NAME

sort - sort or merge records (lines) of text and binary files

SYNOPSIS

sort [**-bcCdfghiRMmnrsuVz**] [**-k** *field1*[,*field2*]] [**-S** *memsize*] [**-T** *dir*] [**-t** *char*] [**-o** *output*] [*file ...*]
sort --help
sort --version

DESCRIPTION

The **sort** utility sorts text and binary files by lines. A line is a record separated from the subsequent record by a newline (default) or NUL '\0' character (-z option). A record can contain any printable or unprintable characters. Comparisons are based on one or more sort keys extracted from each line of input, and are performed lexicographically, according to the current locale's collating rules and the specified command-line options that can tune the actual sorting behavior. By default, if keys are not given, **sort** uses entire lines for comparison.

The command line options are as follows:

-c, **--check**, **-C**, **--check=silent|quiet**
: Check that the single input file is sorted. If the file is not sorted, **sort** produces the appropriate error messages and exits with code 1, otherwise returns 0. If **-C** or **--check=silent** is specified, **sort** produces no output. This is a "silent" version of **-c**.

-m, **--merge**
: Merge only. The input files are assumed to be pre-sorted. If they are not sorted the output order is undefined.

-o *output*, **--output**=*output*
: Print the output to the *output* file instead of the standard output.

-S *size*, **--buffer-size**=*size*
: Use *size* for the maximum size of the memory buffer. Size modifiers %,b,K,M,G,T,P,E,Z,Y can be used. If a memory limit is not explicitly specified, **sort** takes up to about 90% of available memory. If the file size is too big to fit into the memory buffer, the temporary disk files are used to perform the sorting.

-T *dir*, **--temporary-directory**=*dir*
: Store temporary files in the directory *dir*. The default path is the value of the environment variable TMPDIR or */var/tmp* if TMPDIR is not defined.

-u, **--unique**
Unique keys. Suppress all lines that have a key that is equal to an already processed one. This option, similarly to **-s**, implies a stable sort. If used with **-c** or **-C**, **sort** also checks that there are no lines with duplicate keys.

-s Stable sort. This option maintains the original record order of records that have an equal key. This is a non-standard feature, but it is widely accepted and used.

--version
Print the version and silently exits.

--help Print the help text and silently exits.

The following options override the default ordering rules. When ordering options appear independently of key field specifications, they apply globally to all sort keys. When attached to a specific key (see **-k**), the ordering options override all global ordering options for the key they are attached to.

-b, **--ignore-leading-blanks**
Ignore leading blank characters when comparing lines.

-d, **--dictionary-order**
Consider only blank spaces and alphanumeric characters in comparisons.

-f, **--ignore-case**
Convert all lowercase characters to their uppercase equivalent before comparison, that is, perform case-independent sorting.

-g, **--general-numeric-sort**, **--sort=general-numeric**
Sort by general numerical value. As opposed to **-n**, this option handles general floating points. It has a more permissive format than that allowed by **-n** but it has a significant performance drawback.

-h, **--human-numeric-sort**, **--sort=human-numeric**
Sort by numerical value, but take into account the SI suffix, if present. Sort first by numeric sign (negative, zero, or positive); then by SI suffix (either empty, or ‘k’ or ‘K’, or one of ‘MGTPEZY’, in that order); and finally by numeric value. The SI suffix must immediately follow the number. For example, ’12345K’ sorts before ’1M’, because M is "larger" than K. This sort option is useful for sorting the output of a single invocation of ’df’ command with **-h** or **-H** options (human-readable).

-i, **--ignore-nonprinting**

Ignore all non-printable characters.

-M, **--month-sort**, **--sort=month**

Sort by month abbreviations. Unknown strings are considered smaller than the month names.

-n, **--numeric-sort**, **--sort=numeric**

Sort fields numerically by arithmetic value. Fields are supposed to have optional blanks in the beginning, an optional minus sign, zero or more digits (including decimal point and possible thousand separators).

-R, **--random-sort**, **--sort=random**

Sort by a random order. This is a random permutation of the inputs except that the equal keys sort together. It is implemented by hashing the input keys and sorting the hash values. The hash function is chosen randomly. The hash function is randomized by **/dev/random** content, or by file content if it is specified by **--random-source**. Even if multiple sort fields are specified, the same random hash function is used for all of them.

