IEEE P1003.2a Draft 8 – December 1991

Copyright © 1991 by the
Institute of Electrical and Electronics Engineers, Inc.
345 East 47th Street
New York, NY 10017, USA
All rights reserved as an unpublished work.

This is an unapproved and unpublished IEEE Standards Draft, subject to change. The publication, distribution, or copying of this draft, as well as all derivative works based on this draft, is expressly prohibited except as set forth below.

Permission is hereby granted for IEEE Standards Committee participants to reproduce this document for purposes of IEEE standardization activities only, and subject to the restrictions contained herein.

Permission is hereby also granted for member bodies and technical committees of ISO and IEC to reproduce this document for purposes of developing a national position, subject to the restrictions contained herein.

Permission is hereby also granted to the preceding entities to make limited copies of this document in an electronic form only for the stated activities.

The following restrictions apply to reproducing or transmitting the document in any form: 1) all copies or portions thereof must identify the document's IEEE project number and draft number, and must be accompanied by this entire notice in a prominent location; 2) no portion of this document may be redistributed in any modified or abridged form without the prior approval of the IEEE Standards Department.

Other entities seeking permission to reproduce this document, or any portion thereof, for standardization or other activities, must contact the IEEE Standards Department for the appropriate license.

Use of information contained in this unapproved draft is at your own risk.

IEEE Standards Department
Copyright and Permissions
445 Hoes Lane, P.O. Box 1331
Piscataway, NJ 08855-1331, USA
+1 (908) 562-3800
+1 (908) 562-1571 [FAX]

## 5.11 expand — Convert tabs to spaces

### 5.11.1 Synopsis

`expand [–t` *tablist* `] [`*file* `...]`

*Obsolescent Version* :

`expand [-`*tabstop* `] [-`*tab1* `,`*tab2* `,...,`*tabn* `] [`*file* `...]`

### 5.11.2 Description

The `expand` utility shall write files or the standard input to the standard output with `<tab>` characters replaced with one or more `<space>` characters needed to pad to the next tab stop. Any `<backspace>` characters shall be copied to the output and cause the column position count for tab stop calculations to be decremented; the column position count shall not be decremented below zero.

### 5.11.3 Options

The `expand` utility shall conform to the utility argument syntax guidelines described in 2.10.2; the obsolescent version does not.

The following options shall be supported by the implementation:

`–t` *tablist* Specify the tab stops. The argument *tablist* shall consist of a single positive decimal integer or multiple positive decimal integers, separated by `<blank>`s or commas, in ascending order. If a single number is given, tabs shall be set *tablist* column positions apart instead of the default 8. If multiple numbers are given, the tabs shall be set at those specific column positions.

Each tab-stop position *N* shall be an integer value greater than zero, and the list shall be in strictly ascending order. This shall be taken to mean that, from the start of a line of output, tabbing to position *N* shall cause the next character output to be in the (*N*+1)th column position on that line.

In the event of `expand` having to process a `<tab>` character at a position beyond the last of those specified in a multiple tab-stop list, the `<tab>` shall be replaced by a single `<space>` in the output.

In the obsolescent version, the single number is specified as *tabstop* with a leading minus; multiple tab stops are specified after a leading minus as *tab1* , *tab2* , etc.

Copyright © 1991 IEEE. All rights reserved.
This is an unapproved IEEE Standards Draft, subject to change.

### 5.11.4 Operands

The following operand shall be supported by the implementation:

*file* The pathname of a text file to be used as input.

### 5.11.5 External Influences

#### 5.11.5.1 Standard Input

See Input Files.

#### 5.11.5.2 Input Files

Input files shall be text files.

#### 5.11.5.3 Environment Variables

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

**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 interpretation of sequences of bytes of text data as characters (e.g., single- versus multibyte characters in arguments and input files), the processing of `<tab>`s and `<space>`s, and for the determination of the width in column positions each character would occupy on a constant-width-font output device.

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

#### 5.11.5.4 Asynchronous Events

Default.

Copyright © 1991 IEEE. All rights reserved.
This is an unapproved IEEE Standards Draft, subject to change.

### 5.11.6 External Effects

#### 5.11.6.1 Standard Output

The standard output shall be equivalent to the input files with <tab> characters converted into <space>s, as specified by 5.11.2.

#### 5.11.6.2 Standard Error

Used only for diagnostic messages.

#### 5.11.6.3 Output Files

None.

### 5.11.7 Extended Description

None.

### 5.11.8 Exit Status

The expand utility shall exit with one of the following values:

0 Successful completion.

>0 An error occurred.

### 5.11.9 Consequences of Errors

The expand utility shall terminate with an error message and nonzero exit status upon encountering difficulties accessing one of the *file* operands.

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

**Usage, Examples**

The expand utility is useful for preprocessing text files (before sorting, looking at specific columns, etc.) that contain <tab>s.

See the definition of *column position* in 2.2.2.31.

The *tablist* option-argument consists of integers in ascending order. Utility Syntax Guideline 8 (see 2.10.2) mandates that expand shall accept the integers (within the single argument) separated using either commas or <blank>s.

Copyright © 1991 IEEE. All rights reserved.
This is an unapproved IEEE Standards Draft, subject to change.

### History of Decisions Made

None.

## 5.12 fc — Process command history list

### 5.12.1 Synopsis

```
fc [−e editor ] [first [last ]]
fc −l [−nr] [first [last ]]
fc −s [old=new ] [first ]
```

### 5.12.2 Description

The fc utility shall list, or shall edit and re-execute, commands previously entered to an interactive sh (see 4.56).

The command history list shall reference commands by number. The first number in the list is selected arbitrarily. The relationship of a number to its command shall not change except when the user logs in and no other process is accessing the list, at which time the system may reset the numbering to start the oldest retained command at another number (usually 1). When the number reaches an implementation-defined upper limit, which shall be no smaller than the value in **HISTSIZE** or 32 767 (whichever is greater), the shell may wrap the numbers, starting the next command with a lower number (usually 1). However, despite this optional wrapping of numbers, fc shall maintain the time-ordering sequence of the commands. For example, if four commands in sequence are given the numbers 32 766, 32 767, 1 (wrapped), and 2 as they are executed, command 32 767 is considered the command previous to 1, even though its number is higher.

When commands are edited (when the −l option is not specified), the resulting line(s) shall be entered at the end of the history list and then re-executed by sh. The fc command that caused the editing shall not be entered into the history list. If the editor returns a nonzero exit status, this shall suppress the entry into the history list and the command re-execution. Any command line variable assignments or redirection operators used with fc shall affect both the fc command itself as well as the command that results. For example,

```
fc -s -1 2>/dev/null
```

reinvokes the previous command, suppressing standard error for both fc and the previous command.

Copyright © 1991 IEEE. All rights reserved.
This is an unapproved IEEE Standards Draft, subject to change.