option. This option has been omitted because the informational message is not specified by this volume of IEEE Std 1003.1-2001.

FUTURE DIRECTIONS

None.

SEE ALSO

rm, the System Interfaces volume of IEEE Std 1003.1-2001, *remove()*, *rmdir()*, *unlink()*

COPYRIGHT

Portions of this text are reprinted and reproduced in electronic form from IEEE Std 1003.1, 2003 Edition, Standard for Information Technology -- Portable Operating System Interface (POSIX), The Open Group Base Specifications Issue 6, Copyright (C) 2001-2003 by the Institute of Electrical and Electronics Engineers, Inc and The Open Group. In the event of any discrepancy between this version and the original IEEE and The Open Group Standard, the original IEEE and The Open Group Standard is the referee document. The original Standard can be obtained online at <http://www.open-group.org/unix/online.html>.

sleep(P)

sleep(P)

NAME

sleep - suspend execution for an interval

SYNOPSIS

sleep *time*

DESCRIPTION

The *sleep* utility shall suspend execution for at least the integral number of seconds specified by the *time* operand.

OPTIONS

None.

OPERANDS

The following operand shall be supported:

time A non-negative decimal integer specifying the number of seconds for which to suspend execution.

STDIN

Not used.

INPUT FILES

None.

ENVIRONMENT VARIABLES

The following environment variables shall affect the execution of *sleep*:

LANG Provide a default value for the internationalization variables that are unset or null. (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2, Internationalization Variables for the precedence of internationalization variables used to determine the values of locale categories.)

LC_ALL If set to a non-empty string value, override the values of all the other internationalization variables.

LC_CTYPE Determine the locale for the interpretation of sequences of bytes of text data as characters (for example, single-byte as opposed to multi-byte characters in arguments).

LC_MESSAGES Determine the locale that should be used to affect the format and contents of diagnostic messages written to standard error.

NLSPATH Determine the location of message catalogs for the processing of *LC_MESSAGES*.

ASYNCHRONOUS EVENTS

If the *sleep* utility receives a SIGALRM signal, one of the following actions shall be taken:

Terminate normally with a zero exit status.

Effectively ignore the signal.

Provide the default behavior for signals described in the ASYNCHRONOUS EVENTS section of *Utility Description Defaults*. This could include terminating with a non-zero exit status.

The *sleep* utility shall take the standard action for all other signals.

STDOUT

Not used.

STDERR

The standard error shall be used only for diagnostic messages.

sleep(P)

sleep(P)

OUTPUT FILES

None.

EXTENDED DESCRIPTION

None.

EXIT STATUS

The following exit values shall be returned:

- 0 The execution was successfully suspended for at least *time* seconds, or a SIGALRM signal was received. See the ASYNCHRONOUS EVENTS section.
- >0 An error occurred.

CONSEQUENCES OF ERRORS

Default.

The following sections are informative.

APPLICATION USAGE

None.

EXAMPLES

The *sleep* utility can be used to execute a command after a certain amount of time, as in:

```
(sleep 105; command) &
```

or to execute a command every so often, as in:

```
while true
do
    command sleep 37
done
```

RATIONALE

The exit status is allowed to be zero when *sleep* is interrupted by the SIGALRM signal because most implementations of this utility rely on the arrival of that signal to notify them that the requested finishing time has been successfully attained. Such implementations thus do not distinguish this situation from the successful completion case. Other implementations are allowed to catch the signal and go back to sleep until the requested time expires or to provide the normal signal termination procedures.

As with all other utilities that take integral operands and do not specify subranges of allowed values, *sleep* is required by this volume of IEEE Std 1003.1-2001 to deal with *time* requests of up to 2147483647 seconds. This may mean that some implementations have to make multiple calls to the delay mechanism of the underlying operating system if its argument range is less than this.

FUTURE DIRECTIONS

None.

SEE ALSO

wait, the System Interfaces volume of IEEE Std 1003.1-2001, *alarm()*, *sleep()*

COPYRIGHT

Portions of this text are reprinted and reproduced in electronic form from IEEE Std 1003.1, 2003 Edition, Standard for Information Technology -- Portable Operating System Interface (POSIX), The Open Group Base Specifications Issue 6, Copyright (C) 2001-2003 by the Institute of Electrical and Electronics Engineers, Inc and The Open Group. In the event of any discrepancy between this version and the original IEEE and The Open Group Standard, the original IEEE and The Open Group Standard is the referee document. The original Standard can be obtained online at <http://www.open-group.org/unix/online.html>.

NAME

sort - sort, merge, or sequence check text files

SYNOPSIS

sort [-m][**-o** *output*][**-bdfinru**][**-t** *char*][**-k** *keydef*]... [*file*...]

sort -c [**-bdfinru**][**-t** *char*][**-k** *keydef*][*file*]

DESCRIPTION

The *sort* utility shall perform one of the following functions:

Sort lines of all the named files together and write the result to the specified output.

Merge lines of all the named (presorted) files together and write the result to the specified output.

Check that a single input file is correctly presorted.

Comparisons shall be based on one or more sort keys extracted from each line of input (or, if no sort keys are specified, the entire line up to, but not including, the terminating <newline>), and shall be performed using the collating sequence of the current locale.

OPTIONS

The *sort* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section 12.2, Utility Syntax Guidelines, and the **-k** *keydef* option should follow the **-b**, **-d**, **-f**, **-i**, **-n**, and **-r** options.

The following options shall be supported:

- c** Check that the single input file is ordered as specified by the arguments and the collating sequence of the current locale. No output shall be produced; only the exit code shall be affected.
- m** Merge only; the input file shall be assumed to be already sorted.
- o** *output* Specify the name of an output file to be used instead of the standard output. This file can be the same as one of the input *files*.
- u** Unique: suppress all but one in each set of lines having equal keys. If used with the **-c** option, check that there are no lines with duplicate keys, in addition to checking that the input file is sorted.

The following options shall override the default ordering rules. When ordering options appear independent of any key field specifications, the requested field ordering rules shall be applied globally to all sort keys. When attached to a specific key (see **-k**), the specified ordering options shall override all global ordering options for that key.

- d** Specify that only <blank>s and alphanumeric characters, according to the current setting of *LC_CTYPE*, shall be significant in comparisons. The behavior is undefined for a sort key to which **-i** or **-n** also applies.
- f** Consider all lowercase characters that have uppercase equivalents, according to the current setting of *LC_CTYPE*, to be the uppercase equivalent for the purposes of comparison.
- i** Ignore all characters that are non-printable, according to the current setting of *LC_CTYPE*.
- n** Restrict the sort key to an initial numeric string, consisting of optional <blank>s, optional minus sign, and zero or more digits with an optional radix character and thousands separators (as defined in the current locale), which shall be sorted by arithmetic value. An empty digit string shall be treated as zero. Leading zeros and signs on zeros shall not affect ordering.
- r** Reverse the sense of comparisons.

The treatment of field separators can be altered using the options:

- b** Ignore leading <blank>s when determining the starting and ending positions of a restricted sort key. If the **-b** option is specified before the first **-k** option, it shall be applied to all **-k**

options. Otherwise, the **-b** option can be attached independently to each **-k** *field_start* or *field_end* option-argument (see below).

-t *char*

Use *char* as the field separator character; *char* shall not be considered to be part of a field (although it can be included in a sort key). Each occurrence of *char* shall be significant (for example, *<char><char>* delimits an empty field). If **-t** is not specified, *<blank>*s shall be used as default field separators; each maximal non-empty sequence of *<blank>*s that follows a non-*<blank>* shall be a field separator.

Sort keys can be specified using the options:

-k *keydef*

The *keydef* argument is a restricted sort key field definition. The format of this definition is:

field_start[*type*][*field_end*[*type*]]

where *field_start* and *field_end* define a key field restricted to a portion of the line (see the EXTENDED DESCRIPTION section), and *type* is a modifier from the list of characters **'b'**, **'d'**, **'f'**, **'i'**, **'n'**, **'r'**. The **'b'** modifier shall behave like the **-b** option, but shall apply only to the *field_start* or *field_end* to which it is attached. The other modifiers shall behave like the corresponding options, but shall apply only to the key field to which they are attached; they shall have this effect if specified with *field_start*, *field_end*, or both. If any modifier is attached to a *field_start* or to a *field_end*, no option shall apply to either. Implementations shall support at least nine occurrences of the **-k** option, which shall be significant in command line order. If no **-k** option is specified, a default sort key of the entire line shall be used.

When there are multiple key fields, later keys shall be compared only after all earlier keys compare equal. Except when the **-u** option is specified, lines that otherwise compare equal shall be ordered as if none of the options **-d**, **-f**, **-i**, **-n**, or **-k** were present (but with **-r** still in effect, if it was specified) and with all bytes in the lines significant to the comparison. The order in which lines that still compare equal are written is unspecified.

OPERANDS

The following operand shall be supported:

file A pathname of a file to be sorted, merged, or checked. If no *file* operands are specified, or if a *file* operand is **'-'**, the standard input shall be used.

STDIN

The standard input shall be used only if no *file* operands are specified, or if a *file* operand is **'-'**. See the INPUT FILES section.

INPUT FILES

The input files shall be text files, except that the *sort* utility shall add a *<newline>* to the end of a file ending with an incomplete last line.

ENVIRONMENT VARIABLES

The following environment variables shall affect the execution of *sort*:

LANG Provide a default value for the internationalization variables that are unset or null. (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2, Internationalization Variables for the precedence of internationalization variables used to determine the values of locale categories.)

LC_ALL

If set to a non-empty string value, override the values of all the other internationalization variables.

LC_COLLATE

Determine the locale for ordering rules.

LC_CTYPE

Determine the locale for the interpretation of sequences of bytes of text data as characters (for example, single-byte as opposed to multi-byte characters in arguments and input files) and the behavior of character classification for the **-b**, **-d**, **-f**, **-i**, and **-n** options.

LC_MESSAGES

Determine the locale that should be used to affect the format and contents of diagnostic messages written to standard error.

LC_NUMERIC

Determine the locale for the definition of the radix character and thousands separator for the **-n** option.

NLSPATH

Determine the location of message catalogs for the processing of *LC_MESSAGES*.

ASYNCHRONOUS EVENTS

Default.

STDOUT

Unless the **-o** or **-c** options are in effect, the standard output shall contain the sorted input.

STDERR

The standard error shall be used for diagnostic messages. A warning message about correcting an incomplete last line of an input file may be generated, but need not affect the final exit status.

OUTPUT FILES

If the **-o** option is in effect, the sorted input shall be written to the file *output*.

EXTENDED DESCRIPTION

The notation:

-k *field_start*[*type*][,*field_end*[*type*]]

shall define a key field that begins at *field_start* and ends at *field_end* inclusive, unless *field_start* falls beyond the end of the line or after *field_end*, in which case the key field is empty. A missing *field_end* shall mean the last character of the line.

A field comprises a maximal sequence of non-separating characters and, in the absence of option **-t**, any preceding field separator.

The *field_start* portion of the *keydef* option-argument shall have the form:

field_number[.*first_character*]

Fields and characters within fields shall be numbered starting with 1. The *field_number* and *first_character* pieces, interpreted as positive decimal integers, shall specify the first character to be used as part of a sort key. If *first_character* is omitted, it shall refer to the first character of the field.

The *field_end* portion of the *keydef* option-argument shall have the form:

field_number[.*last_character*]

The *field_number* shall be as described above for *field_start*. The *last_character* piece, interpreted as a non-negative decimal integer, shall specify the last character to be used as part of the sort key. If *last_character* evaluates to zero or *last_character* is omitted, it shall refer to the last character of the field specified by *field_number*.

If the **-b** option or **b** type modifier is in effect, characters within a field shall be counted from the first non- <blank> in the field. (This shall apply separately to *first_character* and *last_character*.)

EXIT STATUS

The following exit values shall be returned:

- 0 All input files were output successfully, or **-c** was specified and the input file was correctly sorted.
- 1 Under the **-c** option, the file was not ordered as specified, or if the **-c** and **-u** options were both specified, two input lines were found with equal keys.
- >1 An error occurred.

CONSEQUENCES OF ERRORS

Default.

The following sections are informative.

APPLICATION USAGE

The default value for **-t**, `<blank>`, has different properties from, for example, **-t "<space>"**. If a line contains:

```
<space><space>foo
```

the following treatment would occur with default separation as opposed to specifically selecting a `<space>`:

Field	Default	-t "<space>"
1	<code><space><space>foo</code>	<i>empty</i>
2	<i>empty</i>	<i>empty</i>
3	<i>empty</i>	foo

The leading field separator itself is included in a field when **-t** is not used. For example, this command returns an exit status of zero, meaning the input was already sorted:

```
sort -c -k 2 <<eof
y<tab>b
x<space>a
eof
```

(assuming that a `<tab>` precedes the `<space>` in the current collating sequence). The field separator is not included in a field when it is explicitly set via **-t**. This is historical practice and allows usage such as:

```
sort -t "|" -k 2n <<eof
Atlanta|425022|Georgia
Birmingham|284413|Alabama
Columbia|100385|South Carolina
eof
```

where the second field can be correctly sorted numerically without regard to the non-numeric field separator.

The wording in the OPTIONS section clarifies that the **-b**, **-d**, **-f**, **-i**, **-n**, and **-r** options have to come before the first sort key specified if they are intended to apply to all specified keys. The way it is described in this volume of IEEE Std 1003.1-2001 matches historical practice, not historical documentation. The results are unspecified if these options are specified after a **-k** option.

The **-f** option might not work as expected in locales where there is not a one-to-one mapping between an uppercase and a lowercase letter.

EXAMPLES

The following command sorts the contents of **infile** with the second field as the sort key:

sort -k 2,2 infile

The following command sorts, in reverse order, the contents of **infile1** and **infile2**, placing the output in **outfile** and using the second character of the second field as the sort key (assuming that the first character of the second field is the field separator):

sort -r -o outfile -k 2.2,2.2 infile1 infile2

The following command sorts the contents of **infile1** and **infile2** using the second non- <blank> of the second field as the sort key:

sort -k 2.2b,2.2b infile1 infile2

The following command prints the System V password file (user database) sorted by the numeric user ID (the third colon-separated field):

sort -t : -k 3,3n /etc/passwd

The following command prints the lines of the already sorted file **infile**, suppressing all but one occurrence of lines having the same third field:

sort -um -k 3.1,3.0 infile**RATIONALE**

Examples in some historical documentation state that options **-um** with one input file keep the first in each set of lines with equal keys. This behavior was deemed to be an implementation artifact and was not standardized.

The **-z** option was omitted; it is not standard practice on most systems and is inconsistent with using *sort* to sort several files individually and then merge them together. The text concerning **-z** in historical documentation appeared to require implementations to determine the proper buffer length during the sort phase of operation, but not during the merge.

The **-y** option was omitted because of non-portability. The **-M** option, present in System V, was omitted because of non-portability in international usage.

An undocumented **-T** option exists in some implementations. It is used to specify a directory for intermediate files. Implementations are encouraged to support the use of the *TMPDIR* environment variable instead of adding an option to support this functionality.

The **-k** option was added to satisfy two objections. First, the zero-based counting used by *sort* is not consistent with other utility conventions. Second, it did not meet syntax guideline requirements.

Historical documentation indicates that "setting **-n** implies **-b**". The description of **-n** already states that optional leading <blank>s are tolerated in doing the comparison. If **-b** is enabled, rather than implied, by **-n**, this has unusual side effects. When a character offset is used in a column of numbers (for example, to sort modulo 100), that offset is measured relative to the most significant digit, not to the column. Based upon a recommendation from the author of the original *sort* utility, the **-b** implication has been omitted from this volume of IEEE Std 1003.1-2001, and an application wishing to achieve the previously mentioned side effects has to code the **-b** flag explicitly.

FUTURE DIRECTIONS

None.

