

IEEE Std 1003.1™, 2004 Edition

**The Open Group Technical Standard
Base Specifications, Issue 6**

**Includes IEEE Std 1003.1™-2001, IEEE Std 1003.1™-2001/Cor 1-2002
and IEEE Std 1003.1™-2001/Cor 2-2004**

Standard for Information Technology— Portable Operating System Interface (POSIX®)

Shell and Utilities

Sponsor

**Portable Applications Standards Committee
of the
IEEE Computer Society**

and

The Open Group



THE *Open* GROUP

[This page intentionally left blank]

Abstract

This standard is simultaneously ISO/IEC 9945, IEEE Std 1003.1, and forms the core of the Single UNIX Specification, Version 3.

This 2004 Edition includes IEEE Std 1003.1-2001/Cor 1-2002 and IEEE Std 1003.1-2001/Cor 2-2004 incorporated into IEEE Std 1003.1-2001 (the base document). The two Corrigenda address problems discovered since the approval of IEEE Std 1003.1-2001. These changes are mainly due to resolving integration issues raised by the merger of the base documents that were incorporated into IEEE Std 1003.1-2001, which is the single common revision to IEEE Std 1003.1TM-1996, IEEE Std 1003.2TM-1992, ISO/IEC 9945-1:1996, ISO/IEC 9945-2:1993, and the Base Specifications of The Open Group Single UNIX[®] Specification, Version 2.

This standard defines a standard operating system interface and environment, including a command interpreter (or "shell"), and common utility programs to support applications portability at the source code level. This standard is intended to be used by both applications developers and system implementors and comprises four major components (each in an associated volume):

- General terms, concepts, and interfaces common to all volumes of this standard, including utility conventions and C-language header definitions, are included in the Base Definitions volume.
- Definitions for system service functions and subroutines, language-specific system services for the C programming language, function issues, including portability, error handling, and error recovery, are included in the System Interfaces volume.
- Definitions for a standard source code-level interface to command interpretation services (a "shell") and common utility programs for application programs are included in the Shell and Utilities volume.
- Extended rationale that did not fit well into the rest of the document structure, which contains historical information concerning the contents of this standard and why features were included or discarded by the standard developers, is included in the Rationale (Informative) volume.

The following areas are outside the scope of this standard:

- Graphics interfaces
- Database management system interfaces
- Record I/O considerations
- Object or binary code portability
- System configuration and resource availability

This standard describes the external characteristics and facilities that are of importance to applications developers, rather than the internal construction techniques employed to achieve these capabilities. Special emphasis is placed on those functions and facilities that are needed in a wide variety of commercial applications.

Keywords

application program interface (API), argument, asynchronous, basic regular expression (BRE), batch job, batch system, built-in utility, byte, child, command language interpreter, CPU, extended regular expression (ERE), FIFO, file access control mechanism, input/output (I/O), job control, network, portable operating system interface (POSIX[®]), parent, shell, stream, string, synchronous, system, thread, X/Open System Interface (XSI)

Copyright © 2001-2004 by the Institute of Electrical and Electronics Engineers, Inc. and The Open Group. All rights reserved.

Shell and Utilities, Issue 6

Published 30 April 2004 by the Institute of Electrical and Electronics Engineers, Inc.

3 Park Avenue, New York, NY 10016-5997, U.S.A.

ISBN: 0-7381-4043-0 / SH95236 PDF 0-7381-4044-9 / SS95236

Printed in the United States of America by the IEEE.

Published 30 April 2004 by The Open Group

Apex Plaza, Forbury Road, Reading, Berkshire RG1 1AX, U.K.

Document Number: C048

ISBN: 1-931624-45-3

Printed in the U.K. by The Open Group.

All rights reserved. No part of this publication may be reproduced in any form, in an electronic retrieval system or otherwise, without prior written permission from both the IEEE and The Open Group.

Portions of this standard are derived with permission from copyrighted material owned by Hewlett-Packard Company, International Business Machines Corporation, Novell Inc., The Open Software Foundation, and Sun Microsystems, Inc.

Permissions

Authorization to photocopy portions of this standard for internal or personal use is granted provided that the appropriate fee is paid to the Copyright Clearance Center or the equivalent body outside of the U.S. Permission to make multiple copies for educational purposes in the U.S. requires agreement and a license fee to be paid to the Copyright Clearance Center.

Beyond these provisions, permission to reproduce all or any part of this standard must be with the consent of both copyright holders and may be subject to a license fee. Both copyright holders will need to be satisfied that the other has granted permission. Requests to the copyright holders should be sent by email to austin-group-permissions@opengroup.org.

Feedback

This standard has been prepared by the Austin Group. Feedback relating to the material contained in this standard may be submitted using the Austin Group web site at www.opengroup.org/austin/defectform.html.

IEEE

IEEE Standards documents are developed within the IEEE Societies and the Standards Coordinating Committees of the IEEE Standards Association (IEEE-SA) Standards Board. The IEEE develops its standards through a consensus development process, approved by the American National Standards Institute, which brings together volunteers representing varied viewpoints and interests to achieve the final product. Volunteers are not necessarily members of the Institute and serve without compensation. While the IEEE administers the process and establishes rules to promote fairness in the consensus development process, the IEEE does not independently evaluate, test, or verify the accuracy of any of the information contained in its standards.

Use of an IEEE Standard is wholly voluntary. The IEEE disclaims liability for any personal injury, property, or other damage, of any nature whatsoever, whether special, indirect, consequential, or compensatory, directly or indirectly resulting from the publication, use of, or reliance upon this, or any other IEEE Standard document.

The IEEE does not warrant or represent the accuracy or content of the material contained herein, and expressly disclaims any express or implied warranty, including any implied warranty of merchantability or fitness for a specific purpose, or that the use of the material contained herein is free from patent infringement. IEEE Standards documents are supplied "AS IS".

The existence of an IEEE Standard does not imply that there are no other ways to produce, test, measure, purchase, market, or provide other goods and services related to the scope of the IEEE Standard. Furthermore, the viewpoint expressed at the time a standard is approved and issued is subject to change brought about through developments in the state of the art and comments received from users of the standard. Every IEEE Standard is subjected to review at least every five years for revision or reaffirmation. When a document is more than five years old and has not been reaffirmed, it is reasonable to conclude that its contents, although still of some value, do not wholly reflect the present state of the art. Users are cautioned to check to determine that they have the latest edition of any IEEE Standard.

In publishing and making this document available, the IEEE is not suggesting or rendering professional or other services for, or on behalf of, any person or entity. Nor is the IEEE undertaking to perform any duty owed by any other person or entity to another. Any person utilizing this, and any other IEEE Standards document, should rely upon the advice of a competent professional in determining the exercise of reasonable care in any given circumstances.

Interpretations: Occasionally questions may arise regarding the meaning of portions of standards as they relate to specific applications. When the need for interpretations is brought to the attention of the IEEE, the Institute will initiate action to prepare appropriate responses. Since IEEE Standards represent a consensus of concerned interests, it is important to ensure that any interpretation has also received the concurrence of a balance of interests. For this reason, IEEE and the members of its societies and Standards Coordinating Committees are not able to provide an instant response to interpretation requests except in those cases where the matter has previously received formal consideration. At lectures, symposia, seminars, or educational courses, an individual presenting information on IEEE standards shall make it clear that his or her views should be considered the personal views of that individual rather than the formal position, explanation, or interpretation of the IEEE. Current interpretations can be accessed at <http://standards.ieee.org/reading/ieee/interp/index.html>.

Errata, if any, for this and all other standards can be accessed at <http://standards.ieee.org/reading/ieee/updates/errata/index.html>. Users are encouraged to check this URL for errata periodically.

Comments for revision of IEEE Standards are welcome from any interested party, regardless of membership affiliation with the IEEE.¹ Suggestions for changes in documents should be in the form of a proposed change of text, together with appropriate supporting comments. Comments on standards and requests for interpretations should be addressed to:

Secretary, IEEE-SA Standards Board, 445 Hoes Lane, P.O. Box 1331, Piscataway, NJ 08855-1331, U.S.A.

Attention is called to the possibility that implementation of this standard may require use of subject matter covered by patent rights. By publication of this standard, no position is taken with respect to the existence or validity of any patent rights in connection therewith. The IEEE shall not be responsible for identifying patents for which a license may be required by an IEEE Standard or for conducting inquiries into the legal validity or scope of those patents that are brought to its attention.

A patent holder has filed a statement of assurance that it will grant licenses under these rights without compensation or under reasonable rates and non-discriminatory, reasonable terms and conditions to all applicants desiring to obtain such licenses. The IEEE makes no representation as to the reasonableness of rates and/or terms and conditions of the license agreements offered by patent holders. Further information may be obtained from the IEEE Standards Department.

Authorization to photocopy portions of any individual standard for internal or personal use is granted in the U.S. by the Institute of Electrical and Electronics Engineers, Inc., provided that the appropriate fee is paid to the Copyright Clearance Center.² Permission to photocopy portions of any individual standard for educational classroom use can also be obtained through the Copyright Clearance Center. To arrange for payment of the licensing fee, please contact:

Copyright Clearance Center, Customer Service, 222 Rosewood Drive, Danvers, MA 01923, U.S.A., Tel.: +1 978 750 8400

Amendments, corrigenda, and interpretations for this standard, or information about the IEEE standards development process, may be found at <http://standards.ieee.org>.

The IEEE publications catalog and ordering information are available at <http://shop.ieee.org/store>.

1. For this standard, please send comments via the Austin Group as requested on page ii.

2. Please refer to the special provisions for this standard on page ii concerning permissions from both copyright holders and arrangements to cover photocopying and reproduction across the world, as well as by commercial organizations wishing to license the material for use in product documentation.

The Open Group

The Open Group is a vendor-neutral and technology-neutral consortium, whose vision of Boundaryless Information Flow will enable access to integrated information within and between enterprises based on open standards and global interoperability. The Open Group works with customers, suppliers, consortia, and other standards bodies. Its role is to capture, understand, and address current and emerging requirements, establish policies, and share best practices; to facilitate interoperability, develop consensus, and evolve and integrate specifications and Open Source technologies; to offer a comprehensive set of services to enhance the operational efficiency of consortia; and to operate the industry's premier certification service, including UNIX certification.

Further information on The Open Group is available at www.opengroup.org.

The Open Group has over 15 years' experience in developing and operating certification programs and has extensive experience developing and facilitating industry adoption of test suites used to validate conformance to an open standard or specification. The Open Group portfolio of test suites includes the *Westwood* family of tests for this standard and the associated certification program for Version 3 of the Single UNIX Specification, as well tests for CDE, CORBA, Motif, Linux, LDAP, POSIX.1, POSIX.2, POSIX Realtime, Sockets, UNIX, XPG4, XNFS, XTI, and X11. The Open Group test tools are essential for proper development and maintenance of standards-based products, ensuring conformance of products to industry-standard APIs, applications portability, and interoperability. In-depth testing identifies defects at the earliest possible point in the development cycle, saving costs in development and quality assurance.

More information is available at www.opengroup.org/testing.

The Open Group publishes a wide range of technical documentation, the main part of which is focused on development of Technical and Product Standards and Guides, but which also includes white papers, technical studies, branding and testing documentation, and business titles. Full details and a catalog are available at www.opengroup.org/pubs.

As with all *live* documents, Technical Standards and Specifications require revision to align with new developments and associated international standards. To distinguish between revised specifications which are fully backwards compatible and those which are not:

- A new *Version* indicates there is no change to the definitive information contained in the previous publication of that title, but additions/extensions are included. As such, it *replaces* the previous publication.
- A new *Issue* indicates there is substantive change to the definitive information contained in the previous publication of that title, and there may also be additions/extensions. As such, both previous and new documents are maintained as current publications.

Readers should note that Corrigenda may apply to any publication. Corrigenda information is published at www.opengroup.org/corrigenda.

The Open Group publications catalog and ordering information are available at www.opengroup.org/pubs.

