pwd

NAME

pwd - print name of current/working directory

SYNOPSIS

pwd [OPTION]...

DESCRIPTION

Print the full filename of the current working directory.

-L, --logical

use PWD from environment, even if it contains symlinks

-P, --physical

avoid all symlinks

--help display this help and exit

--version

output version information and exit

EXAMPLE

satyam@ubuntu:~$ pwd

/home/satyam

ls

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 -cftuvSUX nor --sort is

specified.

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 C-style escapes for nongraphic characters

--block-size=SIZE

scale sizes by SIZE before printing them. E.g., '--block-size=M' prints sizes in units of 1,048,576 bytes. See SIZE format

below.

-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, newest first

-C list entries by columns

--color[=WHEN]

colorize the output. WHEN defaults to 'always' or can be 'never' or 'auto'. More info below

-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 -ls --color

-F, --classify

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

--file-type

likewise, except do not append '*'

--format=WORD

across -x, commas -m, horizontal -x, long -l, single-column -1, verbose -l, vertical -C

--full-time

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

-g like -l, but do not list owner

--group-directories-first

group directories before files.

augment with a --sort option, but any use of --sort=none (-U) disables grouping

-G, --no-group

in a long listing, don't print group names

-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), slash (-p), file-type (--file-type), classify (-F)

-i, --inode

print the index number of each file

-I, --ignore=PATTERN

do not list implied entries matching shell PATTERN

-k, --kibibytes

use 1024-byte blocks

-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 user and group IDs

-N, --literal

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

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

-p, --indicator-style=slash

append / indicator to directories

-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

print the allocated size of each file, in blocks

-S sort by file size

--sort=WORD

sort by WORD instead of name: none -U, extension -X, size -S, time -t, version -v

--time=WORD

with -l, show time as WORD instead of modification time: atime -u, access -u, use -u, ctime -c, or status -c; 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, newest first

-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 natural sort of (version) numbers within text

-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

-Z, --context
print any SELinux security context of each file

-1 list one file per line

--help display this help and exit

--version
output version information and exit

SIZE is an integer and optional unit (example: 10M is 10*1024*1024). Units are K, M, G, T, P, E, Z, Y (powers of 1024) or KB, MB,

... (powers of 1000).

Using color to distinguish file types is disabled both by default and with --color=never. With --color=auto, ls emits color codes

only when standard output is connected to a terminal. The LS_COLORS environment variable can change the settings. Use the dircolors

command to set it.

Exit status:

0 if OK,

1 if minor problems (e.g., cannot access subdirectory),

2 if serious trouble (e.g., cannot access command-line argument).

EXAMPLE

```
satyam@ubuntu:~$ ls

3.cpp           clip         ddd               find10.cpp  give2.cpp  new
real1.cpp  temp.cpp    test6.cpp  tr.cpp

a.out           clip.cpp     ddd.cpp           find5.c     give3.cpp  new.cpp
satyam     Templates   test7.cpp  tt3.cpp

c1.cpp          cube         Desktop           find9.cpp   give5.cpp  nine
seven      ten         test8.cpp  two

cg              cube.cpp     Documents         five        give6.cpp
nine.cpp  seven.cpp  ten.cpp     test9.cpp  two.cpp

cg.cpp          dda          Downloads         five.cpp    give7.cpp  one
six        test10.cpp  test.cpp   untitled.cpp

cg.cpp~         dda.cpp      eight             fml.cpp     give8.cpp  one.cpp
six.cpp    test2.cpp   test.cpp~  Untitled Document 1

cgtemplate      dda.cpp~     eight.cpp         g1.cpp      give9.cpp
Pictures  t1.cpp     test3.cpp   three      Untitled Document 1~

cgtemplate.cpp  ddatest.cpp  examples.desktop  give10.cpp  Music      Public
t8.cpp     test4.cpp   three.cpp  Videos
```

mkdir

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 file mode (as in chmod), not a=rwx - umask