SEE ALSO

comm , *join* , *uniq* , the System Interfaces volume of IEEE Std 1003.1-2001, *toupper*()

COPYRIGHT

Portions of this text are reprinted and reproduced in electronic form from IEEE Std 1003.1, 2003 Edition, Standard for Information Technology -- Portable Operating System Interface (POSIX), The Open Group Base Specifications Issue 6, Copyright (C) 2001-2003 by the Institute of Electrical and Electronics Engineers, Inc and The Open Group. In the event of any discrepancy between this version and the

original IEEE and The Open Group Standard, the original IEEE and The Open Group Standard is the referee document. The original Standard can be obtained online at <http://www.open-group.org/unix/online.html> .

NAME

split - split files into pieces

SYNOPSIS

split [-l *line_count*][-a *suffix_length*][*file*[*name*]]

split -b *n*[*k*|*m*][-a *suffix_length*][*file*[*name*]]

DESCRIPTION

The *split* utility shall read an input file and write one or more output files. The default size of each output file shall be 1000 lines. The size of the output files can be modified by specification of the **-b** or **-l** options. Each output file shall be created with a unique suffix. The suffix shall consist of exactly *suffix_length* lowercase letters from the POSIX locale. The letters of the suffix shall be used as if they were a base-26 digit system, with the first suffix to be created consisting of all 'a' characters, the second with a 'b' replacing the last 'a', and so on, until a name of all 'z' characters is created. By default, the names of the output files shall be 'x', followed by a two-character suffix from the character set as described above, starting with "aa", "ab", "ac", and so on, and continuing until the suffix "zz", for a maximum of 676 files.

If the number of files required exceeds the maximum allowed by the suffix length provided, such that the last allowable file would be larger than the requested size, the *split* utility shall fail after creating the last file with a valid suffix; *split* shall not delete the files it created with valid suffixes. If the file limit is not exceeded, the last file created shall contain the remainder of the input file, and may be smaller than the requested size.

OPTIONS

The *split* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section 12.2, Utility Syntax Guidelines.

The following options shall be supported:

-a *suffix_length*

Use *suffix_length* letters to form the suffix portion of the filenames of the split file. If **-a** is not specified, the default suffix length shall be two. If the sum of the *name* operand and the *suffix_length* option-argument would create a filename exceeding {NAME_MAX} bytes, an error shall result; *split* shall exit with a diagnostic message and no files shall be created.

-b *n* Split a file into pieces *n* bytes in size.

-b *nk* Split a file into pieces *n**1024 bytes in size.

-b *nm* Split a file into pieces *n**1048576 bytes in size.

-l *line_count*

Specify the number of lines in each resulting file piece. The *line_count* argument is an unsigned decimal integer. The default is 1000. If the input does not end with a <newline>, the partial line shall be included in the last output file.

OPERANDS

The following operands shall be supported:

file The pathname of the ordinary file to be split. If no input file is given or *file* is '-', the standard input shall be used.

name The prefix to be used for each of the files resulting from the split operation. If no *name* argument is given, 'x' shall be used as the prefix of the output files. The combined length of the basename of *prefix* and *suffix_length* cannot exceed {NAME_MAX} bytes. See the OPTIONS section.

STDIN

See the INPUT FILES section.

INPUT FILES

Any file can be used as input.

ENVIRONMENT VARIABLES

The following environment variables shall affect the execution of *split*:

LANG Provide a default value for the internationalization variables that are unset or null. (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2, Internationalization Variables for the precedence of internationalization variables used to determine the values of locale categories.)

LC_ALL

If set to a non-empty string value, override the values of all the other internationalization variables.

LC_CTYPE

Determine the locale for the interpretation of sequences of bytes of text data as characters (for example, single-byte as opposed to multi-byte characters in arguments and input files).

LC_MESSAGES

Determine the locale that should be used to affect the format and contents of diagnostic messages written to standard error.

NLSPATH

Determine the location of message catalogs for the processing of **LC_MESSAGES**.

ASYNCHRONOUS EVENTS

Default.

STDOUT

Not used.

STDERR

The standard error shall be used only for diagnostic messages.

OUTPUT FILES

The output files contain portions of the original input file; otherwise, unchanged.

EXTENDED DESCRIPTION

None.

EXIT STATUS

The following exit values shall be returned:

- 0 Successful completion.
- >0 An error occurred.

CONSEQUENCES OF ERRORS

Default.

The following sections are informative.

APPLICATION USAGE

None.

EXAMPLES

In the following examples **foo** is a text file that contains 5000 lines.

Create five files, **xaa**, **xab**, **xac**, **xad**, and **xae**:

split foo

Create five files, but the suffixed portion of the created files consists of three letters, **xaaa**, **xaab**, **xaac**, **xaad**, and **xaae**:

split -a 3 foo

Create three files with four-letter suffixes and a supplied prefix, **bar_aaaa**, **bar_aaab**, and **bar_aaac**:

split -a 4 -l 2000 foo bar_

Create as many files as are necessary to contain at most 20*1024 bytes, each with the default prefix of **x** and a five-letter suffix:

split -a 5 -b 20k foo

RATIONALE

The **-b** option was added to provide a mechanism for splitting files other than by lines. While most uses of the **-b** option are for transmitting files over networks, some believed it would have additional uses.

The **-a** option was added to overcome the limitation of being able to create only 676 files.

Consideration was given to deleting this utility, using the rationale that the functionality provided by this utility is available via the *csplit* utility (see *csplit*). Upon reconsideration of the purpose of the User Portability Extension, it was decided to retain both this utility and the *csplit* utility because users use both utilities and have historical expectations of their behavior. Furthermore, the splitting on byte boundaries in *split* cannot be duplicated with the historical *csplit*.

The text "*split* shall not delete the files it created with valid suffixes" would normally be assumed, but since the related utility, *csplit*, does delete files under some circumstances, the historical behavior of *split* is made explicit to avoid misinterpretation.

FUTURE DIRECTIONS

None.

SEE ALSO

csplit

COPYRIGHT

Portions of this text are reprinted and reproduced in electronic form from IEEE Std 1003.1, 2003 Edition, Standard for Information Technology -- Portable Operating System Interface (POSIX), The Open Group Base Specifications Issue 6, Copyright (C) 2001-2003 by the Institute of Electrical and Electronics Engineers, Inc and The Open Group. In the event of any discrepancy between this version and the original IEEE and The Open Group Standard, the original IEEE and The Open Group Standard is the referee document. The original Standard can be obtained online at <http://www.open-group.org/unix/online.html> .

NAME

stty - set the options for a terminal

SYNOPSIS

stty [-a| -g]

stty *operands*

DESCRIPTION

The *stty* utility shall set or report on terminal I/O characteristics for the device that is its standard input. Without options or operands specified, it shall report the settings of certain characteristics, usually those that differ from implementation-defined defaults. Otherwise, it shall modify the terminal state according to the specified operands. Detailed information about the modes listed in the first five groups below are described in the Base Definitions volume of IEEE Std 1003.1-2001, Chapter 11, General Terminal Interface. Operands in the Combination Modes group (see Combination Modes) are implemented using operands in the previous groups. Some combinations of operands are mutually-exclusive on some terminal types; the results of using such combinations are unspecified.

Typical implementations of this utility require a communications line configured to use the **termios** interface defined in the System Interfaces volume of IEEE Std 1003.1-2001. On systems where none of these lines are available, and on lines not currently configured to support the **termios** interface, some of the operands need not affect terminal characteristics.

OPTIONS

The *stty* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section 12.2, Utility Syntax Guidelines.

The following options shall be supported:

- a** Write to standard output all the current settings for the terminal.
- g** Write to standard output all the current settings in an unspecified form that can be used as arguments to another invocation of the *stty* utility on the same system. The form used shall not contain any characters that would require quoting to avoid word expansion by the shell; see *Word Expansions* .

OPERANDS

The following operands shall be supported to set the terminal characteristics.

Control Modes**parenb (-parenb)**

Enable (disable) parity generation and detection. This shall have the effect of setting (not setting) PARENb in the **termios** *c_flag* field, as defined in the Base Definitions volume of IEEE Std 1003.1-2001, Chapter 11, General Terminal Interface.

parodd (-parodd)

Select odd (even) parity. This shall have the effect of setting (not setting) PARODD in the **termios** *c_flag* field, as defined in the Base Definitions volume of IEEE Std 1003.1-2001, Chapter 11, General Terminal Interface.

cs5 cs6 cs7 cs8

Select character size, if possible. This shall have the effect of setting CS5, CS6, CS7, and CS8, respectively, in the **termios** *c_flag* field, as defined in the Base Definitions volume of IEEE Std 1003.1-2001, Chapter 11, General Terminal Interface.

number Set terminal baud rate to the number given, if possible. If the baud rate is set to zero, the modem control lines shall no longer be asserted. This shall have the effect of setting the input and output **termios** baud rate values as defined in the Base Definitions volume of IEEE Std 1003.1-2001, Chapter 11, General Terminal Interface.

ispeed *number*

Set terminal input baud rate to the number given, if possible. If the input baud rate is set to zero, the input baud rate shall be specified by the value of the output baud rate. This shall have the effect of setting the input **termios** baud rate values as defined in the Base Definitions volume of IEEE Std 1003.1-2001, Chapter 11, General Terminal Interface.

ospeed *number*

Set terminal output baud rate to the number given, if possible. If the output baud rate is set to zero, the modem control lines shall no longer be asserted. This shall have the effect of setting the output **termios** baud rate values as defined in the Base Definitions volume of IEEE Std 1003.1-2001, Chapter 11, General Terminal Interface.

hupcl (**-hupcl**)

Stop asserting modem control lines (do not stop asserting modem control lines) on last close. This shall have the effect of setting (not setting) HUPCL in the **termios** *c_cflag* field, as defined in the Base Definitions volume of IEEE Std 1003.1-2001, Chapter 11, General Terminal Interface.

hup (**-hup**)

Equivalent to **hupcl** (**-hupcl**).

cstopb (**-cstopb**)

Use two (one) stop bits per character. This shall have the effect of setting (not setting) CSTOPB in the **termios** *c_cflag* field, as defined in the Base Definitions volume of IEEE Std 1003.1-2001, Chapter 11, General Terminal Interface.

cread (**-cread**)

Enable (disable) the receiver. This shall have the effect of setting (not setting) CREAD in the **termios** *c_cflag* field, as defined in the Base Definitions volume of IEEE Std 1003.1-2001, Chapter 11, General Terminal Interface.

clocal (**-clocal**)

Assume a line without (with) modem control. This shall have the effect of setting (not setting) CLOCAL in the **termios** *c_cflag* field, as defined in the Base Definitions volume of IEEE Std 1003.1-2001, Chapter 11, General Terminal Interface.

It is unspecified whether *stty* shall report an error if an attempt to set a Control Mode fails.

Input Modes**ignbrk** (**-ignbrk**)

Ignore (do not ignore) break on input. This shall have the effect of setting (not setting) IGNBRK in the **termios** *c_iflag* field, as defined in the Base Definitions volume of IEEE Std 1003.1-2001, Chapter 11, General Terminal Interface.

brkint (**-brkint**)

Signal (do not signal) INTR on break. This shall have the effect of setting (not setting) BRKINT in the **termios** *c_iflag* field, as defined in the Base Definitions volume of IEEE Std 1003.1-2001, Chapter 11, General Terminal Interface.

ignpar (**-ignpar**)

Ignore (do not ignore) bytes with parity errors. This shall have the effect of setting (not setting) IGNPAR in the **termios** *c_iflag* field, as defined in the Base Definitions volume of IEEE Std 1003.1-2001, Chapter 11, General Terminal Interface.

parmrk (**-parmrk**)

Mark (do not mark) parity errors. This shall have the effect of setting (not setting) PARMRK in the **termios** *c_iflag* field, as defined in the Base Definitions volume of IEEE Std 1003.1-2001, Chapter 11, General Terminal Interface.

inpck (**-inpck**)

Enable (disable) input parity checking. This shall have the effect of setting (not setting) INPCK in the **termios** *c_iflag* field, as defined in the Base Definitions volume of IEEE Std 1003.1-2001, Chapter 11, General Terminal Interface.

istrip (-istrip)

Strip (do not strip) input characters to seven bits. This shall have the effect of setting (not setting) ISTRIP in the **termios** *c_iflag* field, as defined in the Base Definitions volume of IEEE Std 1003.1-2001, Chapter 11, General Terminal Interface.

inlcr (-inlcr)

Map (do not map) NL to CR on input. This shall have the effect of setting (not setting) INLCR in the **termios** *c_iflag* field, as defined in the Base Definitions volume of IEEE Std 1003.1-2001, Chapter 11, General Terminal Interface.

igncr (-igncr)

Ignore (do not ignore) CR on input. This shall have the effect of setting (not setting) IGNCR in the **termios** *c_iflag* field, as defined in the Base Definitions volume of IEEE Std 1003.1-2001, Chapter 11, General Terminal Interface.

icrnl (-icrnl)

Map (do not map) CR to NL on input. This shall have the effect of setting (not setting) ICRNL in the **termios** *c_iflag* field, as defined in the Base Definitions volume of IEEE Std 1003.1-2001, Chapter 11, General Terminal Interface.

ixon (-ixon)

Enable (disable) START/STOP output control. Output from the system is stopped when the system receives STOP and started when the system receives START. This shall have the effect of setting (not setting) IXON in the **termios** *c_iflag* field, as defined in the Base Definitions volume of IEEE Std 1003.1-2001, Chapter 11, General Terminal Interface.

ixany (-ixany)

Allow any character to restart output. This shall have the effect of setting (not setting) IXANY in the **termios** *c_iflag* field, as defined in the Base Definitions volume of IEEE Std 1003.1-2001, Chapter 11, General Terminal Interface.

ixoff (-ixoff)

Request that the system send (not send) STOP characters when the input queue is nearly full and START characters to resume data transmission. This shall have the effect of setting (not setting) IXOFF in the **termios** *c_iflag* field, as defined in the Base Definitions volume of IEEE Std 1003.1-2001, Chapter 11, General Terminal Interface.

Output Modes**opost (-opost)**

Post-process output (do not post-process output; ignore all other output modes). This shall have the effect of setting (not setting) OPOST in the **termios** *c_oflag* field, as defined in the Base Definitions volume of IEEE Std 1003.1-2001, Chapter 11, General Terminal Interface.

ocrnl (-ocrnl)

Map (do not map) CR to NL on output. This shall have the effect of setting (not setting) OCRNL in the **termios** *c_oflag* field, as defined in the Base Definitions volume of IEEE Std 1003.1-2001, Chapter 11, General Terminal Interface.

onocr (-onocr)

Do not (do) output CR at column zero. This shall have the effect of setting (not setting) ONOCR in the **termios** *c_oflag* field, as defined in the Base Definitions volume of IEEE Std 1003.1-2001, Chapter 11, General Terminal Interface.

onlret (-onlret)

The terminal newline key performs (does not perform) the CR function. This shall have the effect of setting (not setting) ONLRET in the **termios** *c_oflag* field, as defined in the Base Definitions volume of IEEE Std 1003.1-2001, Chapter 11, General Terminal Interface.

ofill (-ofill)

Use fill characters (use timing) for delays. This shall have the effect of setting (not setting) OFILL in the **termios** *c_oflag* field, as defined in the Base Definitions volume of IEEE Std 1003.1-2001, Chapter 11, General Terminal Interface.

ofdel (-ofdel)

Fill characters are DELs (NULs). This shall have the effect of setting (not setting) OFDEL in the **termios** *c_oflag* field, as defined in the Base Definitions volume of IEEE Std 1003.1-2001, Chapter 11, General Terminal Interface.

cr0 cr1 cr2 cr3

Select the style of delay for CRs. This shall have the effect of setting CRDLY to CR0, CR1, CR2, or CR3, respectively, in the **termios** *c_oflag* field, as defined in the Base Definitions volume of IEEE Std 1003.1-2001, Chapter 11, General Terminal Interface.

nl0 nl1

Select the style of delay for NL. This shall have the effect of setting NLDLY to NL0 or NL1, respectively, in the **termios** *c_oflag* field, as defined in the Base Definitions volume of IEEE Std 1003.1-2001, Chapter 11, General Terminal Interface.

tab0 tab1 tab2 tab3

Select the style of delay for horizontal tabs. This shall have the effect of setting TABDLY to TAB0, TAB1, TAB2, or TAB3, respectively, in the **termios** *c_oflag* field, as defined in the Base Definitions volume of IEEE Std 1003.1-2001, Chapter 11, General Terminal Interface. Note that TAB3 has the effect of expanding <tab>s to <space>s.

tabs (-tabs)

Synonym for **tab0 (tab3)**.

bs0 bs1

Select the style of delay for backspaces. This shall have the effect of setting BSDLY to BS0 or BS1, respectively, in the **termios** *c_oflag* field, as defined in the Base Definitions volume of IEEE Std 1003.1-2001, Chapter 11, General Terminal Interface.

ff0 ff1 Select the style of delay for form-feeds. This shall have the effect of setting FFDLY to FF0 or FF1, respectively, in the **termios** *c_oflag* field, as defined in the Base Definitions volume of IEEE Std 1003.1-2001, Chapter 11, General Terminal Interface.

vt0 vt1

Select the style of delay for vertical-tabs. This shall have the effect of setting VTDLY to VT0 or VT1, respectively, in the **termios** *c_oflag* field, as defined in the Base Definitions volume of IEEE Std 1003.1-2001, Chapter 11, General Terminal Interface.