Contents

Chapter	1	Introduction.....	1
1.1	Scope.....	1	
1.2	Conformance	1	
1.3	Normative References	1	
1.4	Change History	1	
1.5	Terminology	1	
1.6	Definitions	3	
1.7	Relationship to Other Documents.....	3	
1.7.1	System Interfaces	3	
1.7.1.1	Process Attributes.....	3	
1.7.1.2	Concurrent Execution of Processes.....	3	
1.7.1.3	File Access Permissions	4	
1.7.1.4	File Read, Write, and Creation	4	
1.7.1.5	File Removal	6	
1.7.1.6	File Time Values	6	
1.7.1.7	File Contents	6	
1.7.1.8	Pathname Resolution.....	7	
1.7.1.9	Changing the Current Working Directory.....	7	
1.7.1.10	Establish the Locale	7	
1.7.1.11	Actions Equivalent to Functions.....	7	
1.7.2	Concepts Derived from the ISO C Standard.....	7	
1.7.2.1	Arithmetic Precision and Operations	7	
1.7.2.2	Mathematical Functions	9	
1.8	Portability	9	
1.8.1	Codes.....	9	
1.9	Utility Limits.....	17	
1.10	Grammar Conventions	19	
1.11	Utility Description Defaults.....	20	
1.12	Considerations for Utilities in Support of Files of Arbitrary Size ...	27	
1.13	Built-In Utilities.....	28	
Chapter	2	Shell Command Language	29
2.1	Shell Introduction	29	
2.2	Quoting.....	30	
2.2.1	Escape Character (Backslash).....	30	
2.2.2	Single-Quotes.....	30	
2.2.3	Double-Quotes	30	
2.3	Token Recognition.....	31	
2.3.1	Alias Substitution	32	
2.4	Reserved Words	33	
2.5	Parameters and Variables.....	33	
2.5.1	Positional Parameters.....	33	

2.5.2	Special Parameters.....	34
2.5.3	Shell Variables.....	34
2.6	Word Expansions.....	36
2.6.1	Tilde Expansion.....	37
2.6.2	Parameter Expansion	37
2.6.3	Command Substitution.....	40
2.6.4	Arithmetic Expansion	41
2.6.5	Field Splitting.....	42
2.6.6	Pathname Expansion.....	42
2.6.7	Quote Removal.....	42
2.7	Redirection	43
2.7.1	Redirecting Input.....	44
2.7.2	Redirecting Output.....	44
2.7.3	Appending Redirected Output	44
2.7.4	Here-Document.....	44
2.7.5	Duplicating an Input File Descriptor.....	45
2.7.6	Duplicating an Output File Descriptor	45
2.7.7	Open File Descriptors for Reading and Writing.....	46
2.8	Exit Status and Errors	46
2.8.1	Consequences of Shell Errors	46
2.8.2	Exit Status for Commands	46
2.9	Shell Commands	47
2.9.1	Simple Commands	47
2.9.1.1	Command Search and Execution	48
2.9.2	Pipelines.....	49
2.9.3	Lists.....	50
2.9.3.1	Asynchronous Lists.....	50
2.9.3.2	Sequential Lists.....	51
2.9.3.3	AND Lists.....	51
2.9.3.4	OR Lists.....	51
2.9.4	Compound Commands.....	52
2.9.4.1	Grouping Commands.....	52
2.9.4.2	The for Loop.....	52
2.9.4.3	Case Conditional Construct	53
2.9.4.4	The if Conditional Construct.....	53
2.9.4.5	The while Loop.....	54
2.9.4.6	The until Loop	54
2.9.5	Function Definition Command.....	54
2.10	Shell Grammar.....	55
2.10.1	Shell Grammar Lexical Conventions.....	55
2.10.2	Shell Grammar Rules	56
2.11	Signals and Error Handling	61
2.12	Shell Execution Environment.....	61
2.13	Pattern Matching Notation	62
2.13.1	Patterns Matching a Single Character	62
2.13.2	Patterns Matching Multiple Characters	63
2.13.3	Patterns Used for Filename Expansion	63
2.14	Special Built-In Utilities.....	64

<i>break</i>	65
<i>colon</i>	67
<i>continue</i>	69
<i>dot</i>	71
<i>eval</i>	73
<i>exec</i>	75
<i>exit</i>	77
<i>export</i>	79
<i>readonly</i>	82
<i>return</i>	84
<i>set</i>	86
<i>shift</i>	92
<i>times</i>	94
<i>trap</i>	96
<i>unset</i>	99
Chapter 3 Batch Environment Services	101
3.1 General Concepts.....	101
Batch Client-Server Interaction.....	101
3.1.2 Batch Queues	101
3.1.3 Batch Job Creation	102
3.1.4 Batch Job Tracking	102
3.1.5 Batch Job Routing	102
3.1.6 Batch Job Execution.....	103
3.1.7 Batch Job Exit.....	103
3.1.8 Batch Job Abort	103
3.1.9 Batch Authorization	103
3.1.10 Batch Administration.....	104
3.1.11 Batch Notification	104
3.2 Batch Services	104
3.2.1 Batch Job States	105
3.2.2 Deferred Batch Services.....	106
3.2.2.1 Batch Job Execution.....	106
3.2.2.2 Batch Job Routing	113
3.2.2.3 Batch Job Exit.....	113
3.2.2.4 Batch Server Restart	114
3.2.2.5 Batch Job Abort	114
3.2.3 Requested Batch Services	115
3.2.3.1 Delete Batch Job Request.....	115
3.2.3.2 Hold Batch Job Request.....	116
3.2.3.3 Batch Job Message Request.....	116
3.2.3.4 Batch Job Status Request	117
3.2.3.5 Locate Batch Job Request	117
3.2.3.6 Modify Batch Job Request.....	117
3.2.3.7 Move Batch Job Request.....	118
3.2.3.8 Queue Batch Job Request	118
3.2.3.9 Batch Queue Status Request.....	119
3.2.3.10 Release Batch Job Request.....	119

3.2.3.11	Rerun Batch Job Request	120
3.2.3.12	Select Batch Jobs Request.....	120
3.2.3.13	Server Shutdown Request.....	120
3.2.3.14	Server Status Request.....	121
3.2.3.15	Signal Batch Job Request	121
3.2.3.16	Track Batch Job Request	121
3.3	Common Behavior for Batch Environment Utilities	122
3.3.1	Batch Job Identifier	122
3.3.2	Destination	123
3.3.3	Multiple Keyword-Value Pairs	123
Chapter 4	Utilities.....	125
	Index.....	1083

List of Figures

4-1	pax Format Archive Example.....	712
-----	---------------------------------	-----

List of Tables

1-1	Actions when Creating a File that Already Exists	5
1-2	Selected ISO C Standard Operators and Control Flow Keywords.....	8
1-3	Utility Limit Minimum Values.....	17
1-4	Symbolic Utility Limits.....	18
1-5	Regular Built-In Utilities	28
3-1	Batch Utilities.....	101
3-2	Environment Variable Summary	105
3-3	Next State Table	107
3-4	Results/Output Table	108
3-5	Batch Services Summary	115
4-1	Expressions in Decreasing Precedence in <i>awk</i>	156
4-2	Escape Sequences in <i>awk</i>	162
4-3	Operators in <i>bc</i>	198
4-4	Programming Environments: Type Sizes	215
4-5	Programming Environments: <i>c99</i> and <i>cc</i> Arguments.....	216
4-6	ASCII to EBCDIC Conversion.....	305
4-7	ASCII to IBM EBCDIC Conversion.....	306
4-8	File Utility Output Strings	446
4-9	Table Size Declarations in <i>lex</i>	538
4-10	Escape Sequences in <i>lex</i>	540
4-11	ERE Precedence in <i>lex</i>	541
4-12	Named Characters in <i>od</i>	681
4-13	ustar Header Block	717
4-14	ustar <i>mode</i> Field	718
4-15	Octet-Oriented <i>cpio</i> Archive Entry.....	720
4-16	Values for <i>cpio c_mode</i> Field	721
4-17	Variable Names and Default Headers in <i>ps</i>	756

Contents

4-18	Environment Variable Values (Utilities)	805
4-19	Control Character Names in <i>stty</i>	890
4-20	Circumflex Control Characters in <i>stty</i>	890
4-21	uuencode Base64 Values.....	974
4-22	Internal Limits in <i>yacc</i>	1076

Foreword

Structure of the Standard

This standard was originally developed by the Austin Group, a joint working group of members of the IEEE, members of The Open Group, and members of ISO/IEC Joint Technical Committee 1, as one of the four volumes of IEEE Std 1003.1-2001. The standard was approved by ISO and IEC and published in four parts, correlating to the original volumes.

A mapping of the parts to the volumes is shown below:

ISO/IEC 9945 Part	IEEE Std 1003.1 Volume	Description
9945-1	Base Definitions	Includes general terms, concepts, and interfaces common to all parts of ISO/IEC 9945, including utility conventions and C-language header definitions.
9945-2	System Interfaces	Includes definitions for system service functions and subroutines, language-specific system services for the C programming language, function issues, including portability, error handling, and error recovery.
9945-3	Shell and Utilities	Includes definitions for a standard source code-level interface to command interpretation services (a “shell”) and common utility programs for application programs.
9945-4	Rationale	Includes extended rationale that did not fit well into the rest of the document structure, containing historical information concerning the contents of ISO/IEC 9945 and why features were included or discarded by the standard developers.

All four parts comprise the entire standard, and are intended to be used together to accommodate significant internal referencing among them. POSIX-conforming systems are required to support all four parts.

Introduction

Note: This introduction is not part of IEEE Std 1003.1-2001, Standard for Information Technology — Portable Operating System Interface (POSIX).

This standard has been jointly developed by the IEEE and The Open Group. It is simultaneously an IEEE Standard, an ISO/IEC Standard, and an Open Group Technical Standard.

The Austin Group

This standard was developed, and is maintained, by a joint working group of members of the IEEE Portable Applications Standards Committee, members of The Open Group, and members of ISO/IEC Joint Technical Committee 1. This joint working group is known as the Austin Group.³ The Austin Group arose out of discussions amongst the parties which started in early 1998, leading to an initial meeting and formation of the group in September 1998. The purpose of the Austin Group has been to revise, combine, and update the following standards: ISO/IEC 9945-1, ISO/IEC 9945-2, IEEE Std 1003.1, IEEE Std 1003.2, and the Base Specifications of The Open Group Single UNIX Specification.

After two initial meetings, an agreement was signed in July 1999 between The Open Group and the Institute of Electrical and Electronics Engineers (IEEE), Inc., to formalize the project with the first draft of the revised specifications being made available at the same time. Under this agreement, The Open Group and IEEE agreed to share joint copyright of the resulting work. The Open Group has provided the chair and secretariat for the Austin Group.

The base document for the revision was The Open Group's Base volumes of its Single UNIX Specification, Version 2. These were selected since they were a superset of the existing POSIX.1 and POSIX.2 specifications and had some organizational aspects that would benefit the audience for the new revision.

The approach to specification development has been one of “write once, adopt everywhere”, with the deliverables being a set of specifications that carry the IEEE POSIX designation, The Open Group's Technical Standard designation, and an ISO/IEC designation. This set of specifications forms the core of the Single UNIX Specification, Version 3.

This unique development has combined both the industry-led efforts and the formal standardization activities into a single initiative, and included a wide spectrum of participants. The Austin Group continues as the maintenance body for this document.

Anyone wishing to participate in the Austin Group should contact the chair with their request. There are no fees for participation or membership. You may participate as an observer or as a contributor. You do not have to attend face-to-face meetings to participate; electronic participation is most welcome. For more information on the Austin Group and how to participate, see <http://www.opengroup.org/austin>.

3. The Austin Group is named after the location of the inaugural meeting held at the IBM facility in Austin, Texas in September 1998.

Background

The developers of this standard represent a cross section of hardware manufacturers, vendors of operating systems and other software development tools, software designers, consultants, academics, authors, applications programmers, and others.

Conceptually, this standard describes a set of fundamental services needed for the efficient construction of application programs. Access to these services has been provided by defining an interface, using the C programming language, a command interpreter, and common utility programs that establish standard semantics and syntax. Since this interface enables application writers to write portable applications—it was developed with that goal in mind—it has been designated POSIX,⁴ an acronym for Portable Operating System Interface.

Although originated to refer to the original IEEE Std 1003.1-1988, the name POSIX more correctly refers to a *family* of related standards: IEEE Std 1003.n and the parts of ISO/IEC 9945. In earlier editions of the IEEE standard, the term POSIX was used as a synonym for IEEE Std 1003.1-1988. A preferred term, POSIX.1, emerged. This maintained the advantages of readability of the symbol “POSIX” without being ambiguous with the POSIX family of standards.

Audience

The intended audience for this standard is all persons concerned with an industry-wide standard operating system based on the UNIX system. This includes at least four groups of people:

1. Persons buying hardware and software systems
2. Persons managing companies that are deciding on future corporate computing directions
3. Persons implementing operating systems, and especially
4. Persons developing applications where portability is an objective

Purpose

Several principles guided the development of this standard:

- Application-Oriented

The basic goal was to promote portability of application programs across UNIX system environments by developing a clear, consistent, and unambiguous standard for the interface specification of a portable operating system based on the UNIX system documentation. This standard codifies the common, existing definition of the UNIX system.

