-- --

IEEE P1003.2 Draft 11.2 − September 1991

4.71 wc — Word, line, and byte count

4.71.1 Synopsis

wc [−clw] [*file* . . .]

4.71.2 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 nonzero-length string of characters delimited by white space.

4.71.3 Options

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

The following options shall be supported by the implementation:

−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.

−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 option(s).

4.71.4 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.

4.71.5 External Influences

4.71.5.1 Standard Input

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

4.71.5.2 Input Files

The input files may be of any type.

4.71.5.3 Environment Variables

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

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) and which characters are defined as ‘‘white space’’ characters.

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

4.71.5.4 Asynchronous Events

Default.

4.71.6 External Effects

4.71.6.1 Standard Output

By default, the standard output shall contain a line for each input file of the form:

`"%d %d %d %s\n"`, *<newlines>*, *<words>*, *<bytes>*, *<file>*

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 the `−c` option is not specified, the number of bytes shall not be written.

If no input *file* operands are specified, no name shall be written and no `<blank>`s preceding the pathname 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 pathname and the total of each column shall be written as appropriate. Such an additional line, if any, shall be written at the end of the output.

4.71.6.2 Standard Error

Used only for diagnostic messages.

4.71.6.3 Output Files

None.

4.71.7 Extended Description

None.

4.71.8 Exit Status

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

0 Successful completion.

>0 An error occurred.

4.71.9 Consequences of Errors

Default.

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

Examples, Usage

None.

History of Decisions Made

The output file format pseudo- *printf* () string was derived from the HP-UX version of `wc`; the System V version:

```
"%7d%7d%7d %s\n"
```

produces possibly ambiguous and unparsable results for very large files, as it assumes no number will exceed six digits.

Some historical implementations use only `<space>`, `<tab>`, and `<newline>` as word separators. The equivalent of the C Standard {7} *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.

Earlier drafts only specified the results when input files were text files. The current specification more closely matches existing 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 output multiple times and having 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 standard should not break many existing applications (and does not break any historical portable applications.)

4.72 `xargs` — Construct argument list(s) and invoke utility

4.72.1 Synopsis

`xargs` [`-t`] [`-n` *number* [`-x`]] [`-s` *size*] [*utility* [*argument* ...]]

4.72.2 Description

The `xargs` utility shall construct a command line consisting of the *utility* and *argument* operands specified followed by as many arguments read in sequence from standard input as will fit in length and number constraints specified by the options. The `xargs` utility shall then invoke the constructed command line and wait for its completion. This sequence shall be repeated until an end-of-file condition is detected on standard input or an invocation of a constructed command line returns an exit status of 255. 1

Arguments in the standard input shall be separated by unquoted `<blank>`s, or unescaped `<blank>`s or `<newline>`s. A string of zero or more nondouble-quote (`"`) and non-`<newline>` characters can be quoted by enclosing them in double-quotes. A string of zero or more nonapostrophe (`'`) and non-`<newline>` characters can be quoted by enclosing them in apostrophes. Any unquoted character can be escaped by preceding it with a backslash. The *utility* shall be executed one or more times until the end-of-file is reached. The results are unspecified if the utility named by *utility* attempts to read from its standard input.

The generated command line length shall be the sum of the size in bytes of the utility name and each argument treated as strings, including a null byte terminator for each of these strings. The `xargs` utility shall limit the command line length such that when the command line is invoked, the combined argument and environment lists (see the *exec* family of functions in POSIX.1 {8} 3.1.2) shall not exceed {ARG_MAX}−2048 bytes. Within this constraint, if neither the `-n` nor the `-s` option is specified, the default command line length shall be at least {LINE_MAX}.

4.72.3 Options

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

The following options shall be supported by the implementation:

`-n` *number* Invoke *utility* using as many standard input arguments as possible, up to *number* (a positive decimal integer) arguments maximum. Fewer arguments shall be used if:

— The command line length accumulated exceeds the size specified by the `-s` option (or {LINE_MAX} if there is no `-s` option), or