Local Modes**isig (-isig)**

Enable (disable) the checking of characters against the special control characters INTR, QUIT, and SUSP. This shall have the effect of setting (not setting) ISIG in the **termios** *c_lflag* field, as defined in the Base Definitions volume of IEEE Std 1003.1-2001, Chapter 11, General Terminal Interface.

icanon (-icanon)

Enable (disable) canonical input (ERASE and KILL processing). This shall have the effect of setting (not setting) ICANON in the **termios** *c_lflag* field, as defined in the Base Definitions volume of IEEE Std 1003.1-2001, Chapter 11, General Terminal Interface.

ixexten (-ixexten)

Enable (disable) any implementation-defined special control characters not currently controlled by **icanon**, **isig**, **ixon**, or **ixoff**. This shall have the effect of setting (not setting) IEXTEN in the **termios** *c_lflag* field, as defined in the Base Definitions volume of IEEE Std 1003.1-2001, Chapter 11, General Terminal Interface.

echo (-echo)

Echo back (do not echo back) every character typed. This shall have the effect of setting (not setting) ECHO in the **termios** *c_lflag* field, as defined in the Base Definitions volume of IEEE Std 1003.1-2001, Chapter 11, General Terminal Interface.

echoe (-echoe)

The ERASE character visually erases (does not erase) the last character in the current line from the display, if possible. This shall have the effect of setting (not setting) ECHOE in the

termios *c_lflag* field, as defined in the Base Definitions volume of IEEE Std 1003.1-2001, Chapter 11, General Terminal Interface.

echok (-echok)

Echo (do not echo) NL after KILL character. This shall have the effect of setting (not setting) ECHOK in the **termios** *c_lflag* field, as defined in the Base Definitions volume of IEEE Std 1003.1-2001, Chapter 11, General Terminal Interface.

echonl (-echonl)

Echo (do not echo) NL, even if **echo** is disabled. This shall have the effect of setting (not setting) ECHONL in the **termios** *c_lflag* field, as defined in the Base Definitions volume of IEEE Std 1003.1-2001, Chapter 11, General Terminal Interface.

noflsh (-noflsh)

Disable (enable) flush after INTR, QUIT, SUSP. This shall have the effect of setting (not setting) NOFLSH in the **termios** *c_lflag* field, as defined in the Base Definitions volume of IEEE Std 1003.1-2001, Chapter 11, General Terminal Interface.

tostop (-tostop)

Send SIGTTOU for background output. This shall have the effect of setting (not setting) TOSTOP in the **termios** *c_lflag* field, as defined in the Base Definitions volume of IEEE Std 1003.1-2001, Chapter 11, General Terminal Interface.

Special Control Character Assignments

<control>-character string

Set *<control>-character* to *string*. If *<control>-character* is one of the character sequences in the first column of the following table, the corresponding Base Definitions volume of IEEE Std 1003.1-2001, Chapter 11, General Terminal Interface control character from the second column shall be recognized. This has the effect of setting the corresponding element of the **termios** *c_cc* array (see the Base Definitions volume of IEEE Std 1003.1-2001, Chapter 13, Headers, *<termios.h>*).

Table: Control Character Names in stty

Control Character	c_cc Subscript	Description
eof	<i>VEOF</i>	<i>EOF character</i>
eol	<i>VEOL</i>	<i>EOL character</i>
erase	<i>VERASE</i>	<i>ERASE character</i>
intr	<i>VINTR</i>	<i>INTR character</i>
kill	<i>VKILL</i>	<i>KILL character</i>
quit	<i>VQUIT</i>	<i>QUIT character</i>
susp	<i>VSUSP</i>	<i>SUSP character</i>
start	<i>VSTART</i>	<i>START character</i>
stop	<i>VSTOP</i>	<i>STOP character</i>

If *string* is a single character, the control character shall be set to that character. If *string* is the two-character sequence "^-" or the string *undef*, the control character shall be set to _POSIX_VDISABLE, if it is in effect for the device; if _POSIX_VDISABLE is not in effect for the device, it shall be treated as an error. In the POSIX locale, if *string* is a two-character sequence beginning with circumflex ('^'), and the second character is one of those listed in the "c" column of the following table, the control character shall be set to the corresponding character value in the Value column of the table. **Table:**

Circumflex Control Characters in stty

^c	Value	^c	Value	^c	Value
<i>a, A</i>	<i><SOH></i>	<i>l, L</i>	<i><FF></i>	<i>w, W</i>	<i><ETB></i>
<i>b, B</i>	<i><STX></i>	<i>m, M</i>	<i><CR></i>	<i>x, X</i>	<i><CAN></i>
<i>c, C</i>	<i><ETX></i>	<i>n, N</i>	<i><SO></i>	<i>y, Y</i>	<i></i>
<i>d, D</i>	<i><EOT></i>	<i>o, O</i>	<i><SI></i>	<i>z, Z</i>	<i><SUB></i>
<i>e, E</i>	<i><ENQ></i>	<i>p, P</i>	<i><DLE></i>	<i>[</i>	<i><ESC></i>
<i>f, F</i>	<i><ACK></i>	<i>q, Q</i>	<i><DC1></i>	<i>\</i>	<i><FS></i>
<i>g, G</i>	<i><BEL></i>	<i>r, R</i>	<i><DC2></i>	<i>]</i>	<i><GS></i>
<i>h, H</i>	<i><BS></i>	<i>s, S</i>	<i><DC3></i>	<i>^</i>	<i><RS></i>

<i>i, I</i>	<HT>	<i>t, T</i>	<DC4>	<i>_</i>	<US>
<i>j, J</i>	<LF>	<i>u, U</i>	<NAK>	<i>?</i>	
<i>k, K</i>	<VT>	<i>v, V</i>	<SYN>		

min *number*

Set the value of MIN to *number*. MIN is used in non-canonical mode input processing (**icanon**).

time *number*

Set the value of TIME to *number*. TIME is used in non-canonical mode input processing (**icanon**).

Combination Modes

saved settings

Set the current terminal characteristics to the saved settings produced by the **-g** option.

evenp or **parity**

Enable **parenb** and **cs7**; disable **parodd**.

oddp

Enable **parenb**, **cs7**, and **parodd**.

-parity, **-evenp**, or **-oddp**

Disable **parenb**, and set **cs8**.

raw (**-raw** or **cooked**)

Enable (disable) raw input and output. Raw mode shall be equivalent to setting:

```
stty cs8 erase ^ - kill ^ - intr ^ - \
quit ^ - eof ^ - eol ^ - -post -inpck
```

nl (**-nl**)

Disable (enable) **icrnl**. In addition, **-nl** unsets **inlcr** and **igncr**.

ek Reset ERASE and KILL characters back to system defaults.

sane

Reset all modes to some reasonable, unspecified, values.

STDIN

Although no input is read from standard input, standard input shall be used to get the current terminal I/O characteristics and to set new terminal I/O characteristics.

INPUT FILES

None.

ENVIRONMENT VARIABLES

The following environment variables shall affect the execution of *stty*:

LANG Provide a default value for the internationalization variables that are unset or null. (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2, Internationalization Variables for the precedence of internationalization variables used to determine the values of locale categories.)

LC_ALL

If set to a non-empty string value, override the values of all the other internationalization variables.

LC_CTYPE

This variable determines the locale for the interpretation of sequences of bytes of text data as characters (for example, single-byte as opposed to multi-byte characters in arguments) and which characters are in the class **print**.

LC_MESSAGES

Determine the locale that should be used to affect the format and contents of diagnostic messages written to standard error.

NLSPATH

Determine the location of message catalogs for the processing of *LC_MESSAGES*.

ASYNCHRONOUS EVENTS

Default.

STDOUT

If operands are specified, no output shall be produced.

If the **-g** option is specified, *stty* shall write to standard output the current settings in a form that can be used as arguments to another instance of *stty* on the same system.

If the **-a** option is specified, all of the information as described in the OPERANDS section shall be written to standard output. Unless otherwise specified, this information shall be written as <space>-separated tokens in an unspecified format, on one or more lines, with an unspecified number of tokens per line. Additional information may be written.

If no options or operands are specified, an unspecified subset of the information written for the **-a** option shall be written.

If speed information is written as part of the default output, or if the **-a** option is specified and if the terminal input speed and output speed are the same, the speed information shall be written as follows:

"speed %d baud;", <speed>

Otherwise, speeds shall be written as:

"ispeed %d baud; ospeed %d baud;", <ispeed>, <ospeed>

In locales other than the POSIX locale, the word **baud** may be changed to something more appropriate in those locales.

If control characters are written as part of the default output, or if the **-a** option is specified, control characters shall be written as:

"%s = %s;", <control-character name>, <value>

where <value> is either the character, or some visual representation of the character if it is non-printable, or the string *undef* if the character is disabled.

STDERR

The standard error shall be used only for diagnostic messages.

OUTPUT FILES

None.

EXTENDED DESCRIPTION

None.

EXIT STATUS

The following exit values shall be returned:

- 0 The terminal options were read or set successfully.
- >0 An error occurred.

CONSEQUENCES OF ERRORS

Default.

The following sections are informative.

APPLICATION USAGE

The **-g** flag is designed to facilitate the saving and restoring of terminal state from the shell level. For example, a program may:

```

saveterm="$ (stty -g)"    # save terminal state
stty (new settings)      # set new state
...                      # ...
stty $saveterm           # restore terminal state

```

Since the format is unspecified, the saved value is not portable across systems.

Since the **-a** format is so loosely specified, scripts that save and restore terminal settings should use the **-g** option.

EXAMPLES

None.

RATIONALE

The original *stty* description was taken directly from System V and reflected the System V terminal driver **termio**. It has been modified to correspond to the terminal driver **termios**.

Output modes are specified only for XSI-conformant systems. All implementations are expected to provide *stty* operands corresponding to all of the output modes they support.

The *stty* utility is primarily used to tailor the user interface of the terminal, such as selecting the preferred ERASE and KILL characters. As an application programming utility, *stty* can be used within shell scripts to alter the terminal settings for the duration of the script.

The **termios** section states that individual disabling of control characters is possible through the option `_POSIX_VDISABLE`. If enabled, two conventions currently exist for specifying this: System V uses `"^"`, and BSD uses *undef*. Both are accepted by *stty* in this volume of IEEE Std 1003.1-2001. The other BSD convention of using the letter **'u'** was rejected because it conflicts with the actual letter **'u'**, which is an acceptable value for a control character.

Early proposals did not specify the mapping of `"^c"` to control characters because the control characters were not specified in the POSIX locale character set description file requirements. The control character set is now specified in the Base Definitions volume of IEEE Std 1003.1-2001, Chapter 3, Definitions so the historical mapping is specified. Note that although the mapping corresponds to control-character key assignments on many terminals that use the ISO/IEC 646:1991 standard (or ASCII) character encodings, the mapping specified here is to the control characters, not their keyboard encodings.

Since **termios** supports separate speeds for input and output, two new options were added to specify each distinctly.

Some historical implementations use standard input to get and set terminal characteristics; others use standard output. Since input from a login TTY is usually restricted to the owner while output to a TTY is frequently open to anyone, using standard input provides fewer chances of accidentally (or maliciously) altering the terminal settings of other users. Using standard input also allows *stty -a* and *stty -g* output to be redirected for later use. Therefore, usage of standard input is required by this volume of IEEE Std 1003.1-2001.

FUTURE DIRECTIONS

None.

SEE ALSO

Shell Command Language , the Base Definitions volume of IEEE Std 1003.1-2001, Chapter 11, General Terminal Interface, *<termios.h>*

COPYRIGHT

Portions of this text are reprinted and reproduced in electronic form from IEEE Std 1003.1, 2003 Edition, Standard for Information Technology -- Portable Operating System Interface (POSIX), The Open Group Base Specifications Issue 6, Copyright (C) 2001-2003 by the Institute of Electrical and Electronics Engineers, Inc and The Open Group. In the event of any discrepancy between this version and the original IEEE and The Open Group Standard, the original IEEE and The Open Group Standard is the referee document. The original Standard can be obtained online at <http://www.open-group.org/unix/online.html> .

NAME

tail - copy the last part of a file

SYNOPSIS

tail [-f] [-c *number*] [-n *number*][*file*]

DESCRIPTION

The *tail* utility shall copy its input file to the standard output beginning at a designated place.

Copying shall begin at the point in the file indicated by the **-c** *number* or **-n** *number* options. The option-argument *number* shall be counted in units of lines or bytes, according to the options **-n** and **-c**. Both line and byte counts start from 1.

Tails relative to the end of the file may be saved in an internal buffer, and thus may be limited in length. Such a buffer, if any, shall be no smaller than {LINE_MAX}*10 bytes.

OPTIONS

The *tail* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section 12.2, Utility Syntax Guidelines.

The following options shall be supported:

-c *number*

The application shall ensure that the *number* option-argument is a decimal integer whose sign affects the location in the file, measured in bytes, to begin the copying:

Sign	Copying Starts
+	Relative to the beginning of the file.
-	Relative to the end of the file.
<i>none</i>	Relative to the end of the file.

The origin for counting shall be 1; that is, **-c** +1 represents the first byte of the file, **-c** -1 the last.

-f If the input file is a regular file or if the *file* operand specifies a FIFO, do not terminate after the last line of the input file has been copied, but read and copy further bytes from the input file when they become available. If no *file* operand is specified and standard input is a pipe, the **-f** option shall be ignored. If the input file is not a FIFO, pipe, or regular file, it is unspecified whether or not the **-f** option shall be ignored.

-n *number*

This option shall be equivalent to **-c** *number*, except the starting location in the file shall be measured in lines instead of bytes. The origin for counting shall be 1; that is, **-n** +1 represents the first line of the file, **-n** -1 the last.

If neither **-c** nor **-n** is specified, **-n** 10 shall be assumed.

OPERANDS

The following operand shall be supported:

file A pathname of an input file. If no *file* operands are specified, the standard input shall be used.

STDIN

The standard input shall be used only if no *file* operands are specified. See the INPUT FILES section.

INPUT FILES

If the **-c** option is specified, the input file can contain arbitrary data; otherwise, the input file shall be a text file.