- Interface, Not Implementation

This standard defines an interface, not an implementation. No distinction is made between library functions and system calls; both are referred to as functions. No details of the implementation of any function are given (although historical practice is sometimes indicated in the RATIONALE section). Symbolic names are given for constants (such as signals and error numbers) rather than numbers.

4. The name POSIX was suggested by Richard Stallman. It is expected to be pronounced *pahz-icks*, as in *positive*, not *poh-six*, or other variations. The pronunciation has been published in an attempt to promulgate a standardized way of referring to a standard operating system interface.

- **Source, Not Object, Portability**

This standard has been written so that a program written and translated for execution on one conforming implementation may also be translated for execution on another conforming implementation. This standard does not guarantee that executable (object or binary) code will execute under a different conforming implementation than that for which it was translated, even if the underlying hardware is identical.

- **The C Language**

The system interfaces and header definitions are written in terms of the standard C language as specified in the ISO C standard.

- **No Superuser, No System Administration**

There was no intention to specify all aspects of an operating system. System administration facilities and functions are excluded from this standard, and functions usable only by the superuser have not been included. Still, an implementation of the standard interface may also implement features not in this standard. This standard is also not concerned with hardware constraints or system maintenance.

- **Minimal Interface, Minimally Defined**

In keeping with the historical design principles of the UNIX system, the mandatory core facilities of this standard have been kept as minimal as possible. Additional capabilities have been added as optional extensions.

- **Broadly Implementable**

The developers of this standard endeavored to make all specified functions implementable across a wide range of existing and potential systems, including:

1. All of the current major systems that are ultimately derived from the original UNIX system code (Version 7 or later)
2. Compatible systems that are not derived from the original UNIX system code
3. Emulations hosted on entirely different operating systems
4. Networked systems
5. Distributed systems
6. Systems running on a broad range of hardware

No direct references to this goal appear in this standard, but some results of it are mentioned in the Rationale (Informative) volume.

- **Minimal Changes to Historical Implementations**

When the original version of IEEE Std 1003.1-2001/Cor 2-2004 was published, there were no known historical implementations that did not have to change. However, there was a broad consensus on a set of functions, types, definitions, and concepts that formed an interface that was common to most historical implementations.

The adoption of the 1988 and 1990 IEEE system interface standards, the 1992 IEEE shell and utilities standard, the various Open Group (formerly X/Open) specifications, and the subsequent revisions and addenda to all of them have consolidated this consensus, and this revision reflects the significantly increased level of consensus arrived at since the original versions. The earlier standards and their modifications specified a number of areas where consensus had not been reached before, and these are now reflected in this revision. The authors of the original versions tried, as much as possible, to follow the principles below

when creating new specifications:

1. By standardizing an interface like one in an historical implementation; for example, directories
2. By specifying an interface that is readily implementable in terms of, and backwards-compatible with, historical implementations, such as the extended *tar* format defined in the *pax* utility
3. By specifying an interface that, when added to an historical implementation, will not conflict with it; for example, the *sigaction()* function

This revision tries to minimize the number of changes required to implementations which conform to the earlier versions of the approved standards to bring them into conformance with the current standard. Specifically, the scope of this work excluded doing any ‘‘new’’ work, but rather collecting into a single document what had been spread across a number of documents, and presenting it in what had been proven in practice to be a more effective way. Some changes to prior conforming implementations were unavoidable, primarily as a consequence of resolving conflicts found in prior revisions, or which became apparent when bringing the various pieces together.

However, since it references the 1999 version of the ISO C standard, and no longer supports ‘‘Common Usage C’’, there are a number of unavoidable changes. Applications portability is similarly affected.

This standard is specifically not a codification of a particular vendor’s product.

It should be noted that implementations will have different kinds of extensions. Some will reflect ‘‘historical usage’’ and will be preserved for execution of pre-existing applications. These functions should be considered ‘‘obsolescent’’ and the standard functions used for new applications. Some extensions will represent functions beyond the scope of this standard. These need to be used with careful management to be able to adapt to future extensions of this standard and/or port to implementations that provide these services in a different manner.

- Minimal Changes to Existing Application Code

A goal of this standard was to minimize additional work for the developers of applications. However, because every known historical implementation will have to change at least slightly to conform, some applications will have to change.

This Standard

This standard defines the Portable Operating System Interface (POSIX) requirements and consists of the following volumes:

- Base Definitions
- Shell and Utilities (this volume)
- System Interfaces
- Rationale (Informative)

This Volume

The Shell and Utilities volume describes the commands and utilities offered to application programs on POSIX-conformant systems. Readers are expected to be familiar with the Base Definitions volume.

This volume is structured as follows:

- Chapter 1 explains the status of this volume and its relationship to other formal standards. It also describes the defaults used by the utility descriptions in Chapter 4.
- Chapter 2 describes the command language used in POSIX-conformant systems.
- Chapter 3 describes a set of services and utilities that are implemented on systems supporting the Batch Environment Services and Utilities option.
- Chapter 4 consists of reference pages for all utilities available on POSIX-conformant systems.

Comprehensive references are available in the index.

Typographical Conventions

The following typographical conventions are used throughout this standard. In the text, this standard is referred to as IEEE Std 1003.1-2001, which is technically identical to The Open Group Base Specifications, Issue 6.

The typographical conventions listed here are for ease of reading only. Editorial inconsistencies in the use of typography are unintentional and have no normative meaning in this standard.

Reference	Example	Notes
C-Language Data Structure	aiocb	
C-Language Data Structure Member	<i>aio_lio_opcode</i>	
C-Language Data Type	long	
C-Language External Variable	<i>errno</i>	
C-Language Function	<i>system()</i>	
C-Language Function Argument	<i>arg1</i>	
C-Language Function Family	<i>exec</i>	
C-Language Header	<sys/stat.h>	
C-Language Keyword	return	
C-Language Macro with Argument	assert()	
C-Language Macro with No Argument	INET_ADDRSTRLEN	
C-Language Preprocessing Directive	#define	
Commands within a Utility	a, c	
Conversion Specification, Specifier/Modifier Character	%A, g, E	1
Environment Variable	PATH	
Error Number	[EINTR]	
Example Output	Hello, World	
Filename	/tmp	
Literal Character	'c', '\r', '\'	2
Literal String	"abcde"	2
Optional Items in Utility Syntax	[]	
Parameter	<directory pathname>	
Special Character	<newline>	3

Reference	Example	Notes
Symbolic Constant	<code>_POSIX_VDISABLE</code>	
Symbolic Limit, Configuration Value	<code>{LINE_MAX}</code>	4
Syntax	<code>#include <sys/stat.h></code>	
User Input and Example Code	<code>echo Hello, World</code>	5
Utility Name	<code>awk</code>	
Utility Operand	<code>file_name</code>	
Utility Option	<code>-c</code>	
Utility Option with Option-Argument	<code>-w width</code>	

Notes:

1. Conversion specifications, specifier characters, and modifier characters are used primarily in date-related functions and utilities and the *fprintf* and *fscanf* formatting functions.
2. Unless otherwise noted, the quotes shall not be used as input or output. When used in a list item, the quotes are omitted. For literal characters, '\'' (or any of the other sequences such as ''') is the same as the C constant '\\\' (or '\\''').
3. The style selected for some of the special characters, such as <newline>, matches the form of the input given to the *localedef* utility. Generally, the characters selected for this special treatment are those that are not visually distinct, such as the control characters <tab> or <newline>.
4. Names surrounded by braces represent symbolic limits or configuration values which may be declared in appropriate headers by means of the C **#define** construct.
5. Brackets shown in this font, "[]", are part of the syntax and do *not* indicate optional items. In syntax the ' | ' symbol is used to separate alternatives, and ellipses (" . . . ") are used to show that additional arguments are optional.

Shading is used to identify extensions and options; see Section 1.8.1 (on page 9).

Footnotes and notes within the body of the normative text are for information only (informative).

Informative sections (such as Rationale, Change History, Application Usage, and so on) are denoted by continuous shading bars in the margins.

Ranges of values are indicated with parentheses or brackets as follows:

- (a, b) means the range of all values from a to b , including neither a nor b .
- $[a, b]$ means the range of all values from a to b , including a and b .
- $[a, b)$ means the range of all values from a to b , including a , but not b .
- $(a, b]$ means the range of all values from a to b , including b , but not a .

Participants

IEEE Std 1003.1-2001 was prepared by the Austin Group, sponsored by the Portable Applications Standards Committee of the IEEE Computer Society, The Open Group, and ISO/SC22 WG15.

The Austin Group

At the time of approval, the membership of the Austin Group was as follows:

Andrew Josey, Chair

Donald W. Cragun, Organizational Representative, IEEE PASC

Nicholas Stoughton, Organizational Representative, ISO/SC22 WG15

Mark Brown, Organizational Representative, The Open Group

Cathy Hughes, Technical Editor

Austin Group Technical Reviewers

Peter Anvin	Michael Gonzalez	Sandra O'Donnell
Bouazza Bachar	Joseph M. Gwinn	Frank Prindle
Theodore P. Baker	Jon Hitchcock	Curtis Royster Jr.
Walter Briscoe	Yvette Ho Sang	Glen Seeds
Mark Brown	Cathy Hughes	Keld Jorn Simonsen
Dave Butenhof	Lowell G. Johnson	Raja Srinivasan
Geoff Clare	Andrew Josey	Nicholas Stoughton
Donald W. Cragun	Michael Kavanaugh	Donn S. Terry
Lee Damico	David Korn	Fred Tydeman
Ulrich Drepper	Marc Aurele La France	Peter Van Der Veen
Paul Eggert	Jim Meyering	James Youngman
Joanna Farley	Gary Miller	Jim Zepeda
Clive D.W. Feather	Finnbarr P. Murphy	Jason Zions
Andrew Gollan	Joseph S. Myers	

Austin Group Working Group Members

Harold C. Adams	Michael Gonzalez	Sandra O'Donnell
Peter Anvin	Karen D. Gordon	Frank Prindle
Pierre-Jean Arcos	Joseph M. Gwinn	Francois Riche
Jay Ashford	Steven A. Haaser	John D. Riley
Bouazza Bachar	Charles E. Hammons	Andrew K. Roach
Theodore P. Baker	Chris J. Harding	Helmut Roth
Robert Barned	Barry Hedquist	Jaideep Roy
Joel Berman	Vincent E. Henley	Curtis Royster Jr.
David J. Blackwood	Karl Heubaum	Stephen C. Schwarm
Shirley Bockstahler-Brandt	Jon Hitchcock	Glen Seeds
James Bottomley	Yvette Ho Sang	Richard Seibel
Walter Briscoe	Niklas Holsti	David L. Shroads Jr.
Andries Brouwer	Thomas Hosmer	W. Olin Sibert
Mark Brown	Cathy Hughes	Keld Jorn Simonsen
Eric W. Burger	Jim D. Isaak	Curtis Smith
Alan Burns	Lowell G. Johnson	Raja Srinivasan
Andries Brouwer	Michael B. Jones	Nicholas Stoughton
Dave Butenhof	Andrew Josey	Marc J. Teller
Keith Chow	Michael J. Karels	Donn S. Terry
Geoff Clare	Michael Kavanaugh	Fred Tydeman
Donald W. Cragun	David Korn	Mark-Rene Uchida
Lee Damico	Steven Kramer	Scott A. Valcourt
Juan Antonio De La Puente	Thomas M. Kurihara	Peter Van Der Veen
Ming De Zhou	Marc Aurele La France	Michael W. Vannier
Steven J. Dovich	C. Douglass Locke	Eric Vought
Richard P. Draves	Nick MacLaren	Frederick N. Webb
Ulrich Drepper	Roger J. Martin	Paul A.T. Wolfgang
Paul Eggert	Craig H. Meyer	Garrett A. Wollman
Philip H. Enslow	Jim Meyering	James Youngman
Joanna Farley	Gary Miller	Oren Yuen
Clive D.W. Feather	Finnbarr P. Murphy	Janusz Zalewski
Pete Forman	Joseph S. Myers	Jim Zepeda
Mark Funkenhauser	John Napier	Jason Zions
Lois Goldthwaite	Peter E. Obermayer	
Andrew Gollan	James T. Oblinger	

Participants

The Open Group

When The Open Group approved the Base Specifications, Issue 6 on 12 September 2001, the membership of The Open Group Base Working Group was as follows:

Andrew Josey, Chair

Finnbarr P. Murphy, Vice-Chair

Mark Brown, Austin Group Liaison

Cathy Hughes, Technical Editor

Base Working Group Members

Bouazza Bachar

Mark Brown

Dave Butenhof

Donald W. Cragun

Larry Dwyer

Joanna Farley

Andrew Gollan

Karen D. Gordon

Gary Miller

Finnbarr P. Murphy

Frank Prindle

Andrew K. Roach

Curtis Royster Jr.

Nicholas Stoughton

Kenjiro Tsuji

IEEE

When the IEEE Standards Board approved IEEE Std 1003.1-2001 on 6 December 2001, the membership of the committees was as follows:

Portable Applications Standards Committee (PASC)

Lowell G. Johnson, Chair
Joseph M. Gwinn, Vice-Chair
Jay Ashford, Functional Chair
Andrew Josey, Functional Chair
Curtis Royster Jr., Functional Chair
Nicholas Stoughton, Secretary

Balloting Committee

The following members of the balloting committee voted on IEEE Std 1003.1-2001. Balloters may have voted for approval, disapproval, or abstention:

Harold C. Adams	Steven A. Haaser	Frank Prindle
Pierre-Jean Arcos	Charles E. Hammons	Francois Riche
Jay Ashford	Chris J. Harding	John D. Riley
Theodore P. Baker	Barry Hedquist	Andrew K. Roach
Robert Barned	Vincent E. Henley	Helmut Roth
David J. Blackwood	Karl Heubaum	Jaideep Roy
Shirley Bockstahler-Brandt	Niklas Holsti	Curtis Royster Jr.
James Bottomley	Thomas Hosmer	Stephen C. Schwarm
Mark Brown	Jim D. Isaak	Richard Seibel
Eric W. Burger	Lowell G. Johnson	David L. Shroads Jr.
Alan Burns	Michael B. Jones	W. Olin Sibert
Dave Butenhof	Andrew Josey	Keld Jorn Simonsen
Keith Chow	Michael J. Karels	Nicholas Stoughton
Donald W. Cragun	Steven Kramer	Donn S. Terry
Juan Antonio De La Puente	Thomas M. Kurihara	Mark-Rene Uchida
Ming De Zhou	C. Douglass Locke	Scott A. Valcourt
Steven J. Dovich	Roger J. Martin	Michael W. Vannier
Richard P. Draves	Craig H. Meyer	Frederick N. Webb
Philip H. Enslow	Finnbarr P. Murphy	Paul A.T. Wolfgang
Michael Gonzalez	John Napier	Oren Yuen
Karen D. Gordon	Peter E. Obermayer	Janusz Zalewski
Joseph M. Gwinn	James T. Oblinger	

The following organizational representative voted on this standard:

Andrew Josey, X/Open Company Ltd.

Participants

IEEE-SA Standards Board

When the IEEE-SA Standards Board approved IEEE Std 1003.1-2001 on 6 December 2001, it had the following membership:

Donald N. Heirman, Chair
James T. Carlo, Vice-Chair
Judith Gorman, Secretary

Satish K. Aggarwal
Mark D. Bowman
Gary R. Engmann
Harold E. Epstein
H. Landis Floyd
Jay Forster*
Howard M. Frazier
Ruben D. Garzon

James H. Gurney
Richard J. Holleman
Lowell G. Johnson
Robert J. Kennelly
Joseph L. Koepfinger*
Peter H. Lips
L. Bruce McClung
Daleep C. Mohla

James W. Moore
Robert F. Munzner
Ronald C. Petersen
Gerald H. Peterson
John B. Posey
Gary S. Robinson
Akio Tojo
Donald W. Zipse

Also included are the following non-voting IEEE-SA Standards Board liaisons:

Alan Cookson, NIST Representative
Donald R. Volzka, TAB Representative
Yvette Ho Sang, Don Messina, Savoula Amanatidis, IEEE Project Editors

* Member Emeritus

IEEE Std 1003.1-2001/Cor 1-2002 was prepared by the Austin Group, sponsored by the Portable Applications Standards Committee of the IEEE Computer Society, The Open Group, and ISO/IEC JTC 1/SC22/WG15.

The Austin Group

At the time of approval, the membership of the Austin Group was as follows:

Andrew Josey, Chair

Donald W. Cragun, Organizational Representative, IEEE PASC

Nicholas Stoughton, Organizational Representative, ISO/IEC JTC 1/SC22/WG15

Mark Brown, Organizational Representative, The Open Group

Cathy Fox, Technical Editor

Austin Group Technical Reviewers

Theodore P. Baker

Julian Blake

Andries Brouwer

Mark Brown

Dave Butenhof

Geoff Clare

Donald W. Cragun

Ken Dawson

Ulrich Drepper

Larry Dwyer

Paul Eggert

Joanna Farley

Clive D.W. Feather

Cathy Fox

Mark Funkenhauser

Lois Goldthwaite

Andrew Gollan

Michael Gonzalez

Bruno Haible

Ben Harris

Jon Hitchcock

Andreas Jaeger

Andrew Josey

Jonathan Lennox

Nick Maclare

Jack McCann

Wilhelm Mueller

Joseph S. Myers

Frank Prindle

Kenneth Raeburn

Tim Robbins

Glen Seeds

Matthew Seitz

Keld Jorn Simonsen

Nicholas Stoughton

Alexander Terekhov

Donn S. Terry

Mike Wilson

Garrett A. Wollman

Mark Ziegast

Participants

Austin Group Working Group Members

Harold C. Adams	Clive D.W. Feather	Wilhelm Mueller
Alejandro Alonso	Yaacov Fenster	Finnbarr P. Murphy
Jay Ashford	Cathy Fox	Joseph S. Myers
Theodore P. Baker	Mark Funkenhauser	Alexey Neyman
David J. Blackwood	Lois Goldthwaite	Charles Ngethe
Julian Blake	Andrew Gollan	Peter Petrov
Mitchell Bonnett	Michael Gonzalez	Frank Prindle
Andries Brouwer	Karen D. Gordon	Vikram Punj
Mark Brown	Scott Gudgel	Kenneth Raeburn
Eric W. Burger	Joseph M. Gwinn	Francois Riche
Alan Burns	Steven A. Haaser	Tim Robbins
Dave Butenhof	Bruno Haible	Curtis Royster Jr.
Keith Chow	Charles E. Hammons	Diane Schleicher
Geoff Clare	Bryan Harold	Gil Shultz
Luis Cordova	Ben Harris	Stephen C. Schwarm
Donald W. Cragun	Barry Hedquist	Glen Seeds
Dragan Cvetkovic	Karl Heubaum	Matthew Seitz
Lee Damico	Jon Hitchcock	Keld Jorn Simonsen
Ken Dawson	Andreas Jaeger	Doug Stevenson
Jeroen Dekkers	Andrew Josey	Nicholas Stoughton
Juan Antonio De La Puente	Kenneth Lang	Alexander Terekhov
Steven J. Dovich	Pi-Cheng Law	Donn S. Terry
Ulrich Drepper	Jonathan Lennox	Mike Wilson
Dr. Sourav Dutta	Nick Maclare	Garrett A. Wollman
Larry Dwyer	Roger J. Martin	Oren Yuen
Paul Eggert	Jack McCann	Mark Ziegast
Joanna Farley	George Miao	

The Open Group

When The Open Group approved the Base Specifications, Issue 6, Technical Corrigendum 1 on 7 February 2003, the membership of The Open Group Base Working Group was as follows:

Andrew Josey, Chair

Finnbarr P. Murphy, Vice-Chair

Mark Brown, Austin Group Liaison

Cathy Fox, Technical Editor

Base Working Group Members

Mark Brown

Dave Butenhof

Donald W. Cragun

Larry Dwyer

Ulrich Drepper

Joanna Farley

Andrew Gollan

Finnbarr P. Murphy

Frank Prindle

Andrew K. Roach

Curtis Royster Jr.

Nicholas Stoughton

Kenjiro Tsuji

Participants

IEEE

When the IEEE Standards Board approved IEEE Std 1003.1-2001/Cor 1-2002 on 11 December 2002, the membership of the committees was as follows:

Portable Applications Standards Committee (PASC)

Lowell G. Johnson, Chair
Joseph M. Gwinn, Vice-Chair
Jay Ashford, Functional Chair
Andrew Josey, Functional Chair
Curtis Royster Jr., Functional Chair
Nicholas Stoughton, Secretary

Balloting Committee

The following members of the balloting committee voted on IEEE Std 1003.1-2001/Cor 1-2002. Balloters may have voted for approval, disapproval, or abstention:

Alejandro Alonso	Michael Gonzalez	Charles Ngethe
Jay Ashford	Scott Gudgel	Peter Petrov
David J. Blackwood	Charles E. Hammons	Frank Prindle
Julian Blake	Bryan Harold	Vikram Punj
Mitchell Bonnett	Barry Hedquist	Francois Riche
Mark Brown	Karl Heubaum	Curtis Royster Jr.
Dave Butenhof	Lowell G. Johnson	Diane Schleicher
Keith Chow	Andrew Josey	Stephen C. Schwarm
Luis Cordova	Kenneth Lang	Gil Shultz
Donald W. Cragun	Pi-Cheng Law	Nicholas Stoughton
Steven J. Dovich	George Miao	Donn S. Terry
Dr. Sourav Dutta	Roger J. Martin	Oren Yuen
Yaakov Fenster	Finnbarr P. Murphy	Juan A. de la Puente

IEEE-SA Standards Board

When the IEEE-SA Standards Board approved IEEE Std 1003.1-2001/Cor 1-2002 on 11 December 2002, the membership was as follows:

James T. Carlo, Chair
James H. Gurney, Vice-Chair
Judith Gorman, Secretary

Sid Bennett	Arnold M. Greenspan	Daleep C. Mohla
H. Stephen Berger	Raymond Hapeman	William J. Moylan
Clyde R. Camp	Donald M. Heirman	Malcolm V. Thaden
Richard DeBlasio	Richard H. Hulett	Geoffrey O. Thompson
Harold E. Epstein	Lowell G. Johnson	Howard L. Wolfman
Julian Forster*	Joseph L. Koepfinger*	Don Wright
Howard M. Frazier	Peter H. Lips	
Toshio Fukuda	Nader Mehravari	

Also included are the following non-voting IEEE-SA Standards Board liaisons:

Alan Cookson, NIST Representative
Satish K. Aggarwal, NRC Representative
Savoula Amanatidis, IEEE Standards Managing Editor

* Member Emeritus

Participants

IEEE Std 1003.1-2001/Cor 2-2004 was prepared by the Austin Group, sponsored by the Portable Applications Standards Committee of the IEEE Computer Society, The Open Group, and ISO/IEC JTC 1/SC22/WG15.

The Austin Group

At the time of approval, the membership of the Austin Group was as follows:

Andrew Josey, Chair

Donald W. Cragun, Organizational Representative, IEEE PASC

Nicholas Stoughton, Organizational Representative, ISO/IEC JTC 1/SC22/WG15

Mark Brown, Organizational Representative, The Open Group

Cathy Fox, Technical Editor

Austin Group Technical Reviewers

Jay Ashford	Clive D.W. Feather	Rajesh Moorkath
Julian Blake	Yaacov Fenster	Peter Petrov
Mitchell Bonnett	Mark Funkenhauser	Franklin Prindle
Andries Brouwer	Ernesto Garcia	Vikram Punj
Mark Brown	Andrew Gollan	Eusebio Rufian-Zilberman
Paul Buerger	Michael Gonzalez	Joerg Schilling
Dave Butenhof	Jean-Denis Gorin	Stephen Schwarm
Keith Chow	Matthew Gream	Gil Shultz
Geoff Clare	Scott Gudgel	Keld Simonsen
Donald W. Cragun	Bruno Haible	Nicholas Stoughton
Lee Damico	Charles Hammons	Alexander Terekhov
Maulik Dave	Barry Hedquist	Donn Terry
Juan A. de la Puente	Jon Hitchcock	Mark-Rene Uchida
Guru Dutt Dhingra	Lowell G. Johnson	Thomas Unsicker
Loic Domaigne	Andrew Josey	Scott Valcourt
Ulrich Drepper	Piotr Karocki	Mats Wichmann
Sourav Dutta	David Leciston	Garrett A. Wollman
Larry Dwyer	Ryan Madron	Oren Yuen
Paul Eggert	Roger J. Martin	Mark Ziegast
Joanna Farley	George Miao	