-p, --parents

no error if existing, make parent directories as needed

-v, --verbose

print a message for each created directory

-Z, --context=CTX

set the SELinux security context of each created directory to CTX

--help display this help and exit

--version

output version information and exit

EXAMPLE

satyam@ubuntu:~/Desktop$ mkdir new_directory

satyam@ubuntu:~/Desktop$ ls

new_directory

rmdir

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 and its ancestors; 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

EXAMPLE

```
satyam@ubuntu:~/Desktop$ ls
dir  folder  new_directory
satyam@ubuntu:~/Desktop$ rmdir folder
satyam@ubuntu:~/Desktop$ ls
dir  new_directory
```

touch

NAME

touch - change file timestamps

SYNOPSIS

touch [OPTION]... FILE...

DESCRIPTION

Update the access and modification times of each FILE to the current time.

A FILE argument that does not exist is created empty, unless -c or -h is supplied.

A FILE argument string of - is handled specially and causes touch to change the times of the file associated with standard output.

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)

-h, --no-dereference

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

-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.

EXAMPLE

satyam@ubuntu:~/Desktop$ ls

dir new_directory

```
satyam@ubuntu:~/Desktop$ touch newfile.txt
satyam@ubuntu:~/Desktop$ ls
dir  new_directory  newfile.txt
```

chmod

NAME

chmod - change file mode bits

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 file mode bits 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 mode bits.

The format of a symbolic mode is [ugoa...][[+-=][perms...]...], where perms is either zero or more letters from the set rwxXst, or a

single letter from the set ugo. Multiple symbolic modes 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 selected file mode bits to be added to the existing file mode bits of each file; - causes them to be

removed; and = causes them to be added and causes unmentioned bits to be removed except that a directory's unmentioned set user and

group ID bits are not affected.

The letters rwxXst select file mode bits for the affected users: read (r), write (w), execute (or search for directories) (x), exe-

cute/search only if the file is a directory or already has execute permission for some user (X), set user or group ID on execution

(s), restricted deletion flag or sticky bit (t). Instead of one or more of these letters, you can specify exactly one of the letters

ugo: 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. Omitted digits are

assumed to be leading zeros. The first digit selects the set user ID (4) and set group ID (2) and restricted deletion or 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.

SETUID AND SETGID BITS

chmod clears the set-group-ID bit of a regular file if the file's group ID does not match the user's effective group ID or one of the

user's supplementary group IDs, unless the user has appropriate privileges. Additional restrictions may cause the set-user-ID and

set-group-ID bits of MODE or RFILE to be ignored. This behavior depends on the policy and functionality of the underlying chmod system call. When in doubt, check the underlying system behavior.

chmod preserves a directory's set-user-ID and set-group-ID bits unless you explicitly specify otherwise. You can set or clear the bits with symbolic modes like u+s and g-s, and you can set (but not clear) the bits with a numeric mode.

RESTRICTED DELETION FLAG OR STICKY BIT

The restricted deletion flag or sticky bit is a single bit, whose interpretation depends on the file type. For directories, it prevents unprivileged users from removing or renaming a file in the directory unless they own the file or the directory; this is called the restricted deletion flag for the directory, and is commonly found on world-writable directories like /tmp. For regular files on some older systems, the bit saves the program's text image on the swap device so it will load more quickly when run; this is called the sticky bit.

OPTIONS

Change the mode of each FILE to MODE. With --reference, change the mode of each FILE to that of RFILE.

-c, --changes

like verbose but report only when a change is made

-f, --silent, --quiet

suppress most error messages

-v, --verbose

output a diagnostic for every file processed

--no-preserve-root

do not treat '/' specially (the default)

--preserve-root

fail to operate recursively on '/'

--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 of the form '[ugoa]*([-+=]([rwxXst]*|[ugo]))+|[-+=][0-7]+'.