ENVIRONMENT VARIABLES

The following environment variables shall affect the execution of *tail*:

LANG Provide a default value for the internationalization variables that are unset or null. (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2, Internationalization Variables for the precedence of internationalization variables used to determine the values of locale categories.)

LC_ALL

If set to a non-empty string value, override the values of all the other internationalization variables.

LC_CTYPE

Determine the locale for the interpretation of sequences of bytes of text data as characters (for example, single-byte as opposed to multi-byte characters in arguments and input files).

LC_MESSAGES

Determine the locale that should be used to affect the format and contents of diagnostic messages written to standard error.

NLSPATH

Determine the location of message catalogs for the processing of *LC_MESSAGES*.

ASYNCHRONOUS EVENTS

Default.

STDOUT

The designated portion of the input file shall be written to standard output.

STDERR

The standard error shall be used only for diagnostic messages.

OUTPUT FILES

None.

EXTENDED DESCRIPTION

None.

EXIT STATUS

The following exit values shall be returned:

- 0 Successful completion.
- >0 An error occurred.

CONSEQUENCES OF ERRORS

Default.

The following sections are informative.

APPLICATION USAGE

The **-c** option should be used with caution when the input is a text file containing multi-byte characters; it may produce output that does not start on a character boundary.

Although the input file to *tail* can be any type, the results might not be what would be expected on some character special device files or on file types not described by the System Interfaces volume of IEEE Std 1003.1-2001. Since this volume of IEEE Std 1003.1-2001 does not specify the block size used when doing input, *tail* need not read all of the data from devices that only perform block transfers.

EXAMPLES

The **-f** option can be used to monitor the growth of a file that is being written by some other process. For example, the command:

```
tail -f fred
```

prints the last ten lines of the file **fred**, followed by any lines that are appended to **fred** between the time *tail* is initiated and killed. As another example, the command:

```
tail -f -c 15 fred
```

prints the last 15 bytes of the file **fred**, followed by any bytes that are appended to **fred** between the time *tail* is initiated and killed.

RATIONALE

This version of *tail* was created to allow conformance to the Utility Syntax Guidelines. The historical **-b** option was omitted because of the general non-portability of block-sized units of text. The **-c** option historically meant "characters", but this volume of IEEE Std 1003.1-2001 indicates that it means "bytes". This was selected to allow reasonable implementations when multi-byte characters are possible; it was not named **-b** to avoid confusion with the historical **-b**.

The origin of counting both lines and bytes is 1, matching all widespread historical implementations.

The restriction on the internal buffer is a compromise between the historical System V implementation of 4096 bytes and the BSD 32768 bytes.

The **-f** option has been implemented as a loop that sleeps for 1 second and copies any bytes that are available. This is sufficient, but if more efficient methods of determining when new data are available are developed, implementations are encouraged to use them.

Historical documentation indicates that *tail* ignores the **-f** option if the input file is a pipe (pipe and FIFO on systems that support FIFOs). On BSD-based systems, this has been true; on System V-based systems, this was true when input was taken from standard input, but it did not ignore the **-f** flag if a FIFO was named as the *file* operand. Since the **-f** option is not useful on pipes and all historical implementations ignore **-f** if no *file* operand is specified and standard input is a pipe, this volume of IEEE Std 1003.1-2001 requires this behavior. However, since the **-f** option is useful on a FIFO, this volume of IEEE Std 1003.1-2001 also requires that if standard input is a FIFO or a FIFO is named, the **-f** option shall not be ignored. Although historical behavior does not ignore the **-f** option for other file types, this is unspecified so that implementations are allowed to ignore the **-f** option if it is known that the file cannot be extended.

This was changed to the current form based on comments noting that **-c** was almost never used without specifying a number and that there was no need to specify **-l** if **-n number** was given.

FUTURE DIRECTIONS

None.

SEE ALSO

head

COPYRIGHT

Portions of this text are reprinted and reproduced in electronic form from IEEE Std 1003.1, 2003 Edition, Standard for Information Technology -- Portable Operating System Interface (POSIX), The Open Group Base Specifications Issue 6, Copyright (C) 2001-2003 by the Institute of Electrical and Electronics Engineers, Inc and The Open Group. In the event of any discrepancy between this version and the original IEEE and The Open Group Standard, the original IEEE and The Open Group Standard is the referee document. The original Standard can be obtained online at <http://www.open-group.org/unix/online.html>.

NAME

tee - duplicate standard input

SYNOPSIS

tee [-ai][*file...*]

DESCRIPTION

The *tee* utility shall copy standard input to standard output, making a copy in zero or more files. The *tee* utility shall not buffer output.

If the **-a** option is not specified, output files shall be written (see *File Read, Write, and Creation* .

OPTIONS

The *tee* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section 12.2, Utility Syntax Guidelines.

The following options shall be supported:

- a** Append the output to the files.
- i** Ignore the SIGINT signal.

OPERANDS

The following operands shall be supported:

- file* A pathname of an output file. Processing of at least 13 *file* operands shall be supported.

STDIN

The standard input can be of any type.

INPUT FILES

None.

ENVIRONMENT VARIABLES

The following environment variables shall affect the execution of *tee*:

LANG Provide a default value for the internationalization variables that are unset or null. (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2, Internationalization Variables for the precedence of internationalization variables used to determine the values of locale categories.)

LC_ALL

If set to a non-empty string value, override the values of all the other internationalization variables.

LC_CTYPE

Determine the locale for the interpretation of sequences of bytes of text data as characters (for example, single-byte as opposed to multi-byte characters in arguments).

LC_MESSAGES

Determine the locale that should be used to affect the format and contents of diagnostic messages written to standard error.

NLSPATH

Determine the location of message catalogs for the processing of *LC_MESSAGES* .

ASYNCHRONOUS EVENTS

Default, except that if the **-i** option was specified, SIGINT shall be ignored.

STDOUT

The standard output shall be a copy of the standard input.

STDERR

The standard error shall be used only for diagnostic messages.

OUTPUT FILES

If any *file* operands are specified, the standard input shall be copied to each named file.

EXTENDED DESCRIPTION

None.

EXIT STATUS

The following exit values shall be returned:

- 0 The standard input was successfully copied to all output files.
- >0 An error occurred.

CONSEQUENCES OF ERRORS

If a write to any successfully opened *file* operand fails, writes to other successfully opened *file* operands and standard output shall continue, but the exit status shall be non-zero. Otherwise, the default actions specified in *Utility Description Defaults* apply.

The following sections are informative.

APPLICATION USAGE

The *tee* utility is usually used in a pipeline, to make a copy of the output of some utility.

The *file* operand is technically optional, but *tee* is no more useful than *cat* when none is specified.

EXAMPLES

Save an unsorted intermediate form of the data in a pipeline:

```
... | tee unsorted | sort > sorted
```

RATIONALE

The buffering requirement means that *tee* is not allowed to use ISO C standard fully buffered or line-buffered writes. It does not mean that *tee* has to do 1-byte reads followed by 1-byte writes.

It should be noted that early versions of BSD ignore any invalid options and accept a single '-' as an alternative to -i. They also print a message if unable to open a file:

```
"tee: cannot access %s\n", <pathname>
```

Historical implementations ignore write errors. This is explicitly not permitted by this volume of IEEE Std 1003.1-2001.

Some historical implementations use O_APPEND when providing append mode; others use the *lseek()* function to seek to the end-of-file after opening the file without O_APPEND. This volume of IEEE Std 1003.1-2001 requires functionality equivalent to using O_APPEND; see *File Read, Write, and Creation*.

FUTURE DIRECTIONS

None.

SEE ALSO

Introduction, *cat*, the System Interfaces volume of IEEE Std 1003.1-2001, *lseek()*

COPYRIGHT

Portions of this text are reprinted and reproduced in electronic form from IEEE Std 1003.1, 2003 Edition, Standard for Information Technology -- Portable Operating System Interface (POSIX), The Open Group Base Specifications Issue 6, Copyright (C) 2001-2003 by the Institute of Electrical and Electronics Engineers, Inc and The Open Group. In the event of any discrepancy between this version and the original IEEE and The Open Group Standard, the original IEEE and The Open Group Standard is the referee document. The original Standard can be obtained online at <http://www.open-group.org/unix/online.html>.

NAME

test - evaluate expression

SYNOPSIS

test [*expression*]

[[*expression*]]

DESCRIPTION

The *test* utility shall evaluate the *expression* and indicate the result of the evaluation by its exit status. An exit status of zero indicates that the expression evaluated as true and an exit status of 1 indicates that the expression evaluated as false.

In the second form of the utility, which uses "[" rather than *test*, the application shall ensure that the square brackets are separate arguments.

OPTIONS

The *test* utility shall not recognize the "--" argument in the manner specified by guideline 10 in the Base Definitions volume of IEEE Std 1003.1-2001, Section 12.2, Utility Syntax Guidelines.

No options shall be supported.

OPERANDS

The application shall ensure that all operators and elements of primaries are presented as separate arguments to the *test* utility.

The following primaries can be used to construct *expression*:

- b file** True if *file* exists and is a block special file.
- c file** True if *file* exists and is a character special file.
- d file** True if *file* exists and is a directory.
- e file** True if *file* exists.
- f file** True if *file* exists and is a regular file.
- g file** True if *file* exists and its set-group-ID flag is set.
- h file** True if *file* exists and is a symbolic link.
- L file** True if *file* exists and is a symbolic link.
- n string**
True if the length of *string* is non-zero.
- p file** True if *file* is a FIFO.
- r file** True if *file* exists and is readable. True shall indicate that permission to read from *file* will be granted, as defined in *File Read, Write, and Creation*.
- S file** True if *file* exists and is a socket.
- s file** True if *file* exists and has a size greater than zero.
- t file_descriptor**
True if the file whose file descriptor number is *file_descriptor* is open and is associated with a terminal.
- u file** True if *file* exists and its set-user-ID flag is set.
- w file** True if *file* exists and is writable. True shall indicate that permission to write from *file* will be granted, as defined in *File Read, Write, and Creation*.
- x file** True if *file* exists and is executable. True shall indicate that permission to execute *file* will be granted, as defined in *File Read, Write, and Creation*. If *file* is a directory, true shall indicate that permission to search *file* will be granted.

-z *string*

True if the length of string *string* is zero.

string True if the string *string* is not the null string.

s1 = *s2*

True if the strings *s1* and *s2* are identical.

s1 != *s2*

True if the strings *s1* and *s2* are not identical.

n1 **-eq** *n2*

True if the integers *n1* and *n2* are algebraically equal.

n1 **-ne** *n2*

True if the integers *n1* and *n2* are not algebraically equal.

n1 **-gt** *n2*

True if the integer *n1* is algebraically greater than the integer *n2*.

n1 **-ge** *n2*

True if the integer *n1* is algebraically greater than or equal to the integer *n2*.

n1 **-lt** *n2*

True if the integer *n1* is algebraically less than the integer *n2*.

n1 **-le** *n2*

True if the integer *n1* is algebraically less than or equal to the integer *n2*.

expression1 **-a** *expression2*

True if both *expression1* and *expression2* are true. The **-a** binary primary is left associative. It has a higher precedence than **-o**.

expression1 **-o** *expression2*

True if either *expression1* or *expression2* is true. The **-o** binary primary is left associative.

With the exception of the **-h** *file* and **-L** *file* primaries, if a *file* argument is a symbolic link, *test* shall evaluate the expression by resolving the symbolic link and using the file referenced by the link.

These primaries can be combined with the following operators:

! *expression*

True if *expression* is false.

(*expression*)

True if *expression* is true. The parentheses can be used to alter the normal precedence and associativity.

The primaries with two elements of the form:

-primary_operator primary_operand

are known as *unary primaries*. The primaries with three elements in either of the two forms:

primary_operand -primary_operator primary_operand

primary_operand primary_operator primary_operand

are known as *binary primaries*. Additional implementation-defined operators and *primary_operators* may be provided by implementations. They shall be of the form **- operator** where the first character of *operator* is not a digit.

The algorithm for determining the precedence of the operators and the return value that shall be

generated is based on the number of arguments presented to *test*. (However, when using the "[...]" form, the right-bracket final argument shall not be counted in this algorithm.)

In the following list, \$1, \$2, \$3, and \$4 represent the arguments presented to *test*:

0 arguments:

Exit false (1).

1 argument:

Exit true (0) if \$1 is not null; otherwise, exit false.

2 arguments:

If \$1 is '!' , exit true if \$2 is null, false if \$2 is not null.

If \$1 is a unary primary, exit true if the unary test is true, false if the unary test is false.

Otherwise, produce unspecified results.

3 arguments:

If \$2 is a binary primary, perform the binary test of \$1 and \$3.

If \$1 is '!' , negate the two-argument test of \$2 and \$3.

If \$1 is '(' and \$3 is ')' , perform the unary test of \$2.

Otherwise, produce unspecified results.

4 arguments:

If \$1 is '!' , negate the three-argument test of \$2, \$3, and \$4.

If \$1 is '(' and \$4 is ')' , perform the two-argument test of \$2 and \$3.

Otherwise, the results are unspecified.

>4 arguments:

The results are unspecified.

On XSI-conformant systems, combinations of primaries and operators shall be evaluated using the precedence and associativity rules described previously. In addition, the string comparison binary primaries '=' and '!=' shall have a higher precedence than any unary primary.

STDIN

Not used.

INPUT FILES

None.

ENVIRONMENT VARIABLES

The following environment variables shall affect the execution of *test*:

LANG Provide a default value for the internationalization variables that are unset or null. (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2, Internationalization Variables for the precedence of internationalization variables used to determine the values of locale categories.)

LC_ALL

If set to a non-empty string value, override the values of all the other internationalization variables.

LC_CTYPE

Determine the locale for the interpretation of sequences of bytes of text data as characters (for example, single-byte as opposed to multi-byte characters in arguments).

LC_MESSAGES

Determine the locale that should be used to affect the format and contents of diagnostic messages written to standard error.

NLSPATH

Determine the location of message catalogs for the processing of **LC_MESSAGES**.

ASYNCHRONOUS EVENTS

Default.

STDOUT

Not used.

STDERR

The standard error shall be used only for diagnostic messages.

OUTPUT FILES

None.

EXTENDED DESCRIPTION

None.

EXIT STATUS

The following exit values shall be returned:

- 0 *expression* evaluated to true.
- 1 *expression* evaluated to false or *expression* was missing.
- >1 An error occurred.

CONSEQUENCES OF ERRORS

Default.

The following sections are informative.

APPLICATION USAGE

Scripts should be careful when dealing with user-supplied input that could be confused with primaries and operators. Unless the application writer knows all the cases that produce input to the script, invocations like:

```
test "$1" -a "$2"
```

should be written as:

```
test "$1" && test "$2"
```

to avoid problems if a user supplied values such as \$1 set to '!' and \$2 set to the null string. That is, in cases where maximal portability is of concern, replace:

```
test expr1 -a expr2
```

with:

```
test expr1 && test expr2
```

and replace:

```
test expr1 -o expr2
```

with:

```
test expr1 || test expr2
```

but note that, in *test*, **-a** has higher precedence than **-o** while **"&&"** and **"||"** have equal precedence in the shell.

Parentheses or braces can be used in the shell command language to effect grouping.

Parentheses must be escaped when using *sh*; for example:

test(P)

test(P)

test \(expr1 -a expr2 \) -o expr3

This command is not always portable outside XSI-conformant systems. The following form can be used instead:

(test expr1 && test expr2) || test expr3

The two commands:

test "\$1"
test ! "\$1"

could not be used reliably on some historical systems. Unexpected results would occur if such a *string* expression were used and \$1 expanded to '!', '(', or a known unary primary. Better constructs are:

test -n "\$1"
test -z "\$1"

respectively.