-r, **--reverse**

Sort in reverse order.

-V, **--version-sort**

Sort version numbers. The input lines are treated as file names in form PREFIX VERSION SUFFIX, where SUFFIX matches the regular expression "(.([A-Za-z~][A-Za-z0-9~]*)?)*". The files are compared by their prefixes and versions (leading zeros are ignored in version numbers, see example below). If an input string does not match the pattern, then it is compared using the byte compare function. All string comparisons are performed in C locale, the locale environment setting is ignored.

Example:

$ ls sort* | sort -V

sort-1.022.tgz

sort-1.23.tgz

sort-1.23.1.tgz

sort-1.024.tgz

sort-1.024.003.

sort-1.024.003.tgz

sort-1.024.07.tgz

sort-1.024.009.tgz

The treatment of field separators can be altered using these options:

-b, **--ignore-leading-blanks**
Ignore leading blank space when determining the start and end of a restricted sort key (see **-k**). If **-b** is specified before the first **-k** option, it applies globally to all key specifications. Otherwise, **-b** can be attached independently to each *field* argument of the key specifications. **-b**.

-k *field1*[,*field2*], **--key**=*field1*[,*field2*]
Define a restricted sort key that has the starting position *field1*, and optional ending position *field2* of a key field. The **-k** option may be specified multiple times, in which case subsequent keys are compared when earlier keys compare equal. The **-k** option replaces the obsolete options **+***pos1* and **-***pos2*, but the old notation is also supported.

-t *char*, **--field-separator**=*char*
Use *char* as a field separator character. The initial *char* is not considered to be part of a field when determining key offsets. Each occurrence of *char* is significant (for example, "*charchar*" delimits an empty field). If **-t** is not specified, the default field separator is a sequence of blank space characters, and consecutive blank spaces do *not* delimit an empty field, however, the initial blank space *is* considered part of a field when determining key offsets. To use NUL as field separator, use **-t** '\0'.

-z, **--zero-terminated**
Use NUL as record separator. By default, records in the files are supposed to be separated by the newline characters. With this option, NUL ('\0') is used as a record separator character.

Other options:

--batch-size=*num*
Specify maximum number of files that can be opened by **sort** at once. This option affects behavior when having many input files or using temporary files. The default value is 16.

--compress-program=*PROGRAM*

Use PROGRAM to compress temporary files. PROGRAM must compress standard input to standard output, when called without arguments. When called with argument **-d** it must decompress standard input to standard output. If PROGRAM fails, **sort** must exit with error. An example of PROGRAM that can be used here is bzip2.

--random-source=*filename*

In random sort, the file content is used as the source of the 'seed' data for the hash function choice. Two invocations of random sort with the same seed data will use the same hash function and will produce the same result if the input is also identical. By default, file **/dev/random** is used.

--debug

Print some extra information about the sorting process to the standard output.

--files0-from=*filename*

Take the input file list from the file *filename*. The file names must be separated by NUL (like the output produced by the command "find ... -print0").

--radixsort

Try to use radix sort, if the sort specifications allow. The radix sort can only be used for trivial locales (C and POSIX), and it cannot be used for numeric or month sort. Radix sort is very fast and stable.

--mergesort

Use mergesort. This is a universal algorithm that can always be used, but it is not always the fastest.

--qsort

Try to use quick sort, if the sort specifications allow. This sort algorithm cannot be used with **-u** and **-s**.

--heapsort

Try to use heap sort, if the sort specifications allow. This sort algorithm cannot be used with **-u** and **-s**.

--mmap

Try to use file memory mapping system call. It may increase speed in some cases.

The following operands are available:

file The 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 is used.

A field is defined as a maximal sequence of characters other than the field separator and record separator (newline by default). Initial blank spaces are included in the field unless **-b** has been specified; the first blank space of a sequence of blank spaces acts as the field separator and is included in the field (unless **-t** is specified). For example, all blank spaces at the beginning of a line are considered to be part of the first field.

Fields are specified by the **-k** *field1*[,*field2*] command-line option. If *field2* is missing, the end of the key defaults to the end of the line.

