-- --

IEEE P1003.2 Draft 11.2 – September 1991

Annex C
(normative)
FORTRAN Development and Runtime Utilities Options

This annex describes utilities used for the development of FORTRAN language applications, including compilation or translation of FORTRAN source code, and the execution of certain FORTRAN applications at runtime.

The utilities described in this annex may be provided by the conforming system; however, any system claiming conformance to the FORTRAN Development Utilities Option shall provide the `fort77` utility and any system claiming conformance to the FORTRAN Runtime Utilities Option shall provide the `asa` utility.

C.0.1 FORTRAN Development and Runtime Utilities Options Rationale.
(This subclause is not a part of P1003.2)

This clause is included in this standard as a temporary measure to accommodate existing FORTRAN developers. It is the intention of the POSIX.2 working group that this annex be moved from this standard to the emerging standard being developed by the POSIX.9 working group, which will specify FORTRAN-specific interfaces to the basic services provided by this standard and POSIX.1. The movement of this annex should occur in a later version of this standard.

See the rationale for `asa` for a description of the FORTRAN Runtime Utilities Option and why it was split off from the FORTRAN Development Utilities Option.

C.1 `asa` — Interpret carriage-control characters

This utility is optional. It shall be provided on systems that support the FORTRAN Runtime Utilities Option.

C.1.1 Synopsis

`asa` [*file* . . .]

C.1.2 Description

The `asa` utility shall write its input files to standard output, mapping carriage-control characters from the text files to line-printer control sequences in an implementation-defined manner.

The first character of every line shall be removed from the input, and the following actions shall be performed:

If the character removed is:

`<space>` The rest of the line shall be output without change.

`0` A `<newline>` shall be output, then the rest of the input line.

`1` One or more implementation-defined characters that causes an advance to the next page shall be output, followed by the rest of the input line.

`+` The `<newline>` of the previous line shall be replaced with one or more implementation-defined characters that causes printing to return to column position 1, followed by the rest of the input line. If the `+` is the first character in the input, it shall have the same effect as `<space>`.

The action of the `asa` utility is unspecified upon encountering any character other than those listed above as the first character in a line.

C.1.3 Options

None.

C.1.4 Operands

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

C.1.5 External Influences

C.1.5.1 Standard Input

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

C.1.5.2 Input Files

The input files shall be text files.

C.1.5.3 Environment Variables

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

- **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.

C.1.5.4 Asynchronous Events

Default.

C.1.6 External Effects

C.1.6.1 Standard Output

The standard output shall be the text from the input file modified as described in C.1.2.

C.1.6.2 Standard Error

None.

C.1.6.3 Output Files

None.

C.1.7 Extended Description

None.

C.1.8 Exit Status

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

0 All input files were output successfully.

>0 An error occurred.

C.1.9 Consequences of Errors

Default.

C.1.10 Rationale. *(This subclause is not a part of P1003.2)*

Examples, Usage

The `asa` utility is needed to map ''standard'' FORTRAN 77 output into a form acceptable to contemporary printers. Usually `asa` is used to pipe data to the `lp` utility (see `lp` in 4.38.)

The following command:

```
asa file
```

permits the viewing of `file` (created by a program using FORTRAN -style carriage control characters) on a terminal.

The following command:

```
a.out | asa | lp
```

formats the FORTRAN output of `a.out` and directs it to the printer.

History of Decisions Made

This utility is generally used only by FORTRAN programs. It was moved to this annex in response to multiple ballot objections requesting its removal. The working group decided to retain `asa` to avoid breaking the existing large base of FORTRAN applications that put carriage control characters in their output files. This is a compromise position to achieve balloting acceptance: the overhead of maintaining a separate option in POSIX. 2 for just this one utility is seen to be small in comparison to the benefit achieved for FORTRAN applications. Since it is a separate option, there is no requirement that a system have a FORTRAN compiler in order to run applications that need `asa`.

Historical implementations have used an ASCII `<form-feed>` character in response to a `'1'`, and an ASCII `<carriage-return>` in response to a `'+'`. It is suggested that implementations treat characters other than `'0'`, `'1'`, and `'+'` as `<space>` in the absence of any compelling reason to do otherwise. However, the action is listed here as ''unspecified,'' permitting an implementation to provide extensions to access fast multiple line slewing and channel seeking in a nonportable manner.

C.2 `fort77` — FORTRAN compiler

This utility is optional. It shall be provided on systems that support the FORTRAN Development Utilities Option.

C.2.1 Synopsis

`fort77` `[−c][−g][−L` *directory* `]`... `[−O` *optlevel* `][−o` *outfile* `][−s][−w]`
operand ...

C.2.2 Description

The `fort77` utility is the interface to the FORTRAN compilation system; it shall accept the full FORTRAN language defined by ISO 1539 {2}. The system conceptually consists of a compiler and link editor. The files referenced by *operand* s are compiled and linked to produce an executable file. (It is unspecified whether the linking occurs entirely within the operation of `fort77`; some systems may produce objects that are not fully resolved until the file is executed.)

If the `−c` option is present, for all pathname operands of the form *file* `.f`, the files

`$(basename` *pathname* `.f).o`

shall be created or overwritten as the result of successful compilation. If the `−c` option is not specified, it is unspecified whether such `.o` files are created or deleted for the *file* `.f` operands.

If there are no options that prevent link editing (such as `−c`) and all operands compile and link without error, the resulting executable file shall be written into the file named by the `−o` option (if present) or to the file `a.out`. The executable file shall be created as specified in 2.9.1.4, except that the file permissions shall be set to

S_IRWXO | S_IRWXG | S_IRWXU

(see POSIX.1 {8} 5.6.1.2) and that the bits specified by the *umask* of the process shall be cleared.

C.2.3 Options

The `fort77` utility shall conform to the utility argument syntax guidelines described in 2.10.2, except that:

- The `−l` *library* operands have the format of options, but their position within a list of operands affects the order in which libraries are searched.
- The order of specifying the multiple `−L` options is significant.
- Conforming applications shall specify each option separately; that is, grouping option letters (e.g., `−cg`) need not be recognized by all implementations.