Historical systems have also been unreliable given the common construct:

test "\$response" = "expected string"

One of the following is a more reliable form:

test "X\$response" = "Xexpected string"
test "expected string" = "\$response"

Note that the second form assumes that *expected string* could not be confused with any unary primary. If *expected string* starts with '-', '(', '!', or even '=', the first form should be used instead. Using the preceding rules without the XSI marked extensions, any of the three comparison forms is reliable, given any input. (However, note that the strings are quoted in all cases.)

Because the string comparison binary primaries, '=' and '!=', have a higher precedence than any unary primary in the greater than 4 argument case, unexpected results can occur if arguments are not properly prepared. For example, in:

test -d \$1 -o -d \$2

If \$1 evaluates to a possible directory name of '=', the first three arguments are considered a string comparison, which shall cause a syntax error when the second -d is encountered. One of the following forms prevents this; the second is preferred:

test \(-d "\$1" \) -o \(-d "\$2" \)
test -d "\$1" || test -d "\$2"

Also in the greater than 4 argument case:

test "\$1" = "bat" -a "\$2" = "ball"

syntax errors occur if \$1 evaluates to '(' or '!'. One of the following forms prevents this; the third is preferred:

test "X\$1" = "Xbat" -a "X\$2" = "Xball"
test "\$1" = "bat" && test "\$2" = "ball"
test "X\$1" = "Xbat" && test "X\$2" = "Xball"

EXAMPLES

Exit if there are not two or three arguments (two variations):

```
if [ $# -ne 2 -a $# -ne 3 ]; then exit 1; fi
if [ $# -lt 2 -o $# -gt 3 ]; then exit 1; fi
```

Perform a *mkdir* if a directory does not exist:

```
test ! -d tempdir && mkdir tempdir
```

Wait for a file to become non-readable:

```
while test -r thefile
do
    sleep 30
done
echo "'thefile' is no longer readable'
```

Perform a command if the argument is one of three strings (two variations):

```
if [ "$1" = "pear" ] || [ "$1" = "grape" ] || [ "$1" = "apple" ]
then
    command
fi
```

```
case "$1" in
    pear|grape|apple) command ;;
esac
```

RATIONALE

The KornShell-derived conditional command (double bracket [[]]) was removed from the shell command language description in an early proposal. Objections were raised that the real problem is misuse of the *test* command ([], and putting it into the shell is the wrong way to fix the problem. Instead, proper documentation and a new shell reserved word (!) are sufficient.

Tests that require multiple *test* operations can be done at the shell level using individual invocations of the *test* command and shell logicals, rather than using the error-prone **-o** flag of *test*.

XSI-conformant systems support more than four arguments.

XSI-conformant systems support the combining of primaries with the following constructs:

expression1 **-a** *expression2*

True if both *expression1* and *expression2* are true.

expression1 **-o** *expression2*

True if at least one of *expression1* and *expression2* are true.

(*expression*)

True if *expression* is true.

In evaluating these more complex combined expressions, the following precedence rules are used:

The unary primaries have higher precedence than the algebraic binary primaries.

The unary primaries have lower precedence than the string binary primaries.

The unary and binary primaries have higher precedence than the unary *string* primary.

The **!** operator has higher precedence than the **-a** operator, and the **-a** operator has higher precedence than the **-o** operator.

The **-a** and **-o** operators are left associative.

The parentheses can be used to alter the normal precedence and associativity.

The BSD and System V versions of **-f** are not the same. The BSD definition was:

-f file True if *file* exists and is not a directory.

The SVID version (true if the file exists and is a regular file) was chosen for this volume of IEEE Std 1003.1-2001 because its use is consistent with the **-b**, **-c**, **-d**, and **-p** operands (*file* exists and is a specific file type).

The **-e** primary, possessing similar functionality to that provided by the C shell, was added because it provides the only way for a shell script to find out if a file exists without trying to open the file. Since implementations are allowed to add additional file types, a portable script cannot use:

test -b foo -o -c foo -o -d foo -o -f foo -o -p foo

to find out if **foo** is an existing file. On historical BSD systems, the existence of a file could be determined by:

test -f foo -o -d foo

but there was no easy way to determine that an existing file was a regular file. An early proposal used the KornShell **-a** primary (with the same meaning), but this was changed to **-e** because there were concerns about the high probability of humans confusing the **-a** primary with the **-a** binary operator.

The following options were not included in this volume of IEEE Std 1003.1-2001, although they are provided by some implementations. These operands should not be used by new implementations for other purposes:

-k file True if *file* exists and its sticky bit is set.

-C file True if *file* is a contiguous file.

-V file True if *file* is a version file.

The following option was not included because it was undocumented in most implementations, has been removed from some implementations (including System V), and the functionality is provided by the shell (see *Parameter Expansion*).

-l string

The length of the string *string*.

The **-b**, **-c**, **-g**, **-p**, **-u**, and **-x** operands are derived from the SVID; historical BSD does not provide them. The **-k** operand is derived from System V; historical BSD does not provide it.

On historical BSD systems, *test -w directory* always returned false because *test* tried to open the directory for writing, which always fails.

Some additional primaries newly invented or from the KornShell appeared in an early proposal as part of the conditional command ([]): *s1 > s2*, *s1 < s2*, *str = pattern*, *str != pattern*, *f1 -nt f2*, *f1 -ot f2*, and *f1 -ef f2*. They were not carried forward into the *test* utility when the conditional command was removed from the shell because they have not been included in the *test* utility built into historical implementations of the *sh* utility.

The **-t file_descriptor** primary is shown with a mandatory argument because the grammar is ambiguous if it can be omitted. Historical implementations have allowed it to be omitted, providing a default of 1.

FUTURE DIRECTIONS

None.

SEE ALSO

File Read, Write, and Creation , find

COPYRIGHT

Portions of this text are reprinted and reproduced in electronic form from IEEE Std 1003.1, 2003 Edition, Standard for Information Technology -- Portable Operating System Interface (POSIX), The Open Group Base Specifications Issue 6, Copyright (C) 2001-2003 by the Institute of Electrical and Electronics Engineers, Inc and The Open Group. In the event of any discrepancy between this version and the original IEEE and The Open Group Standard, the original IEEE and The Open Group Standard is the referee document. The original Standard can be obtained online at <http://www.open-group.org/unix/online.html> .

NAME

touch - change file access and modification times

SYNOPSIS

touch [-acm][-r *ref_file*] -t *time*] *file*...

DESCRIPTION

The *touch* utility shall change the modification times, access times, or both of files. The modification time shall be equivalent to the value of the *st_mtime* member of the **stat** structure for a file, as described in the System Interfaces volume of IEEE Std 1003.1-2001; the access time shall be equivalent to the value of *st_atime*.

The time used can be specified by the -t *time* option-argument, the corresponding time fields of the file referenced by the -r *ref_file* option-argument, or the *date_time* operand, as specified in the following sections. If none of these are specified, *touch* shall use the current time (the value returned by the equivalent of the *time()* function defined in the System Interfaces volume of IEEE Std 1003.1-2001).

For each *file* operand, *touch* shall perform actions equivalent to the following functions defined in the System Interfaces volume of IEEE Std 1003.1-2001:

If *file* does not exist, a *creat()* function call is made with the *file* operand used as the *path* argument and the value of the bitwise-inclusive OR of S_IRUSR, S_IWUSR, S_IRGRP, S_IWGRP, S_IROTH, and S_IWOTH used as the *mode* argument.

The *utime()* function is called with the following arguments: <ol type="a">

The *file* operand is used as the *path* argument.

The **utimbuf** structure members *actime* and *modtime* are determined as described in the OPTIONS section.

OPTIONS

The *touch* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section 12.2, Utility Syntax Guidelines.

The following options shall be supported:

- a Change the access time of *file*. Do not change the modification time unless -m is also specified.
- c Do not create a specified *file* if it does not exist. Do not write any diagnostic messages concerning this condition.
- m Change the modification time of *file*. Do not change the access time unless -a is also specified.
- r *ref_file*
Use the corresponding time of the file named by the pathname *ref_file* instead of the current time.
- t *time*
Use the specified *time* instead of the current time. The option-argument shall be a decimal number of the form:

[[CC]YY]MMDDhhmm[.SS]

where each two digits represents the following:

- MM The month of the year [01,12].
- DD The day of the month [01,31].
- hh The hour of the day [00,23].
- mm The minute of the hour [00,59].
- CC The first two digits of the year (the century).
- YY The second two digits of the year.

SS The second of the minute [00,60].

Both *CC* and *YY* shall be optional. If neither is given, the current year shall be assumed. If *YY* is specified, but *CC* is not, *CC* shall be derived as follows:

If <i>YY</i> is:	<i>CC</i> becomes:
[69,99]	19
[00,68]	20

Note: It is expected that in a future version of IEEE Std 1003.1-2001 the default century inferred from a 2-digit year will change. (This would apply to all commands accepting a 2-digit year as input.)

The resulting time shall be affected by the value of the *TZ* environment variable. If the resulting time value precedes the Epoch, *touch* shall exit immediately with an error status. The range of valid times past the Epoch is implementation-defined, but it shall extend to at least the time 0 hours, 0 minutes, 0 seconds, January 1, 2038, Coordinated Universal Time. Some implementations may not be able to represent dates beyond January 18, 2038, because they use **signed int** as a time holder.

The range for *SS* is [00,60] rather than [00,59] because of leap seconds. If *SS* is 60, and the resulting time, as affected by the *TZ* environment variable, does not refer to a leap second, the resulting time shall be one second after a time where *SS* is 59. If *SS* is not given a value, it is assumed to be zero.

If neither the **-a** nor **-m** options were specified, *touch* shall behave as if both the **-a** and **-m** options were specified.

OPERANDS

The following operands shall be supported:

file A pathname of a file whose times shall be modified.

STDIN

Not used.

INPUT FILES

None.

ENVIRONMENT VARIABLES

The following environment variables shall affect the execution of *touch*:

LANG Provide a default value for the internationalization variables that are unset or null. (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2, Internationalization Variables for the precedence of internationalization variables used to determine the values of locale categories.)

LC_ALL

If set to a non-empty string value, override the values of all the other internationalization variables.

LC_CTYPE

Determine the locale for the interpretation of sequences of bytes of text data as characters (for example, single-byte as opposed to multi-byte characters in arguments).

LC_MESSAGES

Determine the locale that should be used to affect the format and contents of diagnostic messages written to standard error.

NLSPATH

Determine the location of message catalogs for the processing of *LC_MESSAGES*.

TZ

Determine the timezone to be used for interpreting the *time* option-argument. If *TZ* is unset or null, an unspecified default timezone shall be used.

ASYNCHRONOUS EVENTS

Default.

STDOUT

Not used.

STDERR

The standard error shall be used only for diagnostic messages.

OUTPUT FILES

None.

EXTENDED DESCRIPTION

None.

EXIT STATUS

The following exit values shall be returned:

- 0 The utility executed successfully and all requested changes were made.
- >0 An error occurred.

CONSEQUENCES OF ERRORS

Default.

The following sections are informative.

APPLICATION USAGE

The interpretation of time is taken to be *seconds since the Epoch* (see the Base Definitions volume of IEEE Std 1003.1-2001, Section 4.14, Seconds Since the Epoch). It should be noted that implementations conforming to the System Interfaces volume of IEEE Std 1003.1-2001 do not take leap seconds into account when computing seconds since the Epoch. When *SS=60* is used, the resulting time always refers to 1 plus *seconds since the Epoch* for a time when *SS=59*.

Although the **-t time** option-argument specifies values in 1969, the access time and modification time fields are defined in terms of seconds since the Epoch (00:00:00 on 1 January 1970 UTC). Therefore, depending on the value of *TZ* when *touch* is run, there is never more than a few valid hours in 1969 and there need not be any valid times in 1969.

One ambiguous situation occurs if **-t time** is not specified, **-r ref_file** is not specified, and the first operand is an eight or ten-digit decimal number. A portable script can avoid this problem by using:

```
touch -- file
```

or:

```
touch ./file
```

in this case.

EXAMPLES

None.

RATIONALE

The functionality of *touch* is described almost entirely through references to functions in the System Interfaces volume of IEEE Std 1003.1-2001. In this way, there is no duplication of effort required for describing such side effects as the relationship of user IDs to the user database, permissions, and so on.

There are some significant differences between the *touch* utility in this volume of IEEE Std 1003.1-2001 and those in System V and BSD systems. They are upwards-compatible for historical applications from both implementations:

In System V, an ambiguity exists when a pathname that is a decimal number leads the operands; it is treated as a time value. In BSD, no *time* value is allowed; files may only be *touched* to the current time. The **-t time** construct solves these problems for future conforming applications (note that the **-t** option is not historical practice).

The inclusion of the century digits, *CC*, is also new. Note that a ten-digit *time* value is treated as if *YY*, and not *CC*, were specified. The caveat about the range of dates following the Epoch was included as recognition that some implementations are not able to represent dates beyond 18 January 2038 because they use **signed int** as a time holder.

The **-r** option was added because several comments requested this capability. This option was named **-f** in an early proposal, but was changed because the **-f** option is used in the BSD version of *touch* with a different meaning.

At least one historical implementation of *touch* incremented the exit code if **-c** was specified and the file did not exist. This volume of IEEE Std 1003.1-2001 requires exit status zero if no errors occur.

FUTURE DIRECTIONS

Applications should use the **-r** or **-t** options.

SEE ALSO

date, the System Interfaces volume of IEEE Std 1003.1-2001, *creat()*, *time()*, *utime()*, the Base Definitions volume of IEEE Std 1003.1-2001, *<sys/stat.h>*

COPYRIGHT

Portions of this text are reprinted and reproduced in electronic form from IEEE Std 1003.1, 2003 Edition, Standard for Information Technology -- Portable Operating System Interface (POSIX), The Open Group Base Specifications Issue 6, Copyright (C) 2001-2003 by the Institute of Electrical and Electronics Engineers, Inc and The Open Group. In the event of any discrepancy between this version and the original IEEE and The Open Group Standard, the original IEEE and The Open Group Standard is the referee document. The original Standard can be obtained online at <http://www.open-group.org/unix/online.html>.

NAME

tr - translate characters

SYNOPSIS

tr [-c | -C][-s] *string1* *string2*

tr -s [-c | -C] *string1*

tr -d [-c | -C] *string1*

tr -ds [-c | -C] *string1* *string2*

DESCRIPTION

The *tr* utility shall copy the standard input to the standard output with substitution or deletion of selected characters. The options specified and the *string1* and *string2* operands shall control translations that occur while copying characters and single-character collating elements.

OPTIONS

The *tr* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section 12.2, Utility Syntax Guidelines.

The following options shall be supported:

- c** Complement the set of values specified by *string1*. See the EXTENDED DESCRIPTION section.
- C** Complement the set of characters specified by *string1*. See the EXTENDED DESCRIPTION section.
- d** Delete all occurrences of input characters that are specified by *string1*.
- s** Replace instances of repeated characters with a single character, as described in the EXTENDED DESCRIPTION section.

OPERANDS

The following operands shall be supported:

string1, *string2*

Translation control strings. Each string shall represent a set of characters to be converted into an array of characters used for the translation. For a detailed description of how the strings are interpreted, see the EXTENDED DESCRIPTION section.

STDIN

The standard input can be any type of file.

INPUT FILES

None.

ENVIRONMENT VARIABLES

The following environment variables shall affect the execution of *tr*:

LANG Provide a default value for the internationalization variables that are unset or null. (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2, Internationalization Variables for the precedence of internationalization variables used to determine the values of locale categories.)

LC_ALL

If set to a non-empty string value, override the values of all the other internationalization variables.

LC_COLLATE

Determine the locale for the behavior of range expressions and equivalence classes.

LC_CTYPE

Determine the locale for the interpretation of sequences of bytes of text data as characters (for example, single-byte as opposed to multi-byte characters in arguments) and the behavior of character classes.

LC_MESSAGES

Determine the locale that should be used to affect the format and contents of diagnostic messages written to standard error.