Austin Group Working Group Members

Harold C. Adams	Mark Funkenhauser	George Miao
Jay Ashford	Ernesto Garcia	Rajesh Moorkath
Theodore P. Baker	Lois Goldthwaite	Vilhelm Mueller
David J. Blackwood	Andrew Gollan	Joseph S. Myers
Julian Blake	Michael Gonzalez	Peter Petrov
Mitchell Bonnett	Karen D. Gordon	Franklin Prindle
Andries Brouwer	Jean-Denis Gorin	Vikram Punj
Mark Brown	Matthew Gream	Kenneth Raeburn
Paul Buerger	Scott Gudgel	Tim Robbins
Alan Burns	Joseph M. Gwinn	Curtis Royster Jr.
Dave Butenhof	Steven A. Haaser	Eusebio Rufian-Zilberman
Keith Chow	Charles Hammons	Joerg Schilling
Geoff Clare	Ben Harris	Stephen Schwarm
Donald W. Cragun	Barry Hedquist	Glen Seeds
Dragan Cvetkovic	Karl Heubaum	Gil Shultz
Lee Damico	Jon Hitchcock	Keld Simonsen
Maulik Dave	Andreas Jaeger	Nicholas Stoughton
Juan A. de la Puente	Lowell G. Johnson	Alexander Terekhov
Loic Domaigne	Andrew Josey	Donn Terry
Steven J. Dovich	Piotr Karocki	Mark-Rene Uchida
Ulrich Drepper	Kenneth Lang	Thomas Unsicker
Guru Dutt Dhingra	Pi-Cheng Law	Scott Valcourt
Sourav Dutta	David Leciston	Mats Wichmann
Larry Dwyer	Wojtek Lerch	Mike Wilson
Paul Eggert	Jonathan Lennox	Garrett A. Wollman
Joanna Farley	Nick Maclare	Oren Yuen
Clive D.W. Feather	Ryan Madron	Mark Ziegast
Yaakov Fenster	Roger J. Martin	Jason Zions

The Open Group

When The Open Group approved the Base Specifications, Issue 6, Technical Corrigendum 2 on 18 December 2003, the membership of The Open Group Base Working Group was as follows:

Andrew Josey, Chair

Mark Brown, Austin Group Liaison

Cathy Fox, Technical Editor

Base Working Group Members

Mark Brown	IBM Corporation
Dave Butenhof	Hewlett-Packard Company
Donald W. Cragun	Sun Microsystems, Inc.
Larry Dwyer	Hewlett-Packard Company
Ulrich Drepper	Red Hat, Inc.
Joanna Farley	Sun Microsystems, Inc.
Andrew Gollan	Sun Microsystems, Inc.
Andrew K. Roach	Sun Microsystems, Inc.
Curtis Royster Jr.	US DoD DISA
Nicholas Stoughton	USENIX Association
Kenjiro Tsuji	Sun Microsystems, Inc.

IEEE

When the IEEE Standards Board approved IEEE Std 1003.1-2001/Cor 2-2004 on 9 February 2004, the membership of the committees was as follows:

Portable Applications Standards Committee (PASC)

Lowell G. Johnson, Chair
Joseph M. Gwinn, Vice-Chair
Jay Ashford, Functional Chair
Andrew Josey, Functional Chair
Curtis Royster Jr., Functional Chair
Nicholas Stoughton, Secretary

Balloting Committee

The following members of the balloting committee voted on IEEE Std 1003.1-2001/Cor 2-2004. Balloters may have voted for approval, disapproval, or abstention:

Jay Ashford	Matthew Gream	Rajesh Moorkath
Julian Blake	Scott Gudgel	Peter Petrov
Mark Brown	Charles Hammons	Vikram Punj
Keith Chow	Barry Hedquist	Eusebio Rufian-Zilberman
Donald W. Cragun	Andrew Josey	Stephen Schwarm
Juan A. de la Puente	Piotr Karocki	Gil Shultz
Guru Dutt Dhingra	David Leciston	Mark-Rene Uchida
Ernesto Garcia	Ryan Madron	Thomas Unsicker
Michael Gonzalez	Roger J. Martin	Scott Valcourt
Jean-Denis Gorin	George Miao	Oren Yuen

Participants

IEEE-SA Standards Board

When the IEEE-SA Standards Board approved IEEE Std 1003.1-2001/Cor 2-2004 on 9 February 2004, the membership was as follows:

Don Wright, Chair
Judith Gorman, Secretary

Chuck Adams
H. Stephen Berger
Mark D. Bowman
Joseph A. Bruder
Bob Davis
Roberto de Boisson
Julian Forster*
Arnold M. Greenspan
Mark S. Halpin

Raymond Hapeman
Richard J. Holleman
Richard H. Hulett
Lowell G. Johnson
Hermann Koch
Joseph L. Koepfinger*
Thomas J. McGean
Steve M. Mills
Daleep C. Mohla

Paul Nikolich
T. W. Olsen
Ronald C. Petersen
Gary S. Robinson
Frank Stone
Malcolm V. Thaden
Doug Topping
Joe D. Watson

Also included are the following non-voting IEEE-SA Standards Board liaisons:

Satish K. Aggarwal, NRC Representative
Richard DeBlasio, DOE Representative
Alan Cookson, NIST Representative

Savoula Amanatidis, IEEE Standards Managing Editor

* Member Emeritus

Trademarks

The following information is given for the convenience of users of this standard and does not constitute endorsement of these products by The Open Group or the IEEE. There may be other products mentioned in the text that might be covered by trademark protection and readers are advised to verify them independently.

1003.1TM is a trademark of the Institute of Electrical and Electronic Engineers, Inc.

AIX[®] is a registered trademark of IBM Corporation.

AT&T[®] is a registered trademark of AT&T in the U.S.A. and other countries.

BSDTM is a trademark of the University of California, Berkeley, U.S.A.

Hewlett-Packard[®], HP[®], and HP-UX[®] are registered trademarks of Hewlett-Packard Company.

IBM[®] is a registered trademark of International Business Machines Corporation.

Boundaryless Information Flow is a trademark and UNIX and The Open Group are registered trademarks of The Open Group in the United States and other countries.

All other trademarks are the property of their respective owners.

POSIX[®] is a registered trademark of the Institute of Electrical and Electronic Engineers, Inc.

Sun[®] and Sun Microsystems[®] are registered trademarks of Sun Microsystems, Inc.

/usr/group[®] is a registered trademark of UniForum, the International Network of UNIX System Users.

Acknowledgements

The contributions of the following organizations to the development of IEEE Std 1003.1-2001 are gratefully acknowledged:

- AT&T for permission to reproduce portions of its copyrighted System V Interface Definition (SVID) and material from the UNIX System V Release 2.0 documentation.
- The SC22 WG14 Committees.

This standard was prepared by the Austin Group, a joint working group of the IEEE, The Open Group, and ISO SC22 WG15.

Referenced Documents

Normative References

Normative references for this standard are defined in the Base Definitions volume.

Informative References

The following documents are referenced in this standard:

1984 /usr/group Standard

/usr/group Standards Committee, Santa Clara, CA, UniForum 1984.

Almasi and Gottlieb

George S. Almasi and Allan Gottlieb, *Highly Parallel Computing*, The Benjamin/Cummings Publishing Company, Inc., 1989, ISBN: 0-8053-0177-1.

ANSI C

American National Standard for Information Systems: Standard X3.159-1989, Programming Language C.

ANSI X3.226-1994

American National Standard for Information Systems: Standard X3.226-1994, Programming Language Common LISP.

Brawer

Steven Brawer, *Introduction to Parallel Programming*, Academic Press, 1989, ISBN: 0-12-128470-0.

DeRemer and Pennello Article

DeRemer, Frank and Pennello, Thomas J., *Efficient Computation of LALR(1) Look-Ahead Sets*, SigPlan Notices, Volume 15, No. 8, August 1979.

Draft ANSI X3J11.1

IEEE Floating Point draft report of ANSI X3J11.1 (NCEG).

FIPS 151-1

Federal Information Procurement Standard (FIPS) 151-1. Portable Operating System Interface (POSIX)—Part 1: System Application Program Interface (API) [C Language].

FIPS 151-2

Federal Information Procurement Standards (FIPS) 151-2, Portable Operating System Interface (POSIX)—Part 1: System Application Program Interface (API) [C Language].

HP-UX Manual

Hewlett-Packard HP-UX Release 9.0 Reference Manual, Third Edition, August 1992.

IEC 60559: 1989

IEC 60559: 1989, Binary Floating-Point Arithmetic for Microprocessor Systems (previously designated IEC 559: 1989).

IEEE Std 754-1985

IEEE Std 754-1985, IEEE Standard for Binary Floating-Point Arithmetic.

IEEE Std 854-1987

IEEE Std 854-1987, IEEE Standard for Radix-Independent Floating-Point Arithmetic.

Referenced Documents

- IEEE Std 1003.9-1992**
IEEE Std 1003.9-1992, IEEE Standard for Information Technology — POSIX FORTRAN 77 Language Interfaces — Part 1: Binding for System Application Program Interface API.
- IETF RFC 791**
Internet Protocol, Version 4 (IPv4), September 1981.
- IETF RFC 819**
The Domain Naming Convention for Internet User Applications, Z. Su, J. Postel, August 1982.
- IETF RFC 822**
Standard for the Format of ARPA Internet Text Messages, D.H. Crocker, August 1982.
- IETF RFC 919**
Broadcasting Internet Datagrams, J. Mogul, October 1984.
- IETF RFC 920**
Domain Requirements, J. Postel, J. Reynolds, October 1984.
- IETF RFC 921**
Domain Name System Implementation Schedule, J. Postel, October 1984.
- IETF RFC 922**
Broadcasting Internet Datagrams in the Presence of Subnets, J. Mogul, October 1984.
- IETF RFC 1034**
Domain Names — Concepts and Facilities, P. Mockapetris, November 1987.
- IETF RFC 1035**
Domain Names — Implementation and Specification, P. Mockapetris, November 1987.
- IETF RFC 1123**
Requirements for Internet Hosts — Application and Support, R. Braden, October 1989.
- IETF RFC 1886**
DNS Extensions to Support Internet Protocol, Version 6 (IPv6), C. Huitema, S. Thomson, December 1995.
- IETF RFC 2045**
Multipurpose Internet Mail Extensions (MIME), Part 1: Format of Internet Message Bodies, N. Freed, N. Borenstein, November 1996.
- IETF RFC 2181**
Clarifications to the DNS Specification, R. Elz, R. Bush, July 1997.
- IETF RFC 2373**
Internet Protocol, Version 6 (IPv6) Addressing Architecture, S. Deering, R. Hinden, July 1998.
- IETF RFC 2460**
Internet Protocol, Version 6 (IPv6), S. Deering, R. Hinden, December 1998.
- Internationalisation Guide**
Guide, July 1993, Internationalisation Guide, Version 2 (ISBN: 1-859120-02-4, G304), published by The Open Group.
- ISO C (1990)**
ISO/IEC 9899:1990, Programming Languages — C, including Amendment 1:1995 (E), C Integrity (Multibyte Support Extensions (MSE) for ISO C).

ISO 2375:1985

ISO 2375:1985, Data Processing — Procedure for Registration of Escape Sequences.

ISO 8652:1987

ISO 8652:1987, Programming Languages — Ada (technically identical to ANSI standard 1815A-1983).

ISO/IEC 1539:1990

ISO/IEC 1539:1990, Information Technology — Programming Languages — Fortran (technically identical to the ANSI X3.9-1978 standard [FORTRAN 77]).

ISO/IEC 4873:1991

ISO/IEC 4873:1991, Information Technology — ISO 8-bit Code for Information Interchange — Structure and Rules for Implementation.

ISO/IEC 6429:1992

ISO/IEC 6429:1992, Information Technology — Control Functions for Coded Character Sets.

ISO/IEC 6937:1994

ISO/IEC 6937:1994, Information Technology — Coded Character Set for Text Communication — Latin Alphabet.

ISO/IEC 8802-3:1996

ISO/IEC 8802-3:1996, Information Technology — Telecommunications and Information Exchange Between Systems — Local and Metropolitan Area Networks — Specific Requirements — Part 3: Carrier Sense Multiple Access with Collision Detection (CSMA/CD) Access Method and Physical Layer Specifications.

ISO/IEC 8859

ISO/IEC 8859, Information Technology — 8-Bit Single-Byte Coded Graphic Character Sets:

Part 1: Latin Alphabet No. 1

Part 2: Latin Alphabet No. 2

Part 3: Latin Alphabet No. 3

Part 4: Latin Alphabet No. 4

Part 5: Latin/Cyrillic Alphabet

Part 6: Latin/Arabic Alphabet

Part 7: Latin/Greek Alphabet

Part 8: Latin/Hebrew Alphabet

Part 9: Latin Alphabet No. 5

Part 10: Latin Alphabet No. 6

Part 11: Latin/Thai Alphabet

Part 13: Latin Alphabet No. 7

Part 14: Latin Alphabet No. 8

Part 15: Latin Alphabet No. 9

Part 16: Latin Alphabet No. 10

ISO POSIX-1:1996

ISO/IEC 9945-1:1996, Information Technology — Portable Operating System Interface (POSIX) — Part 1: System Application Program Interface (API) [C Language] (identical to ANSI/IEEE Std 1003.1-1996). Incorporating ANSI/IEEE Stds 1003.1-1990, 1003.1b-1993, 1003.1c-1995, and 1003.1i-1995.