The arguments *field1* and *field2* have the form *m.n (m,n > 0)* and can be followed by one or more of the modifiers **b**, **d**, **f**, **i**, **n**, **g**, **M** and **r**, which correspond to the options discussed above. When **b** is specified it applies only to *field1* or *field2* where it is specified while the rest of the modifiers apply to the whole key field regardless if they are specified only with *field1* or *field2* or both. A *field1* position specified by *m.n* is interpreted as the *n*th character from the beginning of the *m*th field. A missing *.n* in *field1* means '.1', indicating the first character of the *m*th field; if the **-b** option is in effect, *n* is counted from the first non-blank character in the *m*th field; *m*.1b refers to the first non-blank character in the *m*th field. 1.*n* refers to the *n*th character from the beginning of the line; if *n* is greater than the length of the line, the field is taken to be empty.

*n*th positions are always counted from the field beginning, even if the field is shorter than the number of specified positions. Thus, the key can really start from a position in a subsequent field.

A *field2* position specified by *m.n* is interpreted as the *n*th character (including separators) from the beginning of the *m*th field. A missing *.n* indicates the last character of the *m*th field; *m* = 0 designates the end of a line. Thus the option **-k** *v.x,w.y* is synonymous with the obsolete option *+v-1.x-1* **-***w-1.y*; when *y* is omitted, **-k** *v.x,w* is synonymous with *+v-1.x-1* **-***w.0*. The obsolete *+pos1* **-***pos2* option is still supported, except for **-***w.0b*, which has no **-k** equivalent.

ENVIRONMENT

LC_COLLATE
: Locale settings to be used to determine the collation for sorting records.

LC_CTYPE
: Locale settings to be used to case conversion and classification of characters, that is, which characters are considered whitespaces, etc.

LC_MESSAGES
: Locale settings that determine the language of output messages that **sort** prints out.

LC_NUMERIC
Locale settings that determine the number format used in numeric sort.

LC_TIME Locale settings that determine the month format used in month sort.

LC_ALL Locale settings that override all of the above locale settings. This environment variable can be used to set all these settings to the same value at once.

LANG Used as a last resort to determine different kinds of locale-specific behavior if neither the respective environment variable, nor LC_ALL are set.

TMPDIR Path to the directory in which temporary files will be stored. Note that TMPDIR may be overridden by the **-T** option.

GNUSORT_NUMERIC_COMPATIBILITY
If defined **-t** will not override the locale numeric symbols, that is, thousand separators and decimal separators. By default, if we specify **-t** with the same symbol as the thousand separator or decimal point, the symbol will be treated as the field separator. Older behavior was less definite; the symbol was treated as both field separator and numeric separator, simultaneously. This environment variable enables the old behavior.

FILES

*/var/tmp/.bsdsort.PID.**	Temporary files.
/dev/random	Default seed file for the random sort.

EXIT STATUS

The **sort** utility shall exit with one of the following values:

0 Successfully sorted the input files or if used with **-c** or **-C**, the input file already met the sorting criteria.
1 On disorder (or non-uniqueness) with the **-c** or **-C** options.
2 An error occurred.

SEE ALSO

comm(1), join(1), uniq(1)

STANDARDS

The **sort** utility is compliant with the IEEE Std 1003.1-2008 ("POSIX.1") specification.

The flags [**-ghRMSsTVz**] are extensions to the POSIX specification.

All long options are extensions to the specification, some of them are provided for compatibility with GNU versions and some of them are own extensions.

The old key notations **+***pos1* and **-***pos2* come from older versions of **sort** and are still supported but their use is highly discouraged.

HISTORY

A **sort** command first appeared in Version 1 AT&T UNIX.

AUTHORS

Gabor Kovesdan <*gabor@FreeBSD.org*>,

Oleg Moskalenko <*mom040267@gmail.com*>

NOTES

This implementation of **sort** has no limits on input line length (other than imposed by available memory) or any restrictions on bytes allowed within lines.

The performance depends highly on locale settings, efficient choice of sort keys and key complexity. The fastest sort is with locale C, on whole lines, with option **-s**. In general, locale C is the fastest, then single-byte locales follow and multi-byte locales as the slowest but the correct collation order is always respected. As for the key specification, the simpler to process the lines the faster the search will be.

When sorting by arithmetic value, using **-n** results in much better performance than **-g** so its use is encouraged whenever possible.