EXAMPLE

```
satyam@ubuntu:~/Desktop$ ls -l
total 8
drwxrwxr-x 2 satyam satyam 4096 Jul 15 09:19 dir
drwxrwxr-x 2 satyam satyam 4096 Jul 15 09:17 new_directory
-rw-rw-r-- 1 satyam satyam    0 Jul 15 09:28 newfile.txt
satyam@ubuntu:~/Desktop$ chmod 777 newfile.txt
satyam@ubuntu:~/Desktop$ ls -l
total 8
```

```
drwxrwxr-x 2 satyam satyam 4096 Jul 15 09:19 dir
drwxrwxr-x 2 satyam satyam 4096 Jul 15 09:17 new_directory
-rwxrwxrwx 1 satyam satyam    0 Jul 15 09:28 newfile.txt
```

grep

NAME

grep, egrep, fgrep, rgrep - print lines matching a pattern

SYNOPSIS

grep [OPTIONS] PATTERN [FILE...]

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

DESCRIPTION

grep searches the named input FILEs (or standard input if no files are named, or if a single hyphen-minus (-) is given as file name)

for lines containing a match to the given PATTERN. By default, grep prints the matching lines.

In addition, three variant programs egrep, fgrep and rgrep are available. egrep is the same as grep -E. fgrep is the same as

grep -F. rgrep is the same as grep -r. Direct invocation as either egrep or fgrep is deprecated, but is provided to allow

historical applications that rely on them to run unmodified.

OPTIONS

Generic Program Information

--help Print a usage message briefly summarizing these command-line options and the bug-reporting address, then exit.

-V, --version

Print the version number of grep to the standard output stream. This version number should be included in all bug reports

(see below).

Matcher Selection

-E, --extended-regexp

Interpret PATTERN as an extended regular expression (ERE, see below). (-E is specified by POSIX.)

-F, --fixed-strings

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

-G, --basic-regexp

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

-P, --perl-regexp

Interpret PATTERN as a Perl regular expression (PCRE, see below). This is highly experimental and grep -P may warn of unimplemented features.

Matching Control

-e PATTERN, --regexp=PATTERN

Use PATTERN as the pattern. This can be used to specify multiple search patterns, or to protect a pattern beginning with a hyphen (-). (-e is specified by POSIX.)

-f FILE, --file=FILE

Obtain patterns from FILE, one per line. The empty file contains zero patterns, and therefore matches nothing. (-f is specified by POSIX.)

-i, --ignore-case

Ignore case distinctions in both the PATTERN and the input files. (-i is specified by POSIX.)

-v, --invert-match

Invert the sense of matching, to select non-matching lines. (-v is specified by POSIX.)

-w, --word-regexp

Select only those lines containing matches that form whole words. The test is that the matching substring must either be at

the beginning of the line, or preceded by a non-word constituent character. Similarly, it must be either at the end of the

line or followed by a non-word constituent character. Word-constituent characters are letters, digits, and the underscore.

-x, --line-regexp

Select only those matches that exactly match the whole line. (-x is specified by POSIX.)

-y Obsolete synonym for -i.

General Output Control

-c, --count

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

below), count non-matching lines. (-c is specified by POSIX.)

--color[=WHEN], --colour[=WHEN]

Surround the matched (non-empty) strings, matching lines, context lines, file names, line numbers, byte offsets, and

separators (for fields and groups of context lines) with escape sequences to display them in color on the terminal. The

colors are defined by the environment variable GREP_COLORS. The deprecated environment variable GREP_COLOR is still

supported, but its setting does not have priority. WHEN is never, always, or auto.

-L, --files-without-match

Suppress normal output; instead print the name of each input file from which no output would normally have been printed. The

scanning will stop on the first match.

-l, --files-with-matches

Suppress normal output; instead print the name of each input file from which output would normally have been printed. The

scanning will stop on the first match. (-l is specified by POSIX.)