ISO POSIX-2:1993

ISO/IEC 9945-2:1993, Information Technology — Portable Operating System Interface (POSIX) — Part 2: Shell and Utilities (identical to ANSI/IEEE Std 1003.2-1992, as amended

Referenced Documents

by ANSI/IEEE Std 1003.2a-1992).

Issue 1

X/Open Portability Guide, July 1985 (ISBN: 0-444-87839-4).

Issue 2

X/Open Portability Guide, January 1987:

- Volume 1: XVS Commands and Utilities (ISBN: 0-444-70174-5)
- Volume 2: XVS System Calls and Libraries (ISBN: 0-444-70175-3)

Issue 3

X/Open Specification, 1988, 1989, February 1992:

- Commands and Utilities, Issue 3 (ISBN: 1-872630-36-7, C211); this specification was formerly X/Open Portability Guide, Issue 3, Volume 1, January 1989, XSI Commands and Utilities (ISBN: 0-13-685835-X, XO/XPG/89/002)
- System Interfaces and Headers, Issue 3 (ISBN: 1-872630-37-5, C212); this specification was formerly X/Open Portability Guide, Issue 3, Volume 2, January 1989, XSI System Interface and Headers (ISBN: 0-13-685843-0, XO/XPG/89/003)
- Curses Interface, Issue 3, contained in Supplementary Definitions, Issue 3 (ISBN: 1-872630-38-3, C213), Chapters 9 to 14 inclusive; this specification was formerly X/Open Portability Guide, Issue 3, Volume 3, January 1989, XSI Supplementary Definitions (ISBN: 0-13-685850-3, XO/XPG/89/004)
- Headers Interface, Issue 3, contained in Supplementary Definitions, Issue 3 (ISBN: 1-872630-38-3, C213), Chapter 19, Cpio and Tar Headers; this specification was formerly X/Open Portability Guide Issue 3, Volume 3, January 1989, XSI Supplementary Definitions (ISBN: 0-13-685850-3, XO/XPG/89/004)

Issue 4

CAE Specification, July 1992, published by The Open Group:

- System Interface Definitions (XBD), Issue 4 (ISBN: 1-872630-46-4, C204)
- Commands and Utilities (XCU), Issue 4 (ISBN: 1-872630-48-0, C203)
- System Interfaces and Headers (XSH), Issue 4 (ISBN: 1-872630-47-2, C202)

Issue 4, Version 2

CAE Specification, August 1994, published by The Open Group:

- System Interface Definitions (XBD), Issue 4, Version 2 (ISBN: 1-85912-036-9, C434)
- Commands and Utilities (XCU), Issue 4, Version 2 (ISBN: 1-85912-034-2, C436)
- System Interfaces and Headers (XSH), Issue 4, Version 2 (ISBN: 1-85912-037-7, C435)

Issue 5

Technical Standard, February 1997, published by The Open Group:

- System Interface Definitions (XBD), Issue 5 (ISBN: 1-85912-186-1, C605)
- Commands and Utilities (XCU), Issue 5 (ISBN: 1-85912-191-8, C604)
- System Interfaces and Headers (XSH), Issue 5 (ISBN: 1-85912-181-0, C606)

Knuth Article

Knuth, Donald E., *On the Translation of Languages from Left to Right*, Information and Control, Volume 8, No. 6, October 1965.

KornShell

Bolsky, Morris I. and Korn, David G., *The New KornShell Command and Programming Language*, March 1995, Prentice Hall.

MSE Working Draft

Working draft of ISO/IEC 9899:1990/Add3: Draft, Addendum 3 — Multibyte Support Extensions (MSE) as documented in the ISO Working Paper SC22/WG14/N205 dated 31 March 1992.

POSIX.0: 1995

IEEE Std 1003.0-1995, IEEE Guide to the POSIX Open System Environment (OSE) (identical to ISO/IEC TR 14252).

POSIX.1: 1988

IEEE Std 1003.1-1988, IEEE Standard for Information Technology — Portable Operating System Interface (POSIX) — Part 1: System Application Program Interface (API) [C Language].

POSIX.1: 1990

IEEE Std 1003.1-1990, IEEE Standard for Information Technology — Portable Operating System Interface (POSIX) — Part 1: System Application Program Interface (API) [C Language].

POSIX.1a

P1003.1a, Standard for Information Technology — Portable Operating System Interface (POSIX) — Part 1: System Application Program Interface (API) — (C Language) Amendment.

POSIX.1d: 1999

IEEE Std 1003.1d-1999, IEEE Standard for Information Technology — Portable Operating System Interface (POSIX) — Part 1: System Application Program Interface (API) — Amendment 4: Additional Realtime Extensions [C Language].

POSIX.1g: 2000

IEEE Std 1003.1g-2000, IEEE Standard for Information Technology — Portable Operating System Interface (POSIX) — Part 1: System Application Program Interface (API) — Amendment 6: Protocol-Independent Interfaces (PII).

POSIX.1j: 2000

IEEE Std 1003.1j-2000, IEEE Standard for Information Technology — Portable Operating System Interface (POSIX) — Part 1: System Application Program Interface (API) — Amendment 5: Advanced Realtime Extensions [C Language].

POSIX.1q: 2000

IEEE Std 1003.1q-2000, IEEE Standard for Information Technology — Portable Operating System Interface (POSIX) — Part 1: System Application Program Interface (API) — Amendment 7: Tracing [C Language].

POSIX.2b

P1003.2b, Standard for Information Technology — Portable Operating System Interface (POSIX) — Part 2: Shell and Utilities — Amendment.

POSIX.2d: 1994

IEEE Std 1003.2d-1994, IEEE Standard for Information Technology — Portable Operating System Interface (POSIX) — Part 2: Shell and Utilities — Amendment 1: Batch Environment.

Referenced Documents

POSIX.13:-1998

IEEE Std 1003.13:1998, IEEE Standard for Information Technology — Standardized Application Environment Profile (AEP) — POSIX Realtime Application Support.

Sarwate Article

Sarwate, Dilip V., *Computation of Cyclic Redundancy Checks via Table Lookup*, Communications of the ACM, Volume 30, No. 8, August 1988.

Sprung, Sha, and Lehoczky

Sprung, B., Sha, L., and Lehoczky, J.P., *Aperiodic Task Scheduling for Hard Real-Time Systems*, The Journal of Real-Time Systems, Volume 1, 1989, Pages 27-60.

SVID, Issue 1

American Telephone and Telegraph Company, System V Interface Definition (SVID), Issue 1; Morristown, NJ, UNIX Press, 1985.

SVID, Issue 2

American Telephone and Telegraph Company, System V Interface Definition (SVID), Issue 2; Morristown, NJ, UNIX Press, 1986.

SVID, Issue 3

American Telephone and Telegraph Company, System V Interface Definition (SVID), Issue 3; Morristown, NJ, UNIX Press, 1989.

The AWK Programming Language

Aho, Alfred V., Kernighan, Brian W., and Weinberger, Peter J., *The AWK Programming Language*, Reading, MA, Addison-Wesley 1988.

UNIX Programmer's Manual

American Telephone and Telegraph Company, *UNIX Time-Sharing System: UNIX Programmer's Manual*, 7th Edition, Murray Hill, NJ, Bell Telephone Laboratories, January 1979.

XNS, Issue 4

CAE Specification, August 1994, Networking Services, Issue 4 (ISBN: 1-85912-049-0, C438), published by The Open Group.

XNS, Issue 5

CAE Specification, February 1997, Networking Services, Issue 5 (ISBN: 1-85912-165-9, C523), published by The Open Group.

XNS, Issue 5.2

Technical Standard, January 2000, Networking Services (XNS), Issue 5.2 (ISBN: 1-85912-241-8, C808), published by The Open Group.

X/Open Courses, Issue 4, Version 2

CAE Specification, May 1996, X/Open Courses, Issue 4, Version 2 (ISBN: 1-85912-171-3, C610), published by The Open Group.

Yacc

Yacc: Yet Another Compiler Compiler, Stephen C. Johnson, 1978.

Source Documents

Parts of the following documents were used to create the base documents for this standard:

AIX 3.2 Manual

AIX Version 3.2 For RISC System/6000, Technical Reference: Base Operating System and Extensions, 1990, 1992 (Part No. SC23-2382-00).

OSF/1

OSF/1 Programmer's Reference, Release 1.2 (ISBN: 0-13-020579-6).

OSF AES

Application Environment Specification (AES) Operating System Programming Interfaces Volume, Revision A (ISBN: 0-13-043522-8).

System V Release 2.0

- UNIX System V Release 2.0 Programmer's Reference Manual (April 1984 - Issue 2).
- UNIX System V Release 2.0 Programming Guide (April 1984 - Issue 2).

System V Release 4.2

Operating System API Reference, UNIX SVR4.2 (1992) (ISBN: 0-13-017658-3).

1

Chapter 1

Introduction

1.1 Scope

The scope of IEEE Std 1003.1-2001 is described in the Base Definitions volume of IEEE Std 1003.1-2001.

1.2 Conformance

Conformance requirements for IEEE Std 1003.1-2001 are defined in the Base Definitions volume of IEEE Std 1003.1-2001, Chapter 2, Conformance.

1.3 Normative References

Normative references for IEEE Std 1003.1-2001 are defined in the Base Definitions volume of IEEE Std 1003.1-2001.

1.4 Change History

Change history is described in the Rationale (Informative) volume of IEEE Std 1003.1-2001, and in the CHANGE HISTORY section of reference pages.

1.5 Terminology

This section appears in the Base Definitions volume of IEEE Std 1003.1-2001, but is repeated here for convenience:

For the purposes of IEEE Std 1003.1-2001, the following terminology definitions apply:

can

Describes a permissible optional feature or behavior available to the user or application. The feature or behavior is mandatory for an implementation that conforms to IEEE Std 1003.1-2001. An application can rely on the existence of the feature or behavior.

implementation-defined

Describes a value or behavior that is not defined by IEEE Std 1003.1-2001 but is selected by an implementor. The value or behavior may vary among implementations that conform to IEEE Std 1003.1-2001. An application should not rely on the existence of the value or behavior. An application that relies on such a value or behavior cannot be assured to be portable across conforming implementations.

The implementor shall document such a value or behavior so that it can be used correctly by an application.

legacy

Describes a feature or behavior that is being retained for compatibility with older applications, but which has limitations which make it inappropriate for developing portable

33 applications. New applications should use alternative means of obtaining equivalent
34 functionality.

35 **may**
36 Describes a feature or behavior that is optional for an implementation that conforms to
37 IEEE Std 1003.1-2001. An application should not rely on the existence of the feature or
38 behavior. An application that relies on such a feature or behavior cannot be assured to be
39 portable across conforming implementations.

40 To avoid ambiguity, the opposite of *may* is expressed as *need not*, instead of *may not*.

41 **shall**
42 For an implementation that conforms to IEEE Std 1003.1-2001, describes a feature or
43 behavior that is mandatory. An application can rely on the existence of the feature or
44 behavior.

45 For an application or user, describes a behavior that is mandatory.

46 **should**
47 For an implementation that conforms to IEEE Std 1003.1-2001, describes a feature or
48 behavior that is recommended but not mandatory. An application should not rely on the
49 existence of the feature or behavior. An application that relies on such a feature or behavior
50 cannot be assured to be portable across conforming implementations.

51 For an application, describes a feature or behavior that is recommended programming
52 practice for optimum portability.

53 **undefined**
54 Describes the nature of a value or behavior not defined by IEEE Std 1003.1-2001 which
55 results from use of an invalid program construct or invalid data input.

56 The value or behavior may vary among implementations that conform to
57 IEEE Std 1003.1-2001. An application should not rely on the existence or validity of the
58 value or behavior. An application that relies on any particular value or behavior cannot be
59 assured to be portable across conforming implementations.

60 **unspecified**
61 Describes the nature of a value or behavior not specified by IEEE Std 1003.1-2001 which
62 results from use of a valid program construct or valid data input.

63 The value or behavior may vary among implementations that conform to
64 IEEE Std 1003.1-2001. An application should not rely on the existence or validity of the
65 value or behavior. An application that relies on any particular value or behavior cannot be
66 assured to be portable across conforming implementations.

67 **1.6 Definitions**

68 Concepts and definitions are defined in the Base Definitions volume of IEEE Std 1003.1-2001.

69 **1.7 Relationship to Other Documents**

70 **1.7.1 System Interfaces**

71 This subsection describes some of the features provided by the System Interfaces volume of
72 IEEE Std 1003.1-2001 that are assumed to be globally available on all systems conforming to this
73 volume of IEEE Std 1003.1-2001. This subsection does not attempt to detail all of the features
74 defined in the System Interfaces volume of IEEE Std 1003.1-2001 that are required by all of the
75 utilities defined in this volume of IEEE Std 1003.1-2001; the utility and function descriptions
76 point out additional functionality required to provide the corresponding specific features
77 needed by each.

78 The following subsections describe frequently used concepts. Many of these concepts are
79 described in the Base Definitions volume of IEEE Std 1003.1-2001. Utility and function
80 description statements override these defaults when appropriate.

81 **1.7.1.1 Process Attributes**

82 The following process attributes, as described in the System Interfaces volume of
83 IEEE Std 1003.1-2001, are assumed to be supported for all processes in this volume of
84 IEEE Std 1003.1-2001:

Controlling Terminal	Real Group ID
Current Working Directory	Real User ID
Effective Group ID	Root Directory
Effective User ID	Saved Set-Group-ID
File Descriptors	Saved Set-User-ID
File Mode Creation Mask	Session Membership
Process Group ID	Supplementary Group IDs
Process ID	

93 A conforming implementation may include additional process attributes.

94 **1.7.1.2 Concurrent Execution of Processes**

95 The following functionality of the *fork()* function defined in the System Interfaces volume of
96 IEEE Std 1003.1-2001 shall be available on all systems conforming to this volume of
97 IEEE Std 1003.1-2001:

- 98 1. Independent processes shall be capable of executing independently without either process
99 terminating.
- 100 2. A process shall be able to create a new process with all of the attributes referenced in
101 Section 1.7.1.1, determined according to the semantics of a call to the *fork()* function
102 defined in the System Interfaces volume of IEEE Std 1003.1-2001 followed by a call in the
103 child process to one of the *exec* functions defined in the System Interfaces volume of
104 IEEE Std 1003.1-2001.

105 1.7.1.3 File Access Permissions

106 The file access control mechanism described by the Base Definitions volume of
107 IEEE Std 1003.1-2001, Section 4.4, File Access Permissions shall apply to all files on an
108 implementation conforming to this volume of IEEE Std 1003.1-2001.

109 1.7.1.4 File Read, Write, and Creation

110 If a file that does not exist is to be written, it shall be created as described below, unless the
111 utility description states otherwise.

112 When a file that does not exist is created, the following features defined in the System Interfaces
113 volume of IEEE Std 1003.1-2001 shall apply unless the utility or function description states
114 otherwise:

- 115 1. The user ID of the file shall be set to the effective user ID of the calling process.
- 116 2. The group ID of the file shall be set to the effective group ID of the calling process or the
117 group ID of the directory in which the file is being created.
- 118 3. If the file is a regular file, the permission bits of the file shall be set to:

119 S_IROTH | S_IWOTH | S_IRGRP | S_IWGRP | S_IRUSR | S_IWUSR

120 (see the description of *File Modes* in the Base Definitions volume of IEEE Std 1003.1-2001,
121 Chapter 13, Headers, <sys/stat.h>) except that the bits specified by the file mode creation
122 mask of the process shall be cleared. If the file is a directory, the permission bits shall be set
123 to:

124 S_IRWXU | S_IRWXG | S_IRWXO

125 except that the bits specified by the file mode creation mask of the process shall be cleared.

- 126 4. The *st_atime*, *st_ctime*, and *st_mtime* fields of the file shall be updated as specified in the
127 System Interfaces volume of IEEE Std 1003.1-2001, Section 2.5, Standard I/O Streams.
- 128 5. If the file is a directory, it shall be an empty directory; otherwise, the file shall have length
129 zero.
- 130 6. If the file is a symbolic link, the effect shall be undefined unless the {POSIX2_SYMLINKS}
131 variable is in effect for the directory in which the symbolic link would be created.
- 132 7. Unless otherwise specified, the file created shall be a regular file.

133 When an attempt is made to create a file that already exists, the utility shall take the action
134 indicated in Table 1-1 (on page 5) corresponding to the type of the file the utility is trying to
135 create and the type of the existing file, unless the utility description states otherwise.

136

Table 1-1 Actions when Creating a File that Already Exists137
138
139
140
141
142
143
144
145
146
147
148
149
150

Existing Type	New Type										Function Creating New	2
	B	C	D	F	L	M	P	Q	R	S	T	
A <i>fattach()</i> -ed STREAM	F	F	F	F	—	—	—	OF	—	U	N/A	
B Block Special	F	F	F	F	F	U	U	U	OF	U	<i>mknod()</i> **	2
C Character Special	F	F	F	F	F	U	U	U	OF	U	<i>mknod()</i> **	2
D Directory	F	F	F	F	—	—	—	F	—	U	<i>mkdir()</i>	2
F FIFO Special File	F	F	F	F	—	—	—	O	—	U	<i>mkfifo()</i>	2
L Symbolic Link	F	F	F	F	—	—	—	FL	—	U	<i>symlink()</i>	2
M Shared Memory	F	F	F	F	—	—	—	—	—	U	<i>shm_open()</i>	2
P Semaphore	F	F	F	F	—	—	—	—	—	U	<i>sem_open()</i>	2
Q Message Queue	F	F	F	F	—	—	—	—	—	U	<i>mq_open()</i>	2
R Regular File	F	F	F	F	—	—	—	RF	—	U	<i>open()</i>	2
S Socket	F	F	F	F	—	—	—	—	—	U	<i>bind()</i>	2
T Typed Memory	F	F	F	F	F	U	U	U	U	U	*	

151

The following codes are used in Table 1-1:

152
153
154

F Fail. The attempt to create the new file shall fail and the utility shall either continue with its operation or exit immediately with a non-zero exit status, depending on the description of the utility.

155
156
157
158

FL Follow link. Unless otherwise specified, the symbolic link shall be followed as specified for pathname resolution, and the operation performed shall be as if the target of the symbolic link (after all resolution) had been named. If the target of the symbolic link does not exist, it shall be as if that nonexistent target had been named directly.

159
160

O Open FIFO. When attempting to create a regular file, and the existing file is a FIFO special file:

161
162

1. If the FIFO is not already open for reading, the attempt shall block until the FIFO is opened for reading.
2. Once the FIFO is open for reading, the utility shall open the FIFO for writing and continue with its operation.

165

OF The named file shall be opened with the consequences defined for that file type.

166

RF Regular file. When attempting to create a regular file, and the existing file is a regular file:

167
168

1. The user ID, group ID, and permission bits of the file shall not be changed.

169

2. The file shall be truncated to zero length.

170

3. The *st_ctime* and *st_mtime* fields shall be marked for update.

171

— The effect is implementation-defined unless specified by the utility description.

2

172

U The effect is unspecified unless specified by the utility description.

173

* There is no portable way to create a file of this type.

174

** Not portable.

175
176When a file is to be appended, the file shall be opened in a manner equivalent to using the O_APPEND flag, without the O_TRUNC flag, in the *open()* function defined in the System Interfaces volume of IEEE Std 1003.1-2001.177
178

When a file is to be read or written, the file shall be opened with an access mode corresponding to the operation to be performed. If file access permissions deny access, the requested operation

179 shall fail.

180 **1.7.1.5 File Removal**

181 When a directory that is the root directory or current working directory of any process is
182 removed, the effect is implementation-defined. If file access permissions deny access, the
183 requested operation shall fail. Otherwise, when a file is removed:

- 184 1. Its directory entry shall be removed from the file system.
- 185 2. The link count of the file shall be decremented.
- 186 3. If the file is an empty directory (see the Base Definitions volume of IEEE Std 1003.1-2001,
187 Section 3.143, Empty Directory):
 - 188 a. If no process has the directory open, the space occupied by the directory shall be
189 freed and the directory shall no longer be accessible.
 - 190 b. If one or more processes have the directory open, the directory contents shall be
191 preserved until all references to the file have been closed.
- 192 4. If the file is a directory that is not empty, the *st_ctime* field shall be marked for update.
- 193 5. If the file is not a directory:
 - 194 a. If the link count becomes zero:
 - 195 i. If no process has the file open, the space occupied by the file shall be freed and
196 the file shall no longer be accessible.
 - 197 ii. If one or more processes have the file open, the file contents shall be preserved
198 until all references to the file have been closed.
 - 199 b. If the link count is not reduced to zero, the *st_ctime* field shall be marked for update.
- 200 6. The *st_ctime* and *st_mtime* fields of the containing directory shall be marked for update.

201 **1.7.1.6 File Time Values**

202 All files shall have the three time values described by the Base Definitions volume of
203 IEEE Std 1003.1-2001, Section 4.7, File Times Update.

204 **1.7.1.7 File Contents**

205 When a reference is made to the contents of a file, *pathname*, this means the equivalent of all of
206 the data placed in the space pointed to by *buf* when performing the *read()* function calls in the
207 following operations defined in the System Interfaces volume of IEEE Std 1003.1-2001:

```
208     while (read (fildes, buf, nbytes) > 0)
209         ;
```

210 If the file is indicated by a pathname *pathname*, the file descriptor shall be determined by the
211 equivalent of the following operation defined in the System Interfaces volume of
212 IEEE Std 1003.1-2001:

```
213     fildes = open (pathname, O_RDONLY);
```

214 The value of *nbytes* in the above sequence is unspecified; if the file is of a type where the data
215 returned by *read()* would vary with different values, the value shall be one that results in the
216 most data being returned.

217 If the *read()* function calls would return an error, it is unspecified whether the contents of the file
218 are considered to include any data from offsets in the file beyond where the error would be
219 returned.

220 **1.7.1.8 Pathname Resolution**

221 The pathname resolution algorithm, described by the Base Definitions volume of
222 IEEE Std 1003.1-2001, Section 4.11, Pathname Resolution, shall be used by implementations
223 conforming to this volume of IEEE Std 1003.1-2001; see also the Base Definitions volume of
224 IEEE Std 1003.1-2001, Section 4.5, File Hierarchy.

225 **1.7.1.9 Changing the Current Working Directory**

226 When the current working directory (see the Base Definitions volume of IEEE Std 1003.1-2001,
227 Section 3.436, Working Directory) is to be changed, unless the utility or function description
228 states otherwise, the operation shall succeed unless a call to the *chdir()* function defined in the
229 System Interfaces volume of IEEE Std 1003.1-2001 would fail when invoked with the new
230 working directory pathname as its argument.

231 **1.7.1.10 Establish the Locale**

232 The functionality of the *setlocale()* function defined in the System Interfaces volume of
233 IEEE Std 1003.1-2001 shall be available on all systems conforming to this volume of
234 IEEE Std 1003.1-2001; that is, utilities that require the capability of establishing an international
235 operating environment shall be permitted to set the specified category of the international
236 environment.

237 **1.7.1.11 Actions Equivalent to Functions**

238 Some utility descriptions specify that a utility performs actions equivalent to a function defined
239 in the System Interfaces volume of IEEE Std 1003.1-2001. Such specifications require only that
240 the external effects be equivalent, not that any effect within the utility and visible only to the
241 utility be equivalent.

242 **1.7.2 Concepts Derived from the ISO C Standard**

243 Some of the standard utilities perform complex data manipulation using their own procedure
244 and arithmetic languages, as defined in their EXTENDED DESCRIPTION or OPERANDS
245 sections. Unless otherwise noted, the arithmetic and semantic concepts (precision, type
246 conversion, control flow, and so on) shall be equivalent to those defined in the ISO C standard,
247 as described in the following sections. Note that there is no requirement that the standard
248 utilities be implemented in any particular programming language.

249 **1.7.2.1 Arithmetic Precision and Operations**

250 Integer variables and constants, including the values of operands and option-arguments, used
251 by the standard utilities listed in this volume of IEEE Std 1003.1-2001 shall be implemented as
252 equivalent to the ISO C standard **signed long** data type; floating point shall be implemented as
253 equivalent to the ISO C standard **double** type. Conversions between types shall be as described
254 in the ISO C standard. All variables shall be initialized to zero if they are not otherwise assigned
255 by the input to the application.

256 Arithmetic operators and control flow keywords shall be implemented as equivalent to those in
257 the cited ISO C standard section, as listed in Table 1-2 (on page 8).

