IEEE P1003.2 Draft 11.2 – September 1991

4.60 `tail` — Copy the last part of a file

4.60.1 Synopsis

`tail` [`−f`] [`−c` *number* | `−n` *number*] [*file*]

Obsolescent versions:

`tail` −[*number*][`c`|`l`][`f`] [*file*]

`tail` +[*number*][`c`|`l`][`f`] [*file*]

4.60.2 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 (or the ±*number* portion of the argument to the obsolescent version). The option-argument *number* shall be counted in units of lines or bytes, according to the options `−n` and `−c` (or, in the obsolescent version, the appended option suffixes `l` or `c`).

Tails relative to the end of the file may be saved in an internal buffer, and thus may be limited in length. Implementations shall ensure that such a buffer, if any, is no smaller than {LINE_MAX}*10 bytes.

4.60.3 Options

The `tail` utility shall conform to the utility argument syntax guidelines described in standard described in 2.10.2, except that the obsolescent version accepts multicharacter options that can preceded by a plus sign.

The following options shall be supported by the implementation in the nonobsolescent version:

`−c` *number* The *number* option-argument shall be 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.
none	Relative to the end of the file.

The origin for counting shall be 1; i.e., `-c +1` represents the first 1
byte of the file, `-c -1` the last. 1

`−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; i.e., `-n +1` represents the first line of the file, `-n -1` the last.

In the obsolescent version, an argument beginning with a `-` or `+` can be used as a single option. The argument ±*number* with the letter `c` specified as a suffix shall be equivalent to `−c` ±*number* ; ±*number* with the letter `l` specified as a suffix, or with neither `c` nor `l` as a suffix, shall be equivalent to `−n` ±*number* . If *number* is not specified in these forms, 10 shall be used. The letter `f` specified as a suffix shall be equivalent to specifying the `−f` option. If the `-[`*number* `]c[f]` form is used and neither *number* nor the `f` suffix is specified, it shall be interpreted as the `−c` *number* option.

In the nonobsolescent form, if neither `−c` nor `−n` is specified, `-n 10` shall be assumed.

4.60.4 Operands

The following operand shall be supported by the implementation:

file A pathname of an input file. If no *file* operands are specified, the standard input shall be used.

4.60.5 External Influences

4.60.5.1 Standard Input

The standard input shall be used only if no *file* operands are specified. See Input Files.

4.60.5.2 Input Files

If the `−c` option is specified, the input file can contain arbitrary data; otherwise, the input file shall be a text file.

4.60.5.3 Environment Variables

The following environment variables shall affect the execution of `tail`:

LANG This variable shall determine the locale to use for the locale categories when both **LC_ALL** and the corresponding environment variable (beginning with **LC_**) do not specify a locale. See 2.6.

LC_ALL This variable shall determine the locale to be used to override any values for locale categories specified by the settings of **LANG** or any environment variables beginning with **LC_**.

LC_CTYPE This variable shall determine the locale for the interpretation of sequences of bytes of text data as characters (e.g., single- versus multibyte characters in arguments and input files).

LC_MESSAGES This variable shall determine the language in which messages should be written.

4.60.5.4 Asynchronous Events

Default.

4.60.6 External Effects

4.60.6.1 Standard Output

The designated portion of the input file shall be written to standard output.

4.60.6.2 Standard Error

Used only for diagnostic messages.

4.60.6.3 Output Files

None.

4.60.7 Extended Description

None.

4.60.8 Exit Status

The `tail` utility shall exit with one of the following values:

- 0 Successful completion.
- >0 An error occurred.

4.60.9 Consequences of Errors

Default.

4.60.10 Rationale. *(This subclause is not a part of P1003.2)*

Usage, Examples

The nonobsolescent version of `tail` was created to allow conformance to the Utility Syntax Guidelines. The historical `−b` option was omitted because of the general nonportability of block-sized units of text. The `−c` option historically meant ‘‘characters,’’ but this standard indicates that it means ‘‘bytes.’’ This was selected to allow reasonable implementations when multibyte characters are possible; it was not named `−b` to avoid confusion with the historical `−b`.

Note that the `−c` option should be used with caution when the input is a text file containing multibyte characters; it may produce output that does not start on a character boundary.

The origin of counting both lines and bytes is 1, matching all widespread histori- 1
cal implementations. 1

The restriction on the internal buffer is a compromise between the historical System V implementation of 4K and the BSD 32K.

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.

Although the input file to `tail` can be any type, the results need not be what would be expected on some character special device files or on file types not described by POSIX.1 {8}. Since the standard 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.

History of Decisions Made

The developers of the standard originally decided that `tail`, and its frequent companion, `head`, were useful mostly to interactive users, and not application programs. However, balloting input suggested that these utilities actually do find significant use in scripts, such as to write out portions of log files. The balloters also challenged the working group's assumption that clever use of `sed` could be an appropriate substitute for `tail`.

The `−f` option has been implemented as a loop that sleeps for one 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 says that `tail` ignores the `−f` option if the input file is a pipe (pipe and FIFO on systems that support FIFO s). On BSD-based systems, this has been true; on System V-based systems, this was true when input was taken from standard input, but behaved as on other files 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, POSIX.2 requires this behavior. However, since the `−f` option is useful on a FIFO , POSIX.2 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.

An earlier draft had the synopsis line:

`tail [−c | −l ][−f][−n` *number* `][`*file*`]`

This was changed to the current form based on comments and objections noting that `−c` was almost never used without specifying a number and there was no need to specify `−l` if `−n` *number* was given.

4.61 `tee` — Duplicate standard input

4.61.1 Synopsis

`tee [−ai] [`*file* ...`]`

4.61.2 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.

The options determine if the specified files are overwritten or appended to.

4.61.3 Options

The `tee` utility shall conform to the utility argument syntax guidelines described in 2.10.2.

The following options shall be supported by the implementation:

`−a` Append the output to the files rather than overwriting them.

`−i` Ignore the SIGINT signal.

4.61.4 Operands

The following operands shall be supported by the implementation:

file A pathname of an output file. Implementations shall support processing of at least 13 *file* operands.

4.61.5 External Influences

4.61.5.1 Standard Input

The standard input can be of any type.

4.61.5.2 Input Files

None.

4.61.5.3 Environment Variables

The following environment variables shall affect the execution of `tee`: