IEEE P1003.2 Draft 11.2 – September 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All rights reserved as an unpublished work.

This is an unapproved and unpublished IEEE Standards Draft, subject to change.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]

4255 `count=`*n* Copy only *n* input blocks.

4256 `conv=`*value* [`,`*value* ...]
4257 Where *value* s are comma-separated symbols from the follow-
4258 ing list.

4259 `block` Treat the input as a sequence of <newline>-terminated or 2
4260 end-of-file-terminated variable length records independent 2
4261 of the input block boundaries. Each record shall be con-
4262 verted to a record with a fixed length specified by the
4263 conversion block size. Any <newline> shall be removed 2
4264 from the input line; <space>s shall be appended to lines
4265 that are shorter than their conversion block size to fill the
4266 block. Lines that are longer than the conversion block size
4267 shall be truncated to the largest number of characters that
4268 will fit into that size; the number of truncated lines shall be
4269 reported (see Standard Error below).

4270 The `block` and `unblock` values are mutually exclusive.

4271 `unblock` Convert fixed length records to variable length. Read a
4272 number of bytes equal to the conversion block size, delete all
4273 trailing <space>s, and append a <newline>. 2

4274 `lcase` Map uppercase characters specified by the LC_CTYPE key-
4275 word `tolower` to the corresponding lowercase character.
4276 Characters for which no mapping is specified shall not be
4277 modified by this conversion.

4278 The `lcase` and `ucase` symbols are mutually exclusive.

4279 `ucase` Map lowercase characters specified by the LC_CTYPE key-
4280 word `toupper` to the corresponding uppercase character.
4281 Characters for which no mapping is specified shall not be
4282 modified by this conversion.

4283 `swab` Swap every pair of input bytes.

4284 `noerror` Do not stop processing on an input error. When an input
4285 error occurs, a diagnostic message shall be written on stan-
4286 dard error, followed by the current input and output block
4287 counts in the same format as used at completion (see Stan-
4288 dard Error). If the `sync` conversion is specified, the missing
4289 input shall be replaced with null bytes and processed nor-
4290 mally; otherwise, the input block shall be omitted from the
4291 output.

4292 `notrunc` Do not truncate the output file. Preserve blocks in the out-
4293 put file not explicitly written by this invocation of the `dd`
4294 utility. (See also the preceding `of=`*file* operand.)

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

`sync` Pad every input block to the size of `ibs=` buffer, appending null bytes.

The behavior is unspecified if operands other than `conv=` are specified more than once.

For the `bs=`, `cbs=`, `ibs=`, and `obs=` operands, the application shall supply an expression specifying a size in bytes. The expression, *expr*, can be:

(1) a positive decimal number;

(2) a positive decimal number followed by `k`, specifying multiplication by 1024;

(3) a positive decimal number followed by `b`, specifying multiplication by 512; or

(4) two or more positive decimal numbers (with or without `k` or `b`) separated by `x`, specifying the product of the indicated values.

4.16.5 External Influences

4.16.5.1 Standard Input

If no `if=` operand is specified, the standard input shall be used. See Input Files.

4.16.5.2 Input Files

The input file can be any file type.

4.16.5.3 Environment Variables

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

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), the classification of characters as upper- or lowercase, and the mapping of characters from one case to the other.

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

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

4.16.5.4 Asynchronous Events

For SIGINT , the `dd` utility shall write status information to standard error before exiting. It shall take the standard action for all other signals; see 2.11.5.4.

4.16.6 External Effects

4.16.6.1 Standard Output

If no `of=` operand is specified, the standard output shall be used. The nature of the output depends on the operands selected.

4.16.6.2 Standard Error

On completion, `dd` shall write the number of input and output blocks to standard error. In the POSIX Locale the following formats shall be used:

`"%u+%u records in\n"`, *<number of whole input blocks>* , *<number of partial input blocks>*

`"%u+%u records out\n"`, *<number of whole output blocks>* , *<number of partial output blocks>*

A partial input block is one for which *read* () returned less than the input block size. A partial output block is one that was written with fewer bytes than specified by the output block size.

In addition, when there is at least one truncated block, the number of truncated blocks shall be written to standard error. In the POSIX Locale, the format shall be:

`"%u truncated %s\n"`, *<number of truncated blocks>* , `"block"` [if *<number of truncated blocks>* is one] `"blocks"` [otherwise]

Diagnostic messages may also be written to standard error.

4.16.6.3 Output Files

If the `of=` operand is used, the output shall be the same as described in Standard Output.

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

4.16.7 Extended Description

None.

4.16.8 Exit Status

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

0 The input file was copied successfully.

>0 An error occurred.

4.16.9 Consequences of Errors

If an input error is detected and the `noerror` conversion has not been specified, any partial output block shall be written to the output file, a diagnostic message shall be written, and the copy operation shall be discontinued. If some other error is detected, a diagnostic message shall be written and the copy operation shall be discontinued.

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

Examples, Usage

The input and output block size can be specified to take advantage of raw physical I/O.

The following command:

```
dd if=/dev/rmt0h of=/dev/rmt1h
```

copies from tape drive 0 to tape drive 1, using a common historical device naming convention.

The following command:

```
dd ibs=10 skip=1
```

strips the first 10 bytes from standard input.

A suggested implementation technique for `conv=noerror,sync` is to zero the input buffer before each read and to write the contents of the input buffer to the output even after an error. In this manner, any data transferred to the input buffer before the error was detected will be preserved. Another point is that a failed read on a regular file or a disk will generally not increment the file offset, and `dd` must then seek past the block on which the error occurred; otherwise, the input error will occur repetitively. When the input is a magnetic tape, however, the tape will normally have passed the block containing the error when the error is reported, and thus no seek is necessary.

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

History of Decisions Made

Table 4-4 – ASCII to EBCDIC Conversion

	0	1	2	3	4	5	6	7
0000	0000	0001	0002	0003	0067	0055	0056	0057
0010	0026	0005	0045	0013	0014	0015	0016	0017
0020	0020	0021	0022	0023	0074	0075	0062	0046
0030	0030	0031	0077	0047	0034	0035	0036	0037
0040	0100	0132	0177	0173	0133	0154	0120	0175
0050	0115	0135	0134	0116	0153	0140	0113	0141
0060	0360	0361	0362	0363	0364	0365	0366	0367
0070	0370	0371	0172	0136	0114	0176	0156	0157
0100	0174	0301	0302	0303	0304	0305	0306	0307
0110	0310	0311	0321	0322	0323	0324	0325	0326
0120	0327	0330	0331	0342	0343	0344	0345	0346
0130	0347	0350	0351	0255	0340	0275	0232	0155
0140	0171	0201	0202	0203	0204	0205	0206	0207
0150	0210	0211	0221	0222	0223	0224	0225	0226
0160	0227	0230	0231	0242	0243	0244	0245	0246
0170	0247	0250	0251	0300	0117	0320	0137	0007
0200	0040	0041	0042	0043	0044	0025	0006	0027
0210	0050	0051	0052	0053	0054	0011	0012	0033
0220	0060	0061	0032	0063	0064	0065	0066	0010
0230	0070	0071	0072	0073	0004	0024	0076	0341
0240	0101	0102	0103	0104	0105	0106	0107	0110
0250	0111	0121	0122	0123	0124	0125	0126	0127
0260	0130	0131	0142	0143	0144	0145	0146	0147
0270	0150	0151	0160	0161	0162	0163	0164	0165
0300	0166	0167	0170	0200	0212	0213	0214	0215
0310	0216	0217	0220	0152	0233	0234	0235	0236
0320	0237	0240	0252	0253	0254	0112	0256	0257
0330	0260	0261	0262	0263	0264	0265	0266	0267
0340	0270	0271	0272	0273	0274	0241	0276	0277
0350	0312	0313	0314	0315	0316	0317	0332	0333
0360	0334	0335	0336	0337	0352	0353	0354	0355
0370	0356	0357	0372	0373	0374	0375	0376	0377

The Options subclause is listed as ''None'' because there are no options recognized by historical **dd** utilities. Certainly, many of the operands could have been designed to use the Utility Syntax Guidelines, which would have resulted in the classic hyphenated option letters. In this version of this standard, **dd** retains its curious JCL-like syntax due to the large number of applications that depend on the historical implementation. ''Fixing'' the interface would cause an excessive compatibility problem. However, due to interest in the international community, the developers of the standard have agreed to provide an alternative syntax for

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

Table 4-5 – ASCII to IBM EBCDIC Conversion

	0	1	2	3	4	5	6	7
0000	0000	0001	0002	0003	0067	0055	0056	0057
0010	0026	0005	0045	0013	0014	0015	0016	0017
0020	0020	0021	0022	0023	0074	0075	0062	0046
0030	0030	0031	0077	0047	0034	0035	0036	0037
0040	0100	0132	0177	0173	0133	0154	0120	0175
0050	0115	0135	0134	0116	0153	0140	0113	0141
0060	0360	0361	0362	0363	0364	0365	0366	0367
0070	0370	0371	0172	0136	0114	0176	0156	0157
0100	0174	0301	0302	0303	0304	0305	0306	0307
0110	0310	0311	0321	0322	0323	0324	0325	0326
0120	0327	0330	0331	0342	0343	0344	0345	0346
0130	0347	0350	0351	0255	0340	0275	0137	0155
0140	0171	0201	0202	0203	0204	0205	0206	0207
0150	0210	0211	0221	0222	0223	0224	0225	0226
0160	0227	0230	0231	0242	0243	0244	0245	0246
0170	0247	0250	0251	0300	0117	0320	0241	0007
0200	0040	0041	0042	0043	0044	0025	0006	0027
0210	0050	0051	0052	0053	0054	0011	0012	0033
0220	0060	0061	0032	0063	0064	0065	0066	0010
0230	0070	0071	0072	0073	0004	0024	0076	0341
0240	0101	0102	0103	0104	0105	0106	0107	0110
0250	0111	0121	0122	0123	0124	0125	0126	0127
0260	0130	0131	0142	0143	0144	0145	0146	0147
0270	0150	0151	0160	0161	0162	0163	0164	0165
0300	0166	0167	0170	0200	0212	0213	0214	0215
0310	0216	0217	0220	0232	0233	0234	0235	0236
0320	0237	0240	0252	0253	0254	0255	0256	0257
0330	0260	0261	0262	0263	0264	0265	0266	0267
0340	0270	0271	0272	0273	0274	0275	0276	0277
0350	0312	0313	0314	0315	0316	0317	0332	0333
0360	0334	0335	0336	0337	0352	0353	0354	0355
0370	0356	0357	0372	0373	0374	0375	0376	0377

the next version of this standard that conforms to the spirit of the Utility Syntax Guidelines. This new syntax will be accompanied by the existing syntax, marked as obsolescent. System implementors are encouraged to develop and promulgate a new syntax for `dd`, perhaps using a different utility name, that can be adopted for the next version of this standard.

The default `ibs=` and `obs=` sizes are specified as 512 bytes because there are existing (largely portable) scripts that assume these values. If they were left unspecified, very strange results could occur if an implementation chose an odd block size.

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

Historical implementations of `dd` used *creat* () when processing `of=file`. This makes the `seek=` operand unusable except on special files. More recent BSD-based implementations use *open* () (without O_TRUNC) instead of *creat* (), but fail to delete output file contents after the data copied. Since balloting showed a desire to make this behavior available, the `conv=notrunc` feature was added.

The `w` multiplier, (historically meaning *word*), is used in System V to mean 2 and in 4.2 BSD to mean 4. Since *word* is inherently nonportable, its use is not supported by POSIX. 2.

All references to US ASCII and to conversions to/from IBM and EBCDIC were removed in preparation for this document's acceptance by the international community. Implementations are free to have such conversions as extensions, using the `ascii`, `ibm`, and `ebcdic` keywords. However, in the interest of promoting consistency of implementation, the original material from an early draft has been restored to the rationale as an example:

> In the two tables, the conversions from ASCII to either standard EBCDIC (Table 4-4) or the IBM version of EBCDIC (Table 4-5) are shown. The differences between the two tables are underlined. In both tables, the ASCII values are the row and column headers and the EBCDIC values are found at their intersections. For example, ASCII 0012 (LF) is the second row, third column, yielding 0045 in EBCDIC. The inverted tables (for EBCDIC to ASCII conversion) are not shown, but are in one-to-one correspondence with these tables. The tables are understood to match recent System V conversion algorithms and there have been reports that earlier System V versions and the BSD version do not always conform to these; however, representatives of the BSD development group have agreed that a future version of their system will use these tables for consistency with System V.
>
> The `cbs` operand is required if any of the `ascii`, `ebcdic`, or `ibm` operands are specified. For the `ascii` operand, the input is handled as described for the `unblock` operand except that characters are converted to ASCII before the trailing `<spaces>`s are deleted. For the `ebcdic` and `ibm` operands, the input is handled as described for the `block` operand except that the characters are converted to EBCDIC or IBM EBCDIC after the trailing `<spaces>`s are added.

The `block` and `unblock` keywords are from historical BSD practice.

Early drafts only allowed two numbers separated by `x` to be used in a product when specifying `bs=`, `cbs=`, `ibs=`, and `obs=` sizes. This was changed to reflect the historical practice of allowing multiple numbers in the product as provided by Version 7 and all releases of System V and BSD.

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

4.17 `diff` — Compare two files

4.17.1 Synopsis

`diff [ -c | -e | -C` *n* `][-br]` *file1 file2*

4.17.2 Description

The `diff` utility shall compare the contents of *file1* and *file2* and write to standard output a list of changes necessary to convert *file1* into *file2*. This list should be minimal. No output shall be produced if the files are identical.

4.17.3 Options

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

The following options shall be supported by the implementation:

`-b` Cause trailing `<blank>`s to be ignored and other strings of `<blank>`s to compare equal.

`-c` Produce output in a form that provides three lines of context.

`-C` *n* Produce output in a form that provides *n* lines of context (where *n* shall be interpreted as a positive decimal integer).

`-e` Produce output in a form suitable as input for the `ed` utility (see 4.20), which can then be used to convert *file1* into *file2*.

`-r` Apply `diff` recursively to files and directories of the same name when *file1* and *file2* are both directories.

4.17.4 Operands

The following operands shall be supported by the implementation:

file1
file2 A pathname of a file be compared. If either the *file1* or *file2* operand is -, the standard input shall be used in its place.

If both *file1* and *file2* are directories, `diff` shall not compare block special files, character special files, or FIFO special files to any files and shall not compare regular files to directories. The system documentation shall specify the behavior of `diff` on implementation-specific file types not specified by POSIX.1 {8} when found in directories. Further details are as specified in 4.17.6.1.1.

If only one of *file1* and *file2* is a directory, `diff` shall be applied to the nondirectory file and the file contained in the directory file with a filename that is the same as the last component of the nondirectory file.

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

4550 ### 4.17.5 External Influences

4551 #### 4.17.5.1 Standard Input

4552 The standard input shall be used only if one of the *file1* or *file2* operands refer-
4553 ences standard input. See Input Files.

4554 #### 4.17.5.2 Input Files

4555 The input files shall be text files.

4556 #### 4.17.5.3 Environment Variables

4557 The following environment variables shall affect the execution of `diff`:

4558 **LANG** This variable shall determine the locale to use for the
4559 locale categories when both **LC_ALL** and the correspond-
4560 ing environment variable (beginning with **LC_**) do not
4561 specify a locale. See 2.6.

4562 **LC_ALL** This variable shall determine the locale to be used to over-
4563 ride any values for locale categories specified by the set-
4564 tings of **LANG** or any environment variables beginning
4565 with **LC_**.

4566 **LC_CTYPE** This variable shall determine the locale for the interpreta-
4567 tion of sequences of bytes of text data as characters (e.g.,
4568 single- versus multibyte characters in arguments and
4569 input files).

4570 **LC_MESSAGES** This variable shall determine the language in which mes-
4571 sages should be written.

4572 **LC_TIME** This variable shall determine the locale for affecting the
4573 format of file time stamps written with the `-C` and `-c`
4574 options.

4575 **TZ** This variable shall determine the locale for affecting the
4576 time zone used for calculating file time stamps written
4577 with the `-C` and `-c` options.

4578 #### 4.17.5.4 Asynchronous Events

4579 Default.

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

4.17.6 External Effects

4.17.6.1 Standard Output

4.17.6.1.1 `diff` Directory Comparison Format

If both *file1* and *file2* are directories, the following output formats shall be used.

In the POSIX Locale, each file that is present in only one directory shall be reported using the following format:

"`Only in %s: %s\n`", <*directory pathname*> , <*filename*>

In the POSIX Locale, subdirectories that are common to the two directories may be reported with the following format:

"`Common subdirectories: %s and %s\n`", <*directory1 pathname*> , <*directory2 pathname*>

For each file common to the two directories if the two files are not to be compared, the following format shall be used in the POSIX Locale:

"`File %s is a %s while file %s is a %s\n`", <*directory1 pathname*> , <*file type of directory1 pathname*> , <*directory2 pathname*> , <*file type of directory2 pathname*>

For each file common to the two directories, if the files are to be compared and are identical, no output shall be written. If the two files differ, the following format shall be written:

"`diff %s %s %s\n`", <*diff_options*> , <*filename1*> , <*filename2*>

where <*diff_options*> are the options as specified on the command line. Depending on these options, one of the following output formats shall be used to write the differences.

All directory pathnames listed in this subclause shall be relative to the original command line arguments. All other names of files listed in this subclause shall be filenames (pathname components).

4.17.6.1.2 `diff` Default Output Format

The default (without `-e`, `-c`, or `-C` options) `diff` utility output contains lines of these forms:

"`%da%d\n`", <*num1*> , <*num2*>

"`%da%d,%d\n`", <*num1*> , <*num2*> , <*num3*>

"`%dd%d\n`", <*num1*> , <*num2*>

"`%d,%dd%d\n`", <*num1*> , <*num2*> , <*num3*>

"`%dc%d\n`", <*num1*> , <*num2*>

"`%d,%dc%d\n`", <*num1*> , <*num2*> , <*num3*>

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

4615 "%dc%d,%d\n", <*num1*> , <*num2*> , <*num3*>

4616 "%d,%dc%d,%d\n", <*num1*> , <*num2*> , <*num3*> , <*num4*>

4617 These lines resemble `ed` subcommands to convert *file1* into *file2*. The line
4618 numbers before the action letters shall pertain to *file1*; those after shall pertain to
4619 *file2*. Thus, by exchanging `'a'` for `'d'` and reading the line in reverse order, one
4620 can also determine how to convert *file2* into *file1*. As in `ed`, identical pairs (where
4621 *num1* = *num2*) are abbreviated as a single number.

4622 Following each of these lines, `diff` shall write to standard output all lines
4623 affected in the first file using the format:

4624 "<Δ%s", <*line*>

4625 and all lines affected in the second file using the format:

4626 ">Δ%s", <*line*>

4627 If there are lines affected in both *file1* and *file2* (as with the `c` subcommand), the
4628 changes are separated with a line consisting of three hyphens:

4629 "---\n"

4630 #### 4.17.6.1.3 `diff` –`e` Output Format

4631 With the –`e` option, a script shall be produced that shall, when provided as input
4632 to `ed` (see 4.20), along with an appended `w` (write) command, convert *file1* into
4633 *file2*. Only the `a` (append), `c` (change), `d` (delete), `i` (insert), and `s` (substitute) com-
4634 mands of `ed` shall be used in this script. Text line(s), except those consisting of
4635 the single character period (`.`), shall be output as they appear in the file.

4636 #### 4.17.6.1.4 `diff` –`c` or –`C` Output Format

4637 With the –`c` or –`C` option, the output format shall consist of affected lines along
4638 with surrounding lines of context. The affected lines shall show which ones need
4639 to be deleted or changed in *file1*, and those added from *file2*. With the –`c` option,
4640 three lines of context, if available, shall be written before and after the affected
4641 lines. With the –`C` option, the user can specify how many lines of context shall be
4642 written. The exact format follows.

4643 The name and last modification time of each file shall be output in the following
4644 format:

4645 "*** %s %s\n", *file1*, <*file1 time stamp*>
4646 "--- %s %s\n", *file2*, <*file2 time stamp*>

4647 and a string of 15 asterisks:

4648 "***************\n"

4649 Each <*file*> field shall be the pathname of the corresponding file being compared.
4650 The pathname written for standard input is unspecified.

4651 In the POSIX Locale, each <*time stamp*> field shall be equivalent to the output
4652 from the following command:

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

4653 `date "+%a %b %e %T %Y"`

4654 without the trailing <newline>, executed at the time of last modification of the
4655 corresponding file (or the current time, if the file is standard input).

4656 Then, the following output formats shall be applied for every set of changes.

4657 First, the range of lines in *file1* shall be written in the following format:

4658 `"*** %d,%d ****\n"`, *<beginning line number>* , *<ending line number>*

4659 Next, the affected lines along with lines of context (unaffected lines) shall be writ-
4660 ten. Unaffected lines shall be written in the following format:

4661 `"ΔΔ%s"`, *<unaffected_line>*

4662 Deleted lines shall be written as:

4663 `"-Δ%s"`, *<deleted_line>*

4664 Changed lines shall be written as:

4665 `"!Δ%s"`, *<changed_line>*

4666 Next, the range of lines in *file2* shall be written in the following format:

4667 `"--- %d,%d ----\n"`, *<beginning line number>* , *<ending line number>*

4668 Then, lines of context and changed lines shall be written as described in the previ-
4669 ous formats. Lines added from *file2* shall be written in the following format:

4670 `"+Δ%s"`, *<added_line>*

4671 #### 4.17.6.2 Standard Error

4672 Used only for diagnostic messages.

4673 #### 4.17.6.3 Output Files

4674 None.

4675 ### 4.17.7 Extended Description

4676 None.

4677 ### 4.17.8 Exit Status

4678 The `diff` utility shall exit with one of the following values:

4679 0 No differences were found.

4680 1 Differences were found.

4681 >1 An error occurred.

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

4.17.9 Consequences of Errors

Default.

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

Examples, Usage

If lines at the end of a file are changed and other lines are added, `diff` output may show this as a delete and add, as a change, or as a change and add; `diff` is not expected to know which happened and users should not care about the difference in output as long as it clearly shows the differences between the files.

If `dir1` is a directory containing a directory named `x`, `dir2` is a directory containing a directory named `x`, `dir1/x` and `dir2/x` both contain files named `date.out`, and `dir2/x` contains a file named `y`, the command:

```
diff -r dir1 dir2
```

could produce output similar to:

```
Common subdirectories: dir1/x and dir2/x
Only in dir2/x: y
diff -r dir1/x/date.out dir2/x/date.out
1c1
< Mon Jul  2 13:12:16 PDT 1990
---
> Tue Jun 19 21:41:39 PDT 1990
```

History of Decisions Made

The `-h` option was removed because it was insufficiently specified and it does not add to application portability.

Current implementations employ algorithms that do not always produce a minimum list of differences; the current language about making every effort is the best the standard can do, as there is no metric that could be employed to judge the quality of implementations against any and all file contents. The statement ''This list should be minimal'' clearly implies that implementations are not expected to provide the following output when comparing two 100-line files that differ in only one character on a single line:

```
1,100c1,100
all 100 lines from file1 preceded with "< "
---
all 100 lines from file2 preceded with "> "
```

The ''Only in'' messages required by this standard when the `-r` option is specified, is not used by most historical implementations if the `-e` option is also specified. It is required here because it provides useful information that must be provided to update a target directory hierarchy to match a source hierarchy. The ''Common subdirectories'' messages are written by System V and 4.3 BSD when the `-r` option is specified. They are allowed here, but are not required because they are

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

reporting on something that is the same, not reporting a difference, and are not needed to update a target hierarchy.

The −c option, which writes output in a format using lines of context, has been included. The format is useful for a variety of reasons, among them being much improved readability, and the ability to understand difference changes when the target file has line numbers that differ from another similar, but slightly different, copy. An important utility, patch, which has proved itself indispensable to the USENET community, often only works with difference listings using the context format. The BSD version of −c takes an optional argument specifying the amount of context. Rather than overloading −c and breaking the Utility Syntax Guidelines for diff, the working group decided to add a separate option for specifying a context diff with a specified amount of context (−C). Also, the format for context diffs was extended slightly in 4.3 BSD to allow multiple changes that are within context lines from each other to be merged together. The output format contains an additional four asterisks after the range of affected lines in the first filename. This was to provide a flag for old programs (like old versions of patch) that only understand the old context format. The version of context described here does not require that multiple changes within context lines be merged, but does not prohibit it either. The extension is upward compatible, so any vendors that wish to retain the old version of diff can do so by just adding the extra four asterisks (that is, utilities that currently use diff and understand the new merged format will also understand the old unmerged format, but not vice-versa).

The substitute command was added as an additional format for the −e option. This was added to provide implementations a way to fix the classic ‘‘dot alone on a line’’ bug present in many versions of diff. Since many implementations have fixed this bug the working group decided not to standardize broken behavior, but rather, provide the necessary tool for fixing the bug. One way to fix this bug is to output two periods whenever a lone period is needed, then terminate the append command with a period, and then use the substitute command to convert the two periods into one period.

The −f flag was not included as it provides no additional functionality over the −e option.

The BSD-derived −r option was added to provide a mechanism for using diff to compare two file system trees. This behavior is useful, is standard practice on all BSD-derived systems, and is not easily reproducible with the find utility.

The requirement that diff not compare files in some circumstances, even though they have the same name, was added in response to ballot objections and digging further into the actual output of historical implementations. The message specified here is already in use when a directory is being compared to a nondirectory. It is extended here to preclude the problems arising from running into FIFOs and other files that would cause diff to hang waiting for input with no indication to the user that diff was hung. In most common usage, diff −r should indicate differences in the file hierarchies, not the difference of contents of devices pointed to by the hierarchies.

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

Many early implementations of `diff` require seekable files. Since POSIX.1 {8} supports named pipes, the working group decided that such a restriction was unreasonable. Note also that the allowed file name – almost always refers to a pipe.

No directory search order is being specified in 4.17.6.1.1. The historical ordering is, in fact, not optimal, in that it prints out all of the differences at the current level, including the statements about all common subdirectories before recursing into those subdirectories.

The message

> `"diff %s %s %s\n"`, <*diff_options*> , <*filename1*> , <*filename2*>

does not vary by locale because it is the representation of a command, not an English sentence.

4.18 `dirname` — Return directory portion of pathname

4.18.1 Synopsis

`dirname` *string*

4.18.2 Description

The *string* operand shall be treated as a pathname, as defined in 2.2.2.102. The string *string* shall be converted to the name of the directory containing the filename corresponding to the last pathname component in *string* , performing actions equivalent to the following steps in order:

(1) If *string* is `//`, skip steps (2) through (5).

(2) If *string* consists entirely of slash characters, *string* shall be set to a single slash character. In this case, skip steps (3) through (8).

(3) If there are any trailing slash characters in *string* , they shall be removed.

(4) If there are no slash characters remaining in *string* , *string* shall be set to a single period character. In this case, skip steps (5) through (8).

(5) If there are any trailing nonslash characters in *string* , they shall be removed.

(6) If the remaining *string* is `//`, it is implementation defined whether steps (7) and (8) are skipped or processed.

(7) If there are any trailing slash characters in *string* , they shall be removed.

(8) If the remaining *string* is empty, *string* shall be set to a single slash character.

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

The resulting string shall be written to standard output.

4.18.3 Options

None.

4.18.4 Operands

The following operand shall be supported by the implementation:

string A string.

4.18.5 External Influences

4.18.5.1 Standard Input

None.

4.18.5.2 Input Files

None.

4.18.5.3 Environment Variables

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

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).

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

4.18.5.4 Asynchronous Events

Default.

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

4827 ### 4.18.6 External Effects

4828 #### 4.18.6.1 Standard Output

4829 The `dirname` utility shall write a line to the standard output in the following for-
4830 mat:

4831 `"%s\n"`, *<resulting string>*

4832 #### 4.18.6.2 Standard Error

4833 Used only for diagnostic messages.

4834 #### 4.18.6.3 Output Files

4835 None.

4836 ### 4.18.7 Extended Description

4837 None.

4838 ### 4.18.8 Exit Status

4839 The `dirname` utility shall exit with one of the following values:

4840 0 Successful completion.

4841 >0 An error occurred.

4842 ### 4.18.9 Consequences of Errors

4843 Default.

4844 ### 4.18.10 Rationale. *(This subclause is not a part of P1003.2)*

4845 **Examples, Usage**

4846 The `dirname` utility originated in System III. It has evolved through the
4847 System V releases to a version that matches the requirements specified in this
4848 description in System V Release 3.

4849 4.3 BSD and earlier versions did not include `dirname`.

4850 Table 4-6 indicates the results required for some invocations of `dirname`.

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

Table 4-6 – `dirname` Examples

Command	Results
`dirname /`	`/`
`dirname //`	`/` or `//`
`dirname /a/b/`	`/a`
`dirname //a//b//`	`//a`
`dirname`	*unspecified*
`dirname a`	`. ($? = 0)`
`dirname ""`	`. ($? = 0)`
`dirname /a`	`/`
`dirname /a/b`	`/a`
`dirname a/b`	`a`

History of Decisions Made

The behaviors of `basename` and `dirname` in this standard have been coordinated so that when *string* is a valid pathname

```
$(basename "string")
```

would be a valid filename for the file in the directory

```
$(dirname "string")
```

This would not work for the versions of these utilities in earlier drafts due to the way processing of trailing slashes was specified. Consideration was given to leaving processing unspecified if there were trailing slashes, but this cannot be done; the POSIX.1 {8} definition of pathname allows trailing slashes. The `basename` and `dirname` utilities have to specify consistent handling for all valid pathnames.

Since the definition of *pathname* in 2.2.2.102 specifies implementation-defined behavior for pathnames starting with two slash characters, Draft 11 has been changed to specify similar implementation-defined behavior for the `basename` and `dirname` utilities. On implementations where the pathname `//` is always treated the same as the pathname `/`, the functionality required by Draft 10 meets all of the Draft 11 requirements.

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

4.19 `echo` — Write arguments to standard output

4.19.1 Synopsis

```
echo [string ...]
```

4.19.2 Description

The `echo` utility shall write its arguments to standard output, followed by a `<newline>` character. If there are no arguments, only the `<newline>` character shall be written.

4.19.3 Options

The `echo` utility shall not recognize the `--` argument in the manner specified by utility syntax guideline 10 in 2.10.2; `--` shall be recognized as a string operand.

Implementations need not support any options.

4.19.4 Operands

The following operands shall be supported by the implementation:

string A string to be written to standard output. If the first operand is `"-n"` or if any of the operands contain a backslash (\) character, the results are implementation defined.

4.19.5 External Influences

4.19.5.1 Standard Input

None.

4.19.5.2 Input Files

None.

4.19.5.3 Environment Variables

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

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.

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

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

4.19.5.4 Asynchronous Events

Default.

4.19.6 External Effects

4.19.6.1 Standard Output

The `echo` utility arguments shall be separated by single `<space>`s and a `<newline>` character shall follow the last argument.

4.19.6.2 Standard Error

Used only for diagnostic messages.

4.19.6.3 Output Files

None.

4.19.7 Extended Description

None.

4.19.8 Exit Status

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

0 Successful completion.

>0 An error occurred.

4.19.9 Consequences of Errors

Default.

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