-m NUM, --max-count=NUM

Stop reading a file after NUM matching lines. If the input is standard input from a regular file, and NUM matching lines are

output, grep ensures that the standard input is positioned to just after the last matching line before exiting, regardless of

the presence of trailing context lines. This enables a calling process to resume a search. When grep stops after NUM

matching lines, it outputs any trailing context lines. When the -c or --count option is also used, grep does not output a

count greater than NUM. When the -v or --invert-match option is also used, grep stops after outputting NUM non-matching

lines.

-o, --only-matching

Print only the matched (non-empty) parts of a matching line, with each such part on a separate output line.

-q, --quiet, --silent

Quiet; do not write anything to standard output. Exit immediately with zero status if any match is found, even if an error

was detected. Also see the -s or --no-messages option. (-q is specified by POSIX.)

-s, --no-messages

Suppress error messages about nonexistent or unreadable files. Portability note: unlike GNU grep, 7th Edition Unix grep did

not conform to POSIX, because it lacked -q and its -s option behaved like GNU grep's -q option. USG-style grep also lacked -q

but its -s option behaved like GNU grep. Portable shell scripts should avoid both -q and -s and should redirect standard and

error output to /dev/null instead. (-s is specified by POSIX.)

Output Line Prefix Control

-b, --byte-offset

Print the 0-based byte offset within the input file before each line of output. If -o (--only-matching) is specified, print

the offset of the matching part itself.

-H, --with-filename

Print the file name for each match. This is the default when there is more than one file to search.

-h, --no-filename

Suppress the prefixing of file names on output. This is the default when there is only one file (or only standard input) to

search.

--label=LABEL

Display input actually coming from standard input as input coming from file LABEL. This is especially useful when

implementing tools like zgrep, e.g., gzip -cd foo.gz | grep --label=foo -H something. See also the -H option.

-n, --line-number

Prefix each line of output with the 1-based line number within its input file. (-n is specified by POSIX.)

-T, --initial-tab

Make sure that the first character of actual line content lies on a tab stop, so that the alignment of tabs looks normal.

This is useful with options that prefix their output to the actual content: -H,-n, and -b. In order to improve the

probability that lines from a single file will all start at the same column, this also causes the line number and byte offset

(if present) to be printed in a minimum size field width.

-u, --unix-byte-offsets

Report Unix-style byte offsets. This switch causes grep to report byte offsets as if the file were a Unix-style text file,

i.e., with CR characters stripped off. This will produce results identical to running grep on a Unix machine. This option

has no effect unless -b option is also used; it has no effect on platforms other than MS-DOS and MS-Windows.

-Z, --null

Output a zero byte (the ASCII NUL character) instead of the character that normally follows a file name. For example, grep

-lZ outputs a zero byte after each file name instead of the usual newline. This option makes the output unambiguous, even in

the presence of file names containing unusual characters like newlines. This option can be used with commands like find

-print0, perl -0, sort -z, and xargs -0 to process arbitrary file names, even those that contain newline characters.

Context Line Control

-A NUM, --after-context=NUM

Print NUM lines of trailing context after matching lines. Places a line containing a group separator (--) between contiguous

groups of matches. With the -o or --only-matching option, this has no effect and a warning is given.

-B NUM, --before-context=NUM

Print NUM lines of leading context before matching lines. Places a line containing a group separator (--) between contiguous

groups of matches. With the -o or --only-matching option, this has no effect and a warning is given.

-C NUM, -NUM, --context=NUM

Print NUM lines of output context. Places a line containing a group separator (--) between contiguous groups of matches.

With the -o or --only-matching option, this has no effect and a warning is given.

File and Directory Selection

-a, --text

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

--binary-files=TYPE

If the first few bytes of a file indicate that the file contains binary data, assume that the file is of type TYPE. By

default, TYPE is binary, and grep normally outputs either a one-line message saying that a binary file matches, or no message

if there is no match. If TYPE is without-match, grep assumes that a binary file does not match; this is equivalent to the -I

option. If TYPE is text, grep processes a binary file as if it were text; this is equivalent to the -a option. Warning: grep

--binary-files=text might output binary garbage, which can have nasty side effects if the output is a terminal and if the

terminal driver interprets some of it as commands.

-D ACTION, --devices=ACTION

If an input file is a device, FIFO or socket, use ACTION to process it. By default, ACTION is read, which means that devices

are read just as if they were ordinary files. If ACTION is skip, devices are silently skipped.

-d ACTION, --directories=ACTION

If an input file is a directory, use ACTION to process it. By default, ACTION is read, i.e., read directories just as if they

were ordinary files. If ACTION is skip, silently skip directories. If ACTION is recurse, read all files under each

directory, recursively, following symbolic links only if they are on the command line. This is equivalent to the -r option.

--exclude=GLOB

Skip files whose base name matches GLOB (using wildcard matching). A file-name glob can use *, ?, and [...] as wildcards,

and \ to quote a wildcard or backslash character literally.

--exclude-from=FILE

Skip files whose base name matches any of the file-name globs read from FILE (using wildcard matching as described under

--exclude).

--exclude-dir=DIR

Exclude directories matching the pattern DIR from recursive searches.

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

--include=GLOB

Search only files whose base name matches GLOB (using wildcard matching as described under --exclude).

-r, --recursive

Read all files under each directory, recursively, following symbolic links only if they are on the command line. This is

equivalent to the -d recurse option.

-R, --dereference-recursive

Read all files under each directory, recursively. Follow all symbolic links, unlike -r.

Other Options

--line-buffered

Use line buffering on output. This can cause a performance penalty.

--mmap If possible, use the mmap(2) system call to read input, instead of the default read(2) system call. In some situations,

--mmap yields better performance. However, --mmap can cause undefined behavior (including core dumps) if an input file

shrinks while grep is operating, or if an I/O error occurs.

-U, --binary

Treat the file(s) as binary. By default, under MS-DOS and MS-Windows, grep guesses the file type by looking at the contents

of the first 32KB read from the file. If grep decides the file is a text file, it strips the CR characters from the original

file contents (to make regular expressions with ^ and $ work correctly). Specifying -U overrules this guesswork, causing all

files to be read and passed to the matching mechanism verbatim; if the file is a text file with CR/LF pairs at the end of each

line, this will cause some regular expressions to fail. This option has no effect on platforms other than MS-DOS and MS-

Windows.

-z, --null-data

Treat the input as a set of lines, each terminated by a zero byte (the ASCII NUL character) instead of a newline. Like the -Z

or --null option, this option can be used with commands like sort -z to process arbitrary file names.

EXAMPLE

satyam@ubuntu:~/Desktop$ cat newfile.txt

Programming is fun

Shell programming is a powerful tool

Shell scripts are more connected to operating system operations

We can write shell programs using text editor

Shell scripts end with .sh extensions

satyam@ubuntu:~/Desktop$ grep -i "program" newfile.txt

Programming is fun

Shell programming is a powerful tool

We can write shell programs using text editor

cal

NAME

cal, ncal — displays a calendar and the date of Easter

SYNOPSIS

cal [-3hjy] [-A number] [-B number] [[month] year]

cal [-3hj] [-A number] [-B number] -m month [year]

ncal [-3bhjJpwySM] [-A number] [-B number] [-s country_code] [[month] year]

ncal [-3bhJeoSM] [-A number] [-B number] [year]

ncal [-CN] [-H yyyy-mm-dd] [-d yyyy-mm]

DESCRIPTION

The cal utility displays a simple calendar in traditional format and ncal offers an alternative layout, more options and the date of

Easter. The new format is a little cramped but it makes a year fit on a 25x80 terminal. If arguments are not specified, the current

month is displayed.

The options are as follows:

-h Turns off highlighting of today.

-J Display Julian Calendar, if combined with the -o option, display date of Orthodox Easter according to the Julian Calendar.

-e Display date of Easter (for western churches).

