IEEE P1003.2a Draft 8 – December 1991

This description of `crontab` is designed to support only users with normal privileges. The format of the input is based on the System V `crontab`; however, there is no requirement here that the actual system database used by the `cron` daemon (or similar mechanism) use this format internally. For example, systems derived from BSD will likely have an additional field appended that indicates the user identity to be used when the job is submitted.

The `−e` option was adopted from the *SVID* as a user convenience, although it does not exist in all historical implementations.

5.6 `csplit` — Split files based on context

5.6.1 Synopsis

`csplit [−ks][−f` *prefix* `][−n` *number* `]` *file arg1 . . . argn*

5.6.2 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.

5.6.3 Options

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

The following options shall be supported by the implementation:

`−f` *prefix* Name the created files *prefix* `00`, *prefix* `01`, . . . , *prefixn* . The default shall be `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.

5.6.4 Operands

The following operands shall be supported by the implementation:

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:

/rexp/[*offset*]
: Create a file 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 *rexp* shall follow the rules for basic regular expressions described in 2.8.3. The optional *offset* shall be a positive or negative integer value representing a number of lines. The integer value shall be preceded by + or −. 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. The pattern match of *rexp* shall be always applied from the current line to the end of the file.

%rexp%[*offset*]
: This operand is the same as */rexp/*[*offset*], except that no file shall be created for the selected section of the input file. 7

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 *rexp* 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.

5.6.5 External Influences

5.6.5.1 Standard Input

See Input Files.

5.6.5.2 Input Files

The input file shall be a text file.

5.6.5.3 Environment Variables

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

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_COLLATE This variable shall determine the locale for character collation information in regular expressions.

LC_CTYPE This variable shall determine the interpretation of sequences of bytes of text data as characters (e.g., single- versus multibyte characters in arguments and input files), and the behavior of character classes within regular expressions.

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

5.6.5.4 Asynchronous Events

If the `-k` option is specified, created files shall be retained. Otherwise the default action occurs.

5.6.6 External Effects

5.6.6.1 Standard Output

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*>

5.6.6.2 Standard Error

Used only for diagnostic messages.

5.6.6.3 Output Files

The output files shall contain portions of the original input file, otherwise unchanged.

5.6.7 Extended Description

None.

5.6.8 Exit Status

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

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

5.6.9 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.

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

Usage, 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 there 7
after, up to 9 999 lines; this is because lines in the file are numbered from 1 rather 7
than zero, for historical reasons. 7

```
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 will create a file containing each separate C routine (up to 21) in `prog.c`.

```
csplit -k prog.c '%main(%'  '/^}/+1' {20}
```

The use of repeated regular expressions with negative offsets produces bizarre results in historical implementations.

History of Decisions Made

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 file name generation used by the historical `split` utility, but the functionality added by the `−n` option was deemed to make alphabetic naming unnecessary.

5.7 `ctags` — Create a tags file

This utility shall be provided on systems that support the User Portability Utilities Option, the Software Development Utilities Option, and either or both of the C Language Development Utilities Option and FORTRAN Development Utilities Option. On other systems it is optional.

5.7.1 Synopsis

`ctags [−ax][−f` *tagsfile* `]` *pathname* ...

5.7.2 Description

The `ctags` utility shall create a *tags* file from C-language or FORTRAN source file(s) specified by the *pathname* operands. The tags file shall list the locators of language-specific objects within the source files. A locator consists of a name, pathname, and either basic regular expression or a line number that can be used in searching for the object definition. The objects that shall be recognized are specified in 5.7.7.

5.7.3 Options

The `ctags` 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 to tags file.

`−f` *tagsfile* Write the object locator lists into *tagsfile* instead of the default file named `tags` in the current directory.

`−x` Produce a list of object names, the line number and file name in which each is defined, as well as the text of that line, and write this to the standard output.