NLSPATH

Determine the location of message catalogs for the processing of *LC_MESSAGES*.

ASYNCHRONOUS EVENTS

Default.

STDOUT

The *tr* output shall be identical to the input, with the exception of the specified transformations.

STDERR

The standard error shall be used only for diagnostic messages.

OUTPUT FILES

None.

EXTENDED DESCRIPTION

The operands *string1* and *string2* (if specified) define two arrays of characters. The constructs in the following list can be used to specify characters or single-character collating elements. If any of the constructs result in multi-character collating elements, *tr* shall exclude, without a diagnostic, those multi-character elements from the resulting array.

character

Any character not described by one of the conventions below shall represent itself.

\octal Octal sequences can be used to represent characters with specific coded values. An octal sequence shall consist of a backslash followed by the longest sequence of one, two, or three-octal-digit characters (01234567). The sequence shall cause the value whose encoding is represented by the one, two, or three-digit octal integer to be placed into the array. If the size of a byte on the system is greater than nine bits, the valid escape sequence used to represent a byte is implementation-defined. Multi-byte characters require multiple, concatenated escape sequences of this type, including the leading '**' for each byte.

\character

The backslash-escape sequences in the Base Definitions volume of IEEE Std 1003.1-2001, Table 5-1, Escape Sequences and Associated Actions ('**' , '*\a*' , '*\b*' , '*\f*' , '*\n*' , '*\r*' , '*\t*' , '*\v*') shall be supported. The results of using any other character, other than an octal digit, following the backslash are unspecified.

c-c In the POSIX locale, this construct shall represent the range of collating elements between the range endpoints (as long as neither endpoint is an octal sequence of the form *\octal*), inclusive, as defined by the collation sequence. The characters or collating elements in the range shall be placed in the array in ascending collation sequence. If the second endpoint precedes the starting endpoint in the collation sequence, it is unspecified whether the range of collating elements is empty, or this construct is treated as invalid. In locales other than the POSIX locale, this construct has unspecified behavior.

If either or both of the range endpoints are octal sequences of the form *\octal*, this shall represent the range of specific coded values between the two range endpoints, inclusive.

:class: Represents all characters belonging to the defined character class, as defined by the current setting of the *LC_CTYPE* locale category. The following character class names shall be accepted when specified in *string1*:

alnum	blank	digit	lower	punct	upper
alpha	cntrl	graph	print	space	xdigit

In addition, character class expressions of the form *[: name:]* shall be recognized in those locales where the *name* keyword has been given a **charclass** definition in the *LC_CTYPE* category.

When both the **-d** and **-s** options are specified, any of the character class names shall be accepted in *string2*. Otherwise, only character class names **lower** or **upper** are valid in *string2* and then only if the corresponding character class (**upper** and **lower**, respectively) is specified in the same relative position in *string1*. Such a specification shall be interpreted as a request for case conversion. When `[: lower:]` appears in *string1* and `[: upper:]` appears in *string2*, the arrays shall contain the characters from the **toupper** mapping in the *LC_CTYPE* category of the current locale. When `[: upper:]` appears in *string1* and `[: lower:]` appears in *string2*, the arrays shall contain the characters from the **tolower** mapping in the *LC_CTYPE* category of the current locale. The first character from each mapping pair shall be in the array for *string1* and the second character from each mapping pair shall be in the array for *string2* in the same relative position.

Except for case conversion, the characters specified by a character class expression shall be placed in the array in an unspecified order.

If the name specified for *class* does not define a valid character class in the current locale, the behavior is undefined.

`=equiv=`

Represents all characters or collating elements belonging to the same equivalence class as *equiv*, as defined by the current setting of the *LC_COLLATE* locale category. An equivalence class expression shall be allowed only in *string1*, or in *string2* when it is being used by the combined **-d** and **-s** options. The characters belonging to the equivalence class shall be placed in the array in an unspecified order.

`x*n`

Represents *n* repeated occurrences of the character *x*. Because this expression is used to map multiple characters to one, it is only valid when it occurs in *string2*. If *n* is omitted or is zero, it shall be interpreted as large enough to extend the *string2*-based sequence to the length of the *string1*-based sequence. If *n* has a leading zero, it shall be interpreted as an octal value. Otherwise, it shall be interpreted as a decimal value.

When the **-d** option is not specified:

Each input character found in the array specified by *string1* shall be replaced by the character in the same relative position in the array specified by *string2*. When the array specified by *string2* is shorter than the one specified by *string1*, the results are unspecified.

If the **-C** option is specified, the complements of the characters specified by *string1* (the set of all characters in the current character set, as defined by the current setting of *LC_CTYPE*, except for those actually specified in the *string1* operand) shall be placed in the array in ascending collation sequence, as defined by the current setting of *LC_COLLATE*.

If the **-c** option is specified, the complement of the values specified by *string1* shall be placed in the array in ascending order by binary value.

Because the order in which characters specified by character class expressions or equivalence class expressions is undefined, such expressions should only be used if the intent is to map several characters into one. An exception is case conversion, as described previously.

When the **-d** option is specified:

Input characters found in the array specified by *string1* shall be deleted.

When the **-C** option is specified with **-d**, all characters except those specified by *string1* shall be deleted. The contents of *string2* are ignored, unless the **-s** option is also specified.

When the **-c** option is specified with **-d**, all values except those specified by *string1* shall be deleted. The contents of *string2* shall be ignored, unless the **-s** option is also specified.

The same string cannot be used for both the **-d** and the **-s** option; when both options are specified, both *string1* (used for deletion) and *string2* (used for squeezing) shall be required.

When the **-s** option is specified, after any deletions or translations have taken place, repeated sequences of the same character shall be replaced by one occurrence of the same character, if the character is found in the array specified by the last operand. If the last operand contains a character class, such as the following example:

tr -s '[:space:]'

the last operand's array shall contain all of the characters in that character class. However, in a case conversion, as described previously, such as:

tr -s '[:upper:]' '[:lower:]'

the last operand's array shall contain only those characters defined as the second characters in each of the **toupper** or **tolower** character pairs, as appropriate.

An empty string used for *string1* or *string2* produces undefined results.

EXIT STATUS

The following exit values shall be returned:

- 0 All input was processed successfully.
- >0 An error occurred.

CONSEQUENCES OF ERRORS

Default.

The following sections are informative.

APPLICATION USAGE

If necessary, *string1* and *string2* can be quoted to avoid pattern matching by the shell.

If an ordinary digit (representing itself) is to follow an octal sequence, the octal sequence must use the full three digits to avoid ambiguity.

When *string2* is shorter than *string1*, a difference results between historical System V and BSD systems. A BSD system pads *string2* with the last character found in *string2*. Thus, it is possible to do the following:

tr 0123456789 d

which would translate all digits to the letter 'd'. Since this area is specifically unspecified in this volume of IEEE Std 1003.1-2001, both the BSD and System V behaviors are allowed, but a conforming application cannot rely on the BSD behavior. It would have to code the example in the following way:

tr 0123456789 '[d*]'

It should be noted that, despite similarities in appearance, the string operands used by *tr* are not regular expressions.

Unlike some historical implementations, this definition of the *tr* utility correctly processes NUL characters in its input stream. NUL characters can be stripped by using:

tr -d '\000'

EXAMPLES

The following example creates a list of all words in **file1** one per line in **file2**, where a word is taken to be a maximal string of letters.

tr -cs "[:alpha:]" "[:\n*]" <file1 >file2

The next example translates all lowercase characters in **file1** to uppercase and writes the results to standard output.

tr "[:lower:]" "[:upper:]" <file1

This example uses an equivalence class to identify accented variants of the base character 'e' in **file1**,

which are stripped of diacritical marks and written to **file2**.

```
tr "[=e]" e <file1 >file2
```

RATIONALE

In some early proposals, an explicit option **-n** was added to disable the historical behavior of stripping NUL characters from the input. It was considered that automatically stripping NUL characters from the input was not correct functionality. However, the removal of **-n** in a later proposal does not remove the requirement that *tr* correctly process NUL characters in its input stream. NUL characters can be stripped by using *tr -d '\000'*.

Historical implementations of *tr* differ widely in syntax and behavior. For example, the BSD version has not needed the bracket characters for the repetition sequence. The *tr* utility syntax is based more closely on the System V and XPG3 model while attempting to accommodate historical BSD implementations. In the case of the short *string2* padding, the decision was to unspecify the behavior and preserve System V and XPG3 scripts, which might find difficulty with the BSD method. The assumption was made that BSD users of *tr* have to make accommodations to meet the syntax defined here. Since it is possible to use the repetition sequence to duplicate the desired behavior, whereas there is no simple way to achieve the System V method, this was the correct, if not desirable, approach.

The use of octal values to specify control characters, while having historical precedents, is not portable. The introduction of escape sequences for control characters should provide the necessary portability. It is recognized that this may cause some historical scripts to break.

An early proposal included support for multi-character collating elements. It was pointed out that, while *tr* does employ some syntactical elements from REs, the aim of *tr* is quite different; ranges, for example, do not have a similar meaning ("any of the chars in the range matches", *versus* "translate each character in the range to the output counterpart"). As a result, the previously included support for multi-character collating elements has been removed. What remains are ranges in current collation order (to support, for example, accented characters), character classes, and equivalence classes.

In XPG3 the `[: class:]` and `[= equiv=]` conventions are shown with double brackets, as in RE syntax. However, *tr* does not implement RE principles; it just borrows part of the syntax. Consequently, `[: class:]` and `[= equiv=]` should be regarded as syntactical elements on a par with `[x* n]`, which is not an RE bracket expression.

The standard developers will consider changes to *tr* that allow it to translate characters between different character encodings, or they will consider providing a new utility to accomplish this.

On historical System V systems, a range expression requires enclosing square-brackets, such as:

```
tr '[a-z]' '[A-Z]'
```

However, BSD-based systems did not require the brackets, and this convention is used here to avoid breaking large numbers of BSD scripts:

```
tr a-z A-Z
```

The preceding System V script will continue to work because the brackets, treated as regular characters, are translated to themselves. However, any System V script that relied on **"a-z"** representing the three characters **'a'**, **'-'**, and **'z'** have to be rewritten as **"az-"**.

The ISO POSIX-2:1993 standard had a **-c** option that behaved similarly to the **-C** option, but did not supply functionality equivalent to the **-c** option specified in IEEE Std 1003.1-2001. This meant that historical practice of being able to specify *tr -d\200-\377* (which would delete all bytes with the top bit set) would have no effect because, in the C locale, bytes with the values octal 200 to octal 377 are not characters.

The earlier version also said that octal sequences referred to collating elements and could be placed adjacent to each other to specify multi-byte characters. However, it was noted that this caused ambiguities because *tr* would not be able to tell whether adjacent octal sequences were intending to specify multi-byte characters or multiple single byte characters. IEEE Std 1003.1-2001 specifies that octal

sequences always refer to single byte binary values.

FUTURE DIRECTIONS

None.

SEE ALSO

sed

COPYRIGHT

Portions of this text are reprinted and reproduced in electronic form from IEEE Std 1003.1, 2003 Edition, Standard for Information Technology -- Portable Operating System Interface (POSIX), The Open Group Base Specifications Issue 6, Copyright (C) 2001-2003 by the Institute of Electrical and Electronics Engineers, Inc and The Open Group. In the event of any discrepancy between this version and the original IEEE and The Open Group Standard, the original IEEE and The Open Group Standard is the referee document. The original Standard can be obtained online at <http://www.open-group.org/unix/online.html> .

NAME

true - return true value

SYNOPSIS

true

DESCRIPTION

The *true* utility shall return with exit code zero.

OPTIONS

None.

OPERANDS

None.

STDIN

Not used.

INPUT FILES

None.

ENVIRONMENT VARIABLES

None.

ASYNCHRONOUS EVENTS

Default.

STDOUT

Not used.

STDERR

Not used.

OUTPUT FILES

None.

EXTENDED DESCRIPTION

None.

EXIT STATUS

Zero.

CONSEQUENCES OF ERRORS

None.

The following sections are informative.

APPLICATION USAGE

This utility is typically used in shell scripts, as shown in the EXAMPLES section. The special built-in utility `:` is sometimes more efficient than *true*.

EXAMPLES

This command is executed forever:

```
while true
do
    command
done
```

RATIONALE

The *true* utility has been retained in this volume of IEEE Std 1003.1-2001, even though the shell special built-in `:` provides similar functionality, because *true* is widely used in historical scripts and is less cryptic to novice script readers.

FUTURE DIRECTIONS

None.

true(P)

true(P)

SEE ALSO

false , *Shell Commands*

COPYRIGHT

Portions of this text are reprinted and reproduced in electronic form from IEEE Std 1003.1, 2003 Edition, Standard for Information Technology -- Portable Operating System Interface (POSIX), The Open Group Base Specifications Issue 6, Copyright (C) 2001-2003 by the Institute of Electrical and Electronics Engineers, Inc and The Open Group. In the event of any discrepancy between this version and the original IEEE and The Open Group Standard, the original IEEE and The Open Group Standard is the referee document. The original Standard can be obtained online at <http://www.open-group.org/unix/online.html> .

NAME

tsort - topological sort

SYNOPSIS

tsort [*file*]

DESCRIPTION

The *tsort* utility shall write to standard output a totally ordered list of items consistent with a partial ordering of items contained in the input.

The application shall ensure that the input consists of pairs of items (non-empty strings) separated by <blank>s. Pairs of different items indicate ordering. Pairs of identical items indicate presence, but not ordering.

OPTIONS

None.

OPERANDS

The following operand shall be supported:

file A pathname of a text file to order. If no *file* operand is given, the standard input shall be used.

STDIN

The standard input shall be a text file that is used if no *file* operand is given.

INPUT FILES

The input file named by the *file* operand is a text file.

ENVIRONMENT VARIABLES

The following environment variables shall affect the execution of *tsort*:

LANG Provide a default value for the internationalization variables that are unset or null. (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2, Internationalization Variables for the precedence of internationalization variables used to determine the values of locale categories.)

LC_ALL If set to a non-empty string value, override the values of all the other internationalization variables.

LC_CTYPE Determine the locale for the interpretation of sequences of bytes of text data as characters (for example, single-byte as opposed to multi-byte characters in arguments and input files).

LC_MESSAGES Determine the locale that should be used to affect the format and contents of diagnostic messages written to standard error.

NLSPATH Determine the location of message catalogs for the processing of *LC_MESSAGES*.

ASYNCHRONOUS EVENTS

Default.

STDOUT

The standard output shall be a text file consisting of the order list produced from the partially ordered input.

STDERR

The standard error shall be used only for diagnostic messages.

OUTPUT FILES

None.

EXTENDED DESCRIPTION

None.

EXIT STATUS

The following exit values shall be returned:

- 0 Successful completion.
- >0 An error occurred.

CONSEQUENCES OF ERRORS

Default.

The following sections are informative.

APPLICATION USAGE

The *LC_COLLATE* variable need not affect the actions of *tsort*. The output ordering is not lexicographic, but depends on the pairs of items given as input.

EXAMPLES

The command:

```
tsort <<EOF
a b c c d e
g g
f g e f
h h
EOF
```

produces the output:

```
a
b
c
d
e
f
g
h
```

RATIONALE

None.

FUTURE DIRECTIONS

None.

SEE ALSO

None.

COPYRIGHT

Portions of this text are reprinted and reproduced in electronic form from IEEE Std 1003.1, 2003 Edition, Standard for Information Technology -- Portable Operating System Interface (POSIX), The Open Group Base Specifications Issue 6, Copyright (C) 2001-2003 by the Institute of Electrical and Electronics Engineers, Inc and The Open Group. In the event of any discrepancy between this version and the original IEEE and The Open Group Standard, the original IEEE and The Open Group Standard is the referee document. The original Standard can be obtained online at <http://www.open-group.org/unix/online.html>.

NAME

tty - return user's terminal name

SYNOPSIS

tty

DESCRIPTION

The *tty* utility shall write to the standard output the name of the terminal that is open as standard input. The name that is used shall be equivalent to the string that would be returned by the *ttyname()* function defined in the System Interfaces volume of IEEE Std 1003.1-2001.

OPTIONS

The *tty* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section 12.2, Utility Syntax Guidelines.

OPERANDS

None.

STDIN

While no input is read from standard input, standard input shall be examined to determine whether or not it is a terminal, and, if so, to determine the name of the terminal.

INPUT FILES

None.

ENVIRONMENT VARIABLES

The following environment variables shall affect the execution of *tty*:

LANG Provide a default value for the internationalization variables that are unset or null. (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2, Internationalization Variables for the precedence of internationalization variables used to determine the values of locale categories.)

LC_ALL

If set to a non-empty string value, override the values of all the other internationalization variables.

LC_CTYPE

Determine the locale for the interpretation of sequences of bytes of text data as characters (for example, single-byte as opposed to multi-byte characters in arguments).

LC_MESSAGES

Determine the locale that should be used to affect the format and contents of diagnostic messages written to standard error and informative messages written to standard output.

NLSPATH

Determine the location of message catalogs for the processing of *LC_MESSAGES*.

ASYNCHRONOUS EVENTS

Default.

STDOUT

If standard input is a terminal device, a pathname of the terminal as specified by the *ttyname()* function defined in the System Interfaces volume of IEEE Std 1003.1-2001 shall be written in the following format:

"%s\n", <terminal name>

Otherwise, a message shall be written indicating that standard input is not connected to a terminal. In the POSIX locale, the *tty* utility shall use the format:

"not a tty\n"

STDERR

The standard error shall be used only for diagnostic messages.

OUTPUT FILES

None.

EXTENDED DESCRIPTION

None.

EXIT STATUS

The following exit values shall be returned:

- 0 Standard input is a terminal.
- 1 Standard input is not a terminal.
- >1 An error occurred.

CONSEQUENCES OF ERRORS

Default.

The following sections are informative.

APPLICATION USAGE

This utility checks the status of the file open as standard input against that of an implementation-defined set of files. It is possible that no match can be found, or that the match found need not be the same file as that which was opened for standard input (although they are the same device).

EXAMPLES

None.

RATIONALE

None.

FUTURE DIRECTIONS

None.

SEE ALSO

The System Interfaces volume of IEEE Std 1003.1-2001, *isatty()*, *ttyname()*

COPYRIGHT

Portions of this text are reprinted and reproduced in electronic form from IEEE Std 1003.1, 2003 Edition, Standard for Information Technology -- Portable Operating System Interface (POSIX), The Open Group Base Specifications Issue 6, Copyright (C) 2001-2003 by the Institute of Electrical and Electronics Engineers, Inc and The Open Group. In the event of any discrepancy between this version and the original IEEE and The Open Group Standard, the original IEEE and The Open Group Standard is the referee document. The original Standard can be obtained online at <http://www.open-group.org/unix/online.html>.

NAME

uname - return system name

SYNOPSIS

uname [-snrvma]

DESCRIPTION

By default, the *uname* utility shall write the operating system name to standard output. When options are specified, symbols representing one or more system characteristics shall be written to the standard output. The format and contents of the symbols are implementation-defined. On systems conforming to the System Interfaces volume of IEEE Std 1003.1-2001, the symbols written shall be those supported by the *uname()* function as defined in the System Interfaces volume of IEEE Std 1003.1-2001.

OPTIONS

The *uname* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section 12.2, Utility Syntax Guidelines.

The following options shall be supported:

- a** Behave as though all of the options **-mnrsv** were specified.
- m** Write the name of the hardware type on which the system is running to standard output.
- n** Write the name of this node within an implementation-defined communications network.
- r** Write the current release level of the operating system implementation.
- s** Write the name of the implementation of the operating system.
- v** Write the current version level of this release of the operating system implementation.

If no options are specified, the *uname* utility shall write the operating system name, as if the **-s** option had been specified.

OPERANDS

None.

STDIN

Not used.

INPUT FILES

None.

ENVIRONMENT VARIABLES

The following environment variables shall affect the execution of *uname*:

LANG Provide a default value for the internationalization variables that are unset or null. (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2, Internationalization Variables for the precedence of internationalization variables used to determine the values of locale categories.)

LC_ALL

If set to a non-empty string value, override the values of all the other internationalization variables.

LC_CTYPE

Determine the locale for the interpretation of sequences of bytes of text data as characters (for example, single-byte as opposed to multi-byte characters in arguments).

LC_MESSAGES

Determine the locale that should be used to affect the format and contents of diagnostic messages written to standard error.

NLSPATH

Determine the location of message catalogs for the processing of **LC_MESSAGES**.

uname(P)

uname(P)

ASYNCHRONOUS EVENTS

Default.

STDOUT

By default, the output shall be a single line of the following form:

```
"%s\n", <sysname>
```

If the **-a** option is specified, the output shall be a single line of the following form:

```
"%s %s %s %s %s\n", <sysname>, <nodename>, <release>,  
    <version>, <machine>
```

Additional implementation-defined symbols may be written; all such symbols shall be written at the end of the line of output before the <newline>.

If options are specified to select different combinations of the symbols, only those symbols shall be written, in the order shown above for the **-a** option. If a symbol is not selected for writing, its corresponding trailing <blank>s also shall not be written.

STDERR

The standard error shall be used only for diagnostic messages.

OUTPUT FILES

None.

EXTENDED DESCRIPTION

None.

EXIT STATUS

The following exit values shall be returned:

- 0 The requested information was successfully written.
- >0 An error occurred.

CONSEQUENCES OF ERRORS

Default.

The following sections are informative.

APPLICATION USAGE

Note that any of the symbols could include embedded <space>s, which may affect parsing algorithms if multiple options are selected for output.

The node name is typically a name that the system uses to identify itself for inter-system communication addressing.

EXAMPLES

The following command:

```
uname -sr
```

writes the operating system name and release level, separated by one or more <blank>s.

RATIONALE

It was suggested that this utility cannot be used portably since the format of the symbols is implementation-defined. The POSIX.1 working group could not achieve consensus on defining these formats in the underlying *uname()* function, and there was no expectation that this volume of IEEE Std 1003.1-2001 would be any more successful. Some applications may still find this historical utility of value. For example, the symbols could be used for system log entries or for comparison with operator or user input.

FUTURE DIRECTIONS

None.

SEE ALSO

The System Interfaces volume of IEEE Std 1003.1-2001, *uname()*

COPYRIGHT

Portions of this text are reprinted and reproduced in electronic form from IEEE Std 1003.1, 2003 Edition, Standard for Information Technology -- Portable Operating System Interface (POSIX), The Open Group Base Specifications Issue 6, Copyright (C) 2001-2003 by the Institute of Electrical and Electronics Engineers, Inc and The Open Group. In the event of any discrepancy between this version and the original IEEE and The Open Group Standard, the original IEEE and The Open Group Standard is the referee document. The original Standard can be obtained online at <http://www.open-group.org/unix/online.html>.

NAME

unexpand - convert spaces to tabs

SYNOPSIS

unexpand [**-a** | **-t** *tablist*] [*file...*]

DESCRIPTION

The *unexpand* utility shall copy files or standard input to standard output, converting <blank>s at the beginning of each line into the maximum number of <tab>s followed by the minimum number of <space>s needed to fill the same column positions originally filled by the translated <blank>s. By default, tabstops shall be set at every eighth column position. Each <backspace> shall be copied to the output, and shall cause the column position count for tab calculations to be decremented; the count shall never be decremented to a value less than one.

OPTIONS

The *unexpand* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section 12.2, Utility Syntax Guidelines.

The following options shall be supported:

-a In addition to translating <blank>s at the beginning of each line, translate all sequences of two or more <blank>s immediately preceding a tab stop to the maximum number of <tab>s followed by the minimum number of <space>s needed to fill the same column positions originally filled by the translated <blank>s.

-t *tablist* Specify the tab stops. The application shall ensure that the *tablist* option-argument is a single argument consisting of a single positive decimal integer or multiple positive decimal integers, separated by <blank>s or commas, in ascending order. If a single number is given, tabs shall be set *tablist* column positions apart instead of the default 8. If multiple numbers are given, the tabs shall be set at those specific column positions.

The application shall ensure that each tab-stop position *N* is an integer value greater than zero, and the list shall be in strictly ascending order. This is taken to mean that, from the start of a line of output, tabbing to position *N* shall cause the next character output to be in the (*N*+1)th column position on that line. When the **-t** option is not specified, the default shall be the equivalent of specifying **-t 8** (except for the interaction with **-a**, described below).

No <space>-to-<tab> conversions shall occur for characters at positions beyond the last of those specified in a multiple tab-stop list.

When **-t** is specified, the presence or absence of the **-a** option shall be ignored; conversion shall not be limited to the processing of leading <blank>s.

OPERANDS

The following operand shall be supported:

file A pathname of a text file to be used as input.

STDIN

See the INPUT FILES section.

INPUT FILES

The input files shall be text files.

ENVIRONMENT VARIABLES

The following environment variables shall affect the execution of *unexpand*:

LANG Provide a default value for the internationalization variables that are unset or null. (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2, Internationalization Variables for the precedence of internationalization variables used to determine the values of locale categories.)

LC_ALL

If set to a non-empty string value, override the values of all the other internationalization variables.

LC_CTYPE

Determine the locale for the interpretation of sequences of bytes of text data as characters (for example, single-byte as opposed to multi-byte characters in arguments and input files), the processing of <tab>s and <space>s, and for the determination of the width in column positions each character would occupy on an output device.

LC_MESSAGES

Determine the locale that should be used to affect the format and contents of diagnostic messages written to standard error.

NLSPATH

Determine the location of message catalogs for the processing of *LC_MESSAGES*.

ASYNCHRONOUS EVENTS

Default.

STDOUT

The standard output shall be equivalent to the input files with the specified <space>-to- <tab> conversions.

STDERR

The standard error shall be used only for diagnostic messages.

OUTPUT FILES

None.

EXTENDED DESCRIPTION

None.

EXIT STATUS

The following exit values shall be returned:

- 0 Successful completion.
- >0 An error occurred.

CONSEQUENCES OF ERRORS

Default.

The following sections are informative.

APPLICATION USAGE

One non-intuitive aspect of *unexpand* is its restriction to leading spaces when neither **-a** nor **-t** is specified. Users who always want to convert all spaces in a file can easily alias *unexpand* to use the **-a** or **-t 8** option.

EXAMPLES

None.

RATIONALE

On several occasions, consideration was given to adding a **-t** option to the *unexpand* utility to complement the **-t** in *expand* (see *expand*). The historical intent of *unexpand* was to translate multiple <blank>s into tab stops, where tab stops were a multiple of eight column positions on most UNIX systems. An early proposal omitted **-t** because it seemed outside the scope of the User Portability Utilities option; it was not described in any of the base documents. However, hard-coding tab stops every eight columns was not suitable for the international community and broke historical precedents for some vendors in the FORTRAN community, so **-t** was restored in conjunction with the list of valid extension categories considered by the standard developers. Thus, *unexpand* is now the logical converse of *expand*.

FUTURE DIRECTIONS

None.

SEE ALSO*expand , tabs***COPYRIGHT**

Portions of this text are reprinted and reproduced in electronic form from IEEE Std 1003.1, 2003 Edition, Standard for Information Technology -- Portable Operating System Interface (POSIX), The Open Group Base Specifications Issue 6, Copyright (C) 2001-2003 by the Institute of Electrical and Electronics Engineers, Inc and The Open Group. In the event of any discrepancy between this version and the original IEEE and The Open Group Standard, the original IEEE and The Open Group Standard is the referee document. The original Standard can be obtained online at <http://www.open-group.org/unix/online.html> .

NAME

uniq - report or filter out repeated lines in a file

SYNOPSIS

uniq [-c|-d|-u][*-f fields*][*-s char*][*input_file* [*output_file*]]

DESCRIPTION

The *uniq* utility shall read an input file comparing adjacent lines, and write one copy of each input line on the output. The second and succeeding copies of repeated adjacent input lines shall not be written.

Repeated lines in the input shall not be detected if they are not adjacent.

OPTIONS

The *uniq* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section 12.2, Utility Syntax Guidelines.

The following options shall be supported:

- c** Precede each output line with a count of the number of times the line occurred in the input.
- d** Suppress the writing of lines that are not repeated in the input.
- f *fields*** Ignore the first *fields* fields on each input line when doing comparisons, where *fields* is a positive decimal integer. A field is the maximal string matched by the basic regular expression:

[[[:blank:]]*[^[:blank:]]*

If the *fields* option-argument specifies more fields than appear on an input line, a null string shall be used for comparison.

- s *chars*** Ignore the first *chars* characters when doing comparisons, where *chars* shall be a positive decimal integer. If specified in conjunction with the **-f** option, the first *chars* characters after the first *fields* fields shall be ignored. If the *chars* option-argument specifies more characters than remain on an input line, a null string shall be used for comparison.
- u** Suppress the writing of lines that are repeated in the input.

OPERANDS

The following operands shall be supported:

input_file

A pathname of the input file. If the *input_file* operand is not specified, or if the *input_file* is **'-'**, the standard input shall be used.

output_file

A pathname of the output file. If the *output_file* operand is not specified, the standard output shall be used. The results are unspecified if the file named by *output_file* is the file named by *input_file*.

STDIN

The standard input shall be used only if no *input_file* operand is specified or if *input_file* is **'-'**. See the INPUT FILES section.

INPUT FILES

The input file shall be a text file.

ENVIRONMENT VARIABLES

The following environment variables shall affect the execution of *uniq*:

LANG Provide a default value for the internationalization variables that are unset or null. (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2, Internationalization Variables for the precedence of internationalization variables used to determine the values of locale categories.)

LC_ALL

If set to a non-empty string value, override the values of all the other internationalization variables.

LC_COLLATE

Determine the locale for ordering rules.

LC_CTYPE

Determine the locale for the interpretation of sequences of bytes of text data as characters (for example, single-byte as opposed to multi-byte characters in arguments and input files) and which characters constitute a <blank> in the current locale.

LC_MESSAGES

Determine the locale that should be used to affect the format and contents of diagnostic messages written to standard error.

NLSPATH

Determine the location of message catalogs for the processing of *LC_MESSAGES*.

ASYNCHRONOUS EVENTS

Default.

STDOUT

The standard output shall be used only if no *output_file* operand is specified. See the OUTPUT FILES section.

STDERR

The standard error shall be used only for diagnostic messages.

OUTPUT FILES

If the **-c** option is specified, the output file shall be empty or each line shall be of the form:

"%d %s", <number of duplicates>, <line>

otherwise, the output file shall be empty or each line shall be of the form:

"%s", <line>

EXTENDED DESCRIPTION

None.

EXIT STATUS

The following exit values shall be returned:

- 0 The utility executed successfully.
- >0 An error occurred.

CONSEQUENCES OF ERRORS

Default.

The following sections are informative.

APPLICATION USAGE

The *sort* utility can be used to cause repeated lines to be adjacent in the input file.

EXAMPLES

The following input file data (but flushed left) was used for a test series on *uniq*:

```
#01 foo0 bar0 foo1 bar1
#02 bar0 foo1 bar1 foo1
#03 foo0 bar0 foo1 bar1
#04
```

```
#05 foo0 bar0 foo1 bar1
#06 foo0 bar0 foo1 bar1
#07 bar0 foo1 bar1 foo0
```

What follows is a series of test invocations of the *uniq* utility that use a mixture of *uniq* options against the input file data. These tests verify the meaning of *adjacent*. The *uniq* utility views the input data as a sequence of strings delimited by '\n'. Accordingly, for the *fields*th member of the sequence, *uniq* interprets unique or repeated adjacent lines strictly relative to the *fields*+1th member.