-j Display Julian days (days one-based, numbered from January 1).

-m month

Display the specified month. If month is specified as a decimal number, it may be followed by the letter ‘f’ or ‘p’ to indi-

cate the following or preceding month of that number, respectively.

-o Display date of Orthodox Easter (Greek and Russian Orthodox Churches).

-p Print the country codes and switching days from Julian to Gregorian Calendar as they are assumed by ncal. The country code as

determined from the local environment is marked with an asterisk.

-s country_code

Assume the switch from Julian to Gregorian Calendar at the date associated with the country_code. If not specified, ncal tries

to guess the switch date from the local environment or falls back to September 2, 1752. This was when Great Britain and her

colonies switched to the Gregorian Calendar.

-w Print the number of the week below each week column.

-y Display a calendar for the specified year. This option is implied when a year but no month are specified on the command line.

-3 Display the previous, current and next month surrounding today.

-1 Display only the current month. This is the default.

-A number

Months to add after. The specified number of months is added to the end of the display. This is in addition to any date range

selected by the -y, -3, or -1 options. For example, “cal -y -B2 -A2” shows everything from November of the previous year to

February of the following year. Negative numbers are allowed, in which case the specified number of months is subtracted. For

example, “cal -y -B-6” shows July to December. And “cal -A11” simply shows the next 12 months.

-B number

Months to add before. The specified number of months is added to the beginning of the display. See -A for examples.

-C Switch to cal mode.

-N Switch to ncal mode.

-d yyyy-mm

Use yyyy-mm as the current date (for debugging of date selection).

-H yyyy-mm-dd

Use yyyy-mm-dd as the current date (for debugging of highlighting).

-M Weeks start on Monday.

-S Weeks start on Sunday.

-b Use oldstyle format for ncal output.

A single parameter specifies the year (1–9999) to be displayed; note the year must be fully specified: “cal 89” will not display a cal-

endar for 1989. Two parameters denote the month and year; the month is either a number between 1 and 12, or a full or abbreviated name

as specified by the current locale. Month and year default to those of the current system clock and time zone (so “cal -m 8” will dis-

play a calendar for the month of August in the current year).

Not all options can be used together. For example, the options -y, -3, and -1 are mutually exclusive. If inconsistent options are

given, the later ones take precedence over the earlier ones.

A year starts on January 1.

EXAMPLE

```
satyam@ubuntu:~/Desktop$ cal
     July 2019
Su Mo Tu We Th Fr Sa
    1  2  3  4  5  6
 7  8  9 10 11 12 13
14 15 16 17 18 19 20
21 22 23 24 25 26 27
28 29 30 31

satyam@ubuntu:~/Desktop$ cal -3
     June 2019              July 2019            August 2019
Su Mo Tu We Th Fr Sa  Su Mo Tu We Th Fr Sa  Su Mo Tu We Th Fr Sa
                   1      1  2  3  4  5  6              1  2  3
 2  3  4  5  6  7  8   7  8  9 10 11 12 13   4  5  6  7  8  9 10
 9 10 11 12 13 14 15  14 15 16 17 18 19 20  11 12 13 14 15 16 17
16 17 18 19 20 21 22  21 22 23 24 25 26 27  18 19 20 21 22 23 24
23 24 25 26 27 28 29  28 29 30 31           25 26 27 28 29 30 31
30
```

cat

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 nonempty output lines, overrides -n

-e equivalent to -vE

-E, --show-ends

display $ at end of each line

-n, --number

number all output lines

-s, --squeeze-blank

suppress repeated empty output lines

-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.

EXAMPLES

cat f - g

Output f's contents, then standard input, then g's contents.

cat Copy standard input to standard output.

EXAMPLE CODE

satyam@ubuntu:~/Desktop$ cat newfile.txt

Programming is fun

Shell programming is a powerful tool

Shell scripts are more connected to operating system operations

We can write shell programs using text editor

Shell scripts end with .sh extensions