This first example tests the line counting option, comparing each line of the input file data starting from the second field:

```
uniq -c -f 1 uniq_0I.t
1 #01 foo0 bar0 foo1 bar1
1 #02 bar0 foo1 bar1 foo0
1 #03 foo0 bar0 foo1 bar1
1 #04
2 #05 foo0 bar0 foo1 bar1
1 #07 bar0 foo1 bar1 foo0
```

The number '2', prefixing the fifth line of output, signifies that the *uniq* utility detected a pair of repeated lines. Given the input data, this can only be true when *uniq* is run using the **-f 1** option (which shall cause *uniq* to ignore the first field on each input line).

The second example tests the option to suppress unique lines, comparing each line of the input file data starting from the second field:

```
uniq -d -f 1 uniq_0I.t
#05 foo0 bar0 foo1 bar1
```

This test suppresses repeated lines, comparing each line of the input file data starting from the second field:

```
uniq -u -f 1 uniq_0I.t
#01 foo0 bar0 foo1 bar1
#02 bar0 foo1 bar1 foo1
#03 foo0 bar0 foo1 bar1
#04
#07 bar0 foo1 bar1 foo0
```

This suppresses unique lines, comparing each line of the input file data starting from the third character:

```
uniq -d -s 2 uniq_0I.t
```

In the last example, the *uniq* utility found no input matching the above criteria.

RATIONALE

Some historical implementations have limited lines to be 1080 bytes in length, which does not meet the implied {LINE_MAX} limit.

FUTURE DIRECTIONS

None.

SEE ALSO

comm, *sort*

COPYRIGHT

Portions of this text are reprinted and reproduced in electronic form from IEEE Std 1003.1, 2003 Edition, Standard for Information Technology -- Portable Operating System Interface (POSIX), The Open Group Base Specifications Issue 6, Copyright (C) 2001-2003 by the Institute of Electrical and

Electronics Engineers, Inc and The Open Group. In the event of any discrepancy between this version and the original IEEE and The Open Group Standard, the original IEEE and The Open Group Standard is the referee document. The original Standard can be obtained online at <http://www.open-group.org/unix/online.html> .

unlink(P)

unlink(P)

NAME

unlink - call the unlink function

SYNOPSIS

unlink *file*

DESCRIPTION

The *unlink* utility shall perform the function call:

unlink(*file*);

A user may need appropriate privilege to invoke the *unlink* utility.

OPTIONS

None.

OPERANDS

The following operands shall be supported:

file The pathname of an existing file.

STDIN

Not used.

INPUT FILES

Not used.

ENVIRONMENT VARIABLES

The following environment variables shall affect the execution of *unlink*:

LANG Provide a default value for the internationalization variables that are unset or null. (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2, Internationalization Variables for the precedence of internationalization variables used to determine the values of locale categories.)

LC_ALL If set to a non-empty string value, override the values of all the other internationalization variables.

LC_CTYPE Determine the locale for the interpretation of sequences of bytes of text data as characters (for example, single-byte as opposed to multi-byte characters in arguments).

LC_MESSAGES Determine the locale that should be used to affect the format and contents of diagnostic messages written to standard error.

NLSPATH Determine the location of message catalogs for the processing of *LC_MESSAGES*.

ASYNCHRONOUS EVENTS

Default.

STDOUT

None.

STDERR

The standard error shall be used only for diagnostic messages.

OUTPUT FILES

None.

EXTENDED DESCRIPTION

None.

unlink(P)

unlink(P)

EXIT STATUS

The following exit values shall be returned:

- 0 Successful completion.
- >0 An error occurred.

CONSEQUENCES OF ERRORS

Default.

The following sections are informative.

APPLICATION USAGE

None.

EXAMPLES

None.

RATIONALE

None.

FUTURE DIRECTIONS

None.

SEE ALSO

link() , *rm* , the System Interfaces volume of IEEE Std 1003.1-2001, *unlink()*

COPYRIGHT

Portions of this text are reprinted and reproduced in electronic form from IEEE Std 1003.1, 2003 Edition, Standard for Information Technology -- Portable Operating System Interface (POSIX), The Open Group Base Specifications Issue 6, Copyright (C) 2001-2003 by the Institute of Electrical and Electronics Engineers, Inc and The Open Group. In the event of any discrepancy between this version and the original IEEE and The Open Group Standard, the original IEEE and The Open Group Standard is the referee document. The original Standard can be obtained online at <http://www.open-group.org/unix/online.html> .

NAME

wc - word, line, and byte or character count

SYNOPSIS

wc [-c|-m][-lw][*file...*]

DESCRIPTION

The *wc* utility shall read one or more input files and, by default, write the number of <newline>s, words, and bytes contained in each input file to the standard output.

The utility also shall write a total count for all named files, if more than one input file is specified.

The *wc* utility shall consider a *word* to be a non-zero-length string of characters delimited by white space.

OPTIONS

The *wc* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section 12.2, Utility Syntax Guidelines.

The following options shall be supported:

- c** Write to the standard output the number of bytes in each input file.
- l** Write to the standard output the number of <newline>s in each input file.
- m** Write to the standard output the number of characters in each input file.
- w** Write to the standard output the number of words in each input file.

When any option is specified, *wc* shall report only the information requested by the specified options.

OPERANDS

The following operand shall be supported:

- file* A pathname of an input file. If no *file* operands are specified, the standard input shall be used.

STDIN

The standard input shall be used only if no *file* operands are specified. See the INPUT FILES section.

INPUT FILES

The input files may be of any type.

ENVIRONMENT VARIABLES

The following environment variables shall affect the execution of *wc*:

LANG Provide a default value for the internationalization variables that are unset or null. (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2, Internationalization Variables for the precedence of internationalization variables used to determine the values of locale categories.)

LC_ALL

If set to a non-empty string value, override the values of all the other internationalization variables.

LC_CTYPE

Determine the locale for the interpretation of sequences of bytes of text data as characters (for example, single-byte as opposed to multi-byte characters in arguments and input files) and which characters are defined as white space characters.

LC_MESSAGES

Determine the locale that should be used to affect the format and contents of diagnostic messages written to standard error and informative messages written to standard output.

NLSPATH

Determine the location of message catalogs for the processing of *LC_MESSAGES*.

ASYNCHRONOUS EVENTS

Default.

STDOUT

By default, the standard output shall contain an entry for each input file of the form:

```
"%d %d %d %s\n", <newlines>, <words>, <bytes>, <file>
```

If the **-m** option is specified, the number of characters shall replace the *<bytes>* field in this format.

If any options are specified and the **-l** option is not specified, the number of *<newline>*s shall not be written.

If any options are specified and the **-w** option is not specified, the number of words shall not be written.

If any options are specified and neither **-c** nor **-m** is specified, the number of bytes or characters shall not be written.

If no input *file* operands are specified, no name shall be written and no *<blank>*s preceding the path-name shall be written.

If more than one input *file* operand is specified, an additional line shall be written, of the same format as the other lines, except that the word **total** (in the POSIX locale) shall be written instead of a path-name and the total of each column shall be written as appropriate. Such an additional line, if any, is written at the end of the output.

STDERR

The standard error shall be used only for diagnostic messages.

OUTPUT FILES

None.

EXTENDED DESCRIPTION

None.

EXIT STATUS

The following exit values shall be returned:

- | | |
|----|------------------------|
| 0 | Successful completion. |
| >0 | An error occurred. |

CONSEQUENCES OF ERRORS

Default.

The following sections are informative.

APPLICATION USAGE

The **-m** option is not a switch, but an option at the same level as **-c**. Thus, to produce the full default output with character counts instead of bytes, the command required is:

```
wc -mlw
```

EXAMPLES

None.

RATIONALE

The output file format pseudo- *printf()* string differs from the System V version of *wc*:

```
"%7d%7d%7d %s\n"
```

which produces possibly ambiguous and unparsable results for very large files, as it assumes no number shall exceed six digits.

Some historical implementations use only *<space>*, *<tab>*, and *<newline>* as word separators. The equivalent of the ISO C standard *isspace()* function is more appropriate.

The **-c** option stands for "character" count, even though it counts bytes. This stems from the sometimes erroneous historical view that bytes and characters are the same size. Due to international requirements, the **-m** option (reminiscent of "multi-byte") was added to obtain actual character counts.

Early proposals only specified the results when input files were text files. The current specification more closely matches historical practice. (Bytes, words, and <newline>s are counted separately and the results are written when an end-of-file is detected.)

Historical implementations of the *wc* utility only accepted one argument to specify the options **-c**, **-l**, and **-w**. Some of them also had multiple occurrences of an option cause the corresponding count to be written multiple times and had the order of specification of the options affect the order of the fields on output, but did not document either of these. Because common usage either specifies no options or only one option, and because none of this was documented, the changes required by this volume of IEEE Std 1003.1-2001 should not break many historical applications (and do not break any historical conforming applications).

FUTURE DIRECTIONS

None.

SEE ALSO

cksum

COPYRIGHT

Portions of this text are reprinted and reproduced in electronic form from IEEE Std 1003.1, 2003 Edition, Standard for Information Technology -- Portable Operating System Interface (POSIX), The Open Group Base Specifications Issue 6, Copyright (C) 2001-2003 by the Institute of Electrical and Electronics Engineers, Inc and The Open Group. In the event of any discrepancy between this version and the original IEEE and The Open Group Standard, the original IEEE and The Open Group Standard is the referee document. The original Standard can be obtained online at <http://www.open-group.org/unix/online.html>.

NAME

who - display who is on the system

SYNOPSIS

who [-mTu]

who [-mu]-s[-bHlprt][*file*]

who [-mTu][-abdHlprt][*file*]

who -q [*file*]

who am i

who am I

DESCRIPTION

The *who* utility shall list various pieces of information about accessible users. The domain of accessibility is implementation-defined.

Based on the options given, *who* can also list the user's name, terminal line, login time, elapsed time since activity occurred on the line, and the process ID of the command interpreter for each current system user.

OPTIONS

The *who* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section 12.2, Utility Syntax Guidelines.

The following options shall be supported. The metavariables, such as *<line>*, refer to fields described in the STDOUT section.

- a** Process the implementation-defined database or named file with the **-b**, **-d**, **-l**, **-p**, **-r**, **-t**, **-T** and **-u** options turned on.
- b** Write the time and date of the last reboot.
- d** Write a list of all processes that have expired and not been respawned by the *init* system process. The *<exit>* field shall appear for dead processes and contain the termination and exit values of the dead process. This can be useful in determining why a process terminated.
- H** Write column headings above the regular output.
- l** (The letter ell.) List only those lines on which the system is waiting for someone to login. The *<name>* field shall be **LOGIN** in such cases. Other fields shall be the same as for user entries except that the *<state>* field does not exist.
- m** Output only information about the current terminal.
- p** List any other process that is currently active and has been previously spawned by *init*.
- q** (Quick.) List only the names and the number of users currently logged on. When this option is used, all other options shall be ignored.
- r** Write the current *run-level* of the *init* process.
- s** List only the *<name>*, *<line>*, and *<time>* fields. This is the default case.
- t** Indicate the last change to the system clock.
- T** Show the state of each terminal, as described in the STDOUT section.
- u** Write "idle time" for each displayed user in addition to any other information. The idle time is the time since any activity occurred on the user's terminal. The method of determining this is unspecified. This option shall list only those users who are currently logged in. The *<name>* is the user's login name. The *<line>* is the name of the line as found in the directory */dev*. The *<time>* is the time that the user logged in. The *<activity>* is the number of hours and minutes

since activity last occurred on that particular line. A dot indicates that the terminal has seen activity in the last minute and is therefore "current". If more than twenty-four hours have elapsed or the line has not been used since boot time, the entry shall be marked *<old>*. This field is useful when trying to determine whether a person is working at the terminal or not. The *<pid>* is the process ID of the user's login process.

OPERANDS

The following operands shall be supported:

am i, am I

In the POSIX locale, limit the output to describing the invoking user, equivalent to the **-m** option. The **am** and **i** or **I** must be separate arguments.

file Specify a pathname of a file to substitute for the implementation-defined database of logged-on users that *who* uses by default.

STDIN

Not used.

INPUT FILES

None.

ENVIRONMENT VARIABLES

The following environment variables shall affect the execution of *who*:

LANG Provide a default value for the internationalization variables that are unset or null. (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2, Internationalization Variables for the precedence of internationalization variables used to determine the values of locale categories.)

LC_ALL

If set to a non-empty string value, override the values of all the other internationalization variables.

LC_CTYPE

Determine the locale for the interpretation of sequences of bytes of text data as characters (for example, single-byte as opposed to multi-byte characters in arguments).

LC_MESSAGES

Determine the locale that should be used to affect the format and contents of diagnostic messages written to standard error.

LC_TIME

Determine the locale used for the format and contents of the date and time strings.

NLSPATH

Determine the location of message catalogs for the processing of *LC_MESSAGES*.

TZ

Determine the timezone used when writing date and time information. If *TZ* is unset or null, an unspecified default timezone shall be used.

ASYNCHRONOUS EVENTS

Default.

STDOUT

The *who* utility shall write its default format to the standard output in an implementation-defined format, subject only to the requirement of containing the information described above.

XSI-conformant systems shall write the default information to the standard output in the following general format:

```
<name>[<state>]<line><time>[<activity>][<pid>][<comment>][<exit>] 
```

The following format shall be used for the **-T** option:

```
"%s %c %s %s\n" <name>, <terminal state>, <terminal name>,
<time of login>
```

where *<terminal state>* is one of the following characters:

- + The terminal allows write access to other users.
- The terminal denies write access to other users.
- ? The terminal write-access state cannot be determined.

In the POSIX locale, the *<time of login>* shall be equivalent in format to the output of:

```
date +"%b %e %H:%M"
```

If the **-u** option is used with **-T**, the idle time shall be added to the end of the previous format in an unspecified format.

STDERR

The standard error shall be used only for diagnostic messages.

OUTPUT FILES

None.

EXTENDED DESCRIPTION

None.

EXIT STATUS

The following exit values shall be returned:

- 0 Successful completion.
- >0 An error occurred.

CONSEQUENCES OF ERRORS

Default.

The following sections are informative.

APPLICATION USAGE

The name *init* used for the system process is the most commonly used on historical systems, but it may vary.

The "domain of accessibility" referred to is a broad concept that permits interpretation either on a very secure basis or even to allow a network-wide implementation like the historical *rwho*.

EXAMPLES

None.

RATIONALE

Due to differences between historical implementations, the base options provided were a compromise to allow users to work with those functions. The standard developers also considered removing all the options, but felt that these options offered users valuable functionality. Additional options to match historical systems are available on XSI-conformant systems.

It is recognized that the *who* command may be of limited usefulness, especially in a multi-level secure environment. The standard developers considered, however, that having some standard method of determining the "accessibility" of other users would aid user portability.

No format was specified for the default *who* output for systems not supporting the XSI Extension. In such a user-oriented command, designed only for human use, this was not considered to be a deficiency.

The format of the terminal name is unspecified, but the descriptions of *ps*, *talk*, and *write* require that they use the same format.

It is acceptable for an implementation to produce no output for an invocation of *who* **mil**.

FUTURE DIRECTIONS

None.

SEE ALSO

msg

COPYRIGHT

Portions of this text are reprinted and reproduced in electronic form from IEEE Std 1003.1, 2003 Edition, Standard for Information Technology -- Portable Operating System Interface (POSIX), The Open Group Base Specifications Issue 6, Copyright (C) 2001-2003 by the Institute of Electrical and Electronics Engineers, Inc and The Open Group. In the event of any discrepancy between this version and the original IEEE and The Open Group Standard, the original IEEE and The Open Group Standard is the referee document. The original Standard can be obtained online at <http://www.open-group.org/unix/online.html>.