Table 1-2 Selected ISO C Standard Operators and Control Flow Keywords

Operation	ISO C Standard Equivalent Reference
<code>()</code>	Section 6.5.1, Primary Expressions
<code>postfix ++</code> <code>postfix --</code>	Section 6.5.2, Postfix Operators
<code>unary +</code> <code>unary -</code> <code>prefix ++</code> <code>prefix --</code> <code>~</code> <code>!</code> <code>sizeof()</code>	Section 6.5.3, Unary Operators
<code>*</code> <code>/</code> <code>%</code>	Section 6.5.5, Multiplicative Operators
<code>+</code> <code>-</code>	Section 6.5.6, Additive Operators
<code><<</code> <code>>></code>	Section 6.5.7, Bitwise Shift Operators
<code><, <=</code> <code>>, >=</code>	Section 6.5.8, Relational Operators
<code>==</code> <code>!=</code>	Section 6.5.9, Equality Operators
<code>&</code>	Section 6.5.10, Bitwise AND Operator
<code>^</code>	Section 6.5.11, Bitwise Exclusive OR Operator
<code> </code>	Section 6.5.12, Bitwise Inclusive OR Operator
<code>&&</code>	Section 6.5.13, Logical AND Operator
<code> </code>	Section 6.5.14, Logical OR Operator
<code>expr?expr:expr</code>	Section 6.5.15, Conditional Operator
<code>=, *=, /=, %=, +=, -=</code> <code><<=, >>=, &=, ^=, =</code>	Section 6.5.16, Assignment Operators
<code>if ()</code> <code>if () ... else</code> <code>switch ()</code>	Section 6.8.4, Selection Statements
<code>while ()</code> <code>do ... while ()</code> <code>for ()</code>	Section 6.8.5, Iteration Statements
<code>goto</code> <code>continue</code> <code>break</code> <code>return</code>	Section 6.8.6, Jump Statements

The evaluation of arithmetic expressions shall be equivalent to that described in Section 6.5, Expressions, of the ISO C standard.

302 1.7.2.2 Mathematical Functions

303 Any mathematical functions with the same names as those in the following sections of the ISO C
304 standard:

- 305 • Section 7.12, Mathematics, <math.h>
- 306 • Section 7.20.2, Pseudo-Random Sequence Generation Functions

307 shall be implemented to return the results equivalent to those returned from a call to the
308 corresponding function described in the ISO C standard.

309 1.8 Portability

310 Some of the utilities in the Shell and Utilities volume of IEEE Std 1003.1-2001 and functions in
311 the System Interfaces volume of IEEE Std 1003.1-2001 describe functionality that might not be
312 fully portable to systems meeting the requirements for POSIX conformance (see the Base
313 Definitions volume of IEEE Std 1003.1-2001, Chapter 2, Conformance).

314 Where optional, enhanced, or reduced functionality is specified, the text is shaded and a code in
315 the margin identifies the nature of the option, extension, or warning (see Section 1.8.1). For
316 maximum portability, an application should avoid such functionality.

317 Unless the primary task of a utility is to produce textual material on its standard output,
318 application developers should not rely on the format or content of any such material that may be
319 produced. Where the primary task *is* to provide such material, but the output format is
320 incompletely specified, the description is marked with the OF margin code and shading.
321 Application developers are warned not to expect that the output of such an interface on one
322 system is any guide to its behavior on another system.

323 1.8.1 Codes

324 Codes and their meanings are listed in the Base Definitions volume of IEEE Std 1003.1-2001, but
325 are repeated here for convenience:

326 ADV Advisory Information

327 The functionality described is optional. The functionality described is also an extension to the
328 ISO C standard.

329 Where applicable, functions are marked with the ADV margin legend in the SYNOPSIS section.
330 Where additional semantics apply to a function, the material is identified by use of the ADV
331 margin legend.

332 AIO Asynchronous Input and Output

333 The functionality described is optional. The functionality described is also an extension to the
334 ISO C standard.

335 Where applicable, functions are marked with the AIO margin legend in the SYNOPSIS section.
336 Where additional semantics apply to a function, the material is identified by use of the AIO
337 margin legend.

338 BAR Barriers

339 The functionality described is optional. The functionality described is also an extension to the
340 ISO C standard.

341 Where applicable, functions are marked with the BAR margin legend in the SYNOPSIS section.
342 Where additional semantics apply to a function, the material is identified by use of the BAR
343 margin legend.

344	BE	Batch Environment Services and Utilities
345		The functionality described is optional.
346		Where applicable, utilities are marked with the BE margin legend in the SYNOPSIS section.
347		Where additional semantics apply to a utility, the material is identified by use of the BE margin legend.
349	CD	C-Language Development Utilities
350		The functionality described is optional.
351		Where applicable, utilities are marked with the CD margin legend in the SYNOPSIS section.
352		Where additional semantics apply to a utility, the material is identified by use of the CD margin legend.
354	CPT	Process CPU-Time Clocks
355		The functionality described is optional. The functionality described is also an extension to the ISO C standard.
357		Where applicable, functions are marked with the CPT margin legend in the SYNOPSIS section.
358		Where additional semantics apply to a function, the material is identified by use of the CPT margin legend.
360	CS	Clock Selection
361		The functionality described is optional. The functionality described is also an extension to the ISO C standard.
363		Where applicable, functions are marked with the CS margin legend in the SYNOPSIS section.
364		Where additional semantics apply to a function, the material is identified by use of the CS margin legend.
366	CX	Extension to the ISO C standard
367		The functionality described is an extension to the ISO C standard. Application writers may make use of an extension as it is supported on all IEEE Std 1003.1-2001-conforming systems.
369		With each function or header from the ISO C standard, a statement to the effect that “any conflict is unintentional” is included. That is intended to refer to a direct conflict. IEEE Std 1003.1-2001 acts in part as a profile of the ISO C standard, and it may choose to further constrain behaviors allowed to vary by the ISO C standard. Such limitations are not considered conflicts.
374		Where additional semantics apply to a function or header, the material is identified by use of the CX margin legend.
376	FD	FORTRAN Development Utilities
377		The functionality described is optional.
378		Where applicable, utilities are marked with the FD margin legend in the SYNOPSIS section.
379		Where additional semantics apply to a utility, the material is identified by use of the FD margin legend.
381	FR	FORTRAN Runtime Utilities
382		The functionality described is optional.
383		Where applicable, utilities are marked with the FR margin legend in the SYNOPSIS section.
384		Where additional semantics apply to a utility, the material is identified by use of the FR margin legend.
386	FSC	File Synchronization
387		The functionality described is optional. The functionality described is also an extension to the ISO C standard.

389	Where applicable, functions are marked with the FSC margin legend in the SYNOPSIS section.
390	Where additional semantics apply to a function, the material is identified by use of the FSC margin legend.
391	
392	IPV6
393	The functionality described is optional. The functionality described is also an extension to the ISO C standard.
394	
395	Where applicable, functions are marked with the IP6 margin legend in the SYNOPSIS section.
396	Where additional semantics apply to a function, the material is identified by use of the IP6 margin legend.
397	
398	MC1 Advisory Information and either Memory Mapped Files or Shared Memory Objects
399	The functionality described is optional. The functionality described is also an extension to the ISO C standard.
400	
401	This is a shorthand notation for combinations of multiple option codes.
402	Where applicable, functions are marked with the MC1 margin legend in the SYNOPSIS section.
403	Where additional semantics apply to a function, the material is identified by use of the MC1 margin legend.
404	
405	Refer to the Base Definitions volume of IEEE Std 1003.1-2001, Section 1.5.2, Margin Code Notation.
406	
407	MC2 Memory Mapped Files, Shared Memory Objects, or Memory Protection
408	The functionality described is optional. The functionality described is also an extension to the ISO C standard.
409	
410	This is a shorthand notation for combinations of multiple option codes.
411	Where applicable, functions are marked with the MC2 margin legend in the SYNOPSIS section.
412	Where additional semantics apply to a function, the material is identified by use of the MC2 margin legend.
413	
414	Refer to the Base Definitions volume of IEEE Std 1003.1-2001, Section 1.5.2, Margin Code Notation.
415	
416	MC3 Memory Mapped Files, Shared Memory Objects, or Typed Memory Objects
417	The functionality described is optional. The functionality described is also an extension to the ISO C standard.
418	
419	This is a shorthand notation for combinations of multiple option codes.
420	Where applicable, functions are marked with the MC3 margin legend in the SYNOPSIS section.
421	Where additional semantics apply to a function, the material is identified by use of the MC3 margin legend.
422	
423	Refer to the Base Definitions volume of IEEE Std 1003.1-2001, Section 1.5.2, Margin Code Notation.
424	
425	MF Memory Mapped Files
426	The functionality described is optional. The functionality described is also an extension to the ISO C standard.
427	
428	Where applicable, functions are marked with the MF margin legend in the SYNOPSIS section.
429	Where additional semantics apply to a function, the material is identified by use of the MF margin legend.
430	
431	ML Process Memory Locking
432	The functionality described is optional. The functionality described is also an extension to the

433	ISO C standard.
434	Where applicable, functions are marked with the ML margin legend in the SYNOPSIS section.
435	Where additional semantics apply to a function, the material is identified by use of the ML margin legend.
436	
437	Range Memory Locking
438	The functionality described is optional. The functionality described is also an extension to the ISO C standard.
439	
440	Where applicable, functions are marked with the MLR margin legend in the SYNOPSIS section.
441	Where additional semantics apply to a function, the material is identified by use of the MLR margin legend.
442	
443	Monotonic Clock
444	The functionality described is optional. The functionality described is also an extension to the ISO C standard.
445	
446	Where applicable, functions are marked with the MON margin legend in the SYNOPSIS section.
447	Where additional semantics apply to a function, the material is identified by use of the MON margin legend.
448	
449	Memory Protection
450	The functionality described is optional. The functionality described is also an extension to the ISO C standard.
451	
452	Where applicable, functions are marked with the MPR margin legend in the SYNOPSIS section.
453	Where additional semantics apply to a function, the material is identified by use of the MPR margin legend.
454	
455	Message Passing
456	The functionality described is optional. The functionality described is also an extension to the ISO C standard.
457	
458	Where applicable, functions are marked with the MSG margin legend in the SYNOPSIS section.
459	Where additional semantics apply to a function, the material is identified by use of the MSG margin legend.
460	
461	IEC 60559 Floating-Point Option
462	The functionality described is optional. The functionality described is also an extension to the ISO C standard.
463	
464	Where applicable, functions are marked with the MX margin legend in the SYNOPSIS section.
465	Where additional semantics apply to a function, the material is identified by use of the MX margin legend.
466	
467	Obsolescent
468	The functionality described may be withdrawn in a future version of this volume of IEEE Std 1003.1-2001. Strictly Conforming POSIX Applications and Strictly Conforming XSI Applications shall not use obsolescent features.
469	
470	
471	Where applicable, the material is identified by use of the OB margin legend.
472	Output Format Incompletely Specified
473	The functionality described is an XSI extension. The format of the output produced by the utility is not fully specified. It is therefore not possible to post-process this output in a consistent fashion. Typical problems include unknown length of strings and unspecified field delimiters.
474	
475	
476	Where applicable, the material is identified by use of the OF margin legend.

477	OH	Optional Header
478		In the SYNOPSIS section of some interfaces in the System Interfaces volume of IEEE Std 1003.1-2001 an included header is marked as in the following example:
480	OH	#include <sys/types.h> #include <grp.h> struct group *getgrnam(const char *name);
483		The OH margin legend indicates that the marked header is not required on XSI-conformant systems.
485	PIO	Prioritized Input and Output
486		The functionality described is optional. The functionality described is also an extension to the ISO C standard.
488		Where applicable, functions are marked with the PIO margin legend in the SYNOPSIS section. Where additional semantics apply to a function, the material is identified by use of the PIO margin legend.
491	PS	Process Scheduling
492		The functionality described is optional. The functionality described is also an extension to the ISO C standard.
494		Where applicable, functions are marked with the PS margin legend in the SYNOPSIS section. Where additional semantics apply to a function, the material is identified by use of the PS margin legend.
497	RS	Raw Sockets
498		The functionality described is optional. The functionality described is also an extension to the ISO C standard.
500		Where applicable, functions are marked with the RS margin legend in the SYNOPSIS section. Where additional semantics apply to a function, the material is identified by use of the RS margin legend.
503	RTS	Realtime Signals Extension
504		The functionality described is optional. The functionality described is also an extension to the ISO C standard.
506		Where applicable, functions are marked with the RTS margin legend in the SYNOPSIS section. Where additional semantics apply to a function, the material is identified by use of the RTS margin legend.
509	SD	Software Development Utilities
510		The functionality described is optional.
511		Where applicable, utilities are marked with the SD margin legend in the SYNOPSIS section. Where additional semantics apply to a utility, the material is identified by use of the SD margin legend.
514	SEM	Semaphores
515		The functionality described is optional. The functionality described is also an extension to the ISO C standard.
517		Where applicable, functions are marked with the SEM margin legend in the SYNOPSIS section. Where additional semantics apply to a function, the material is identified by use of the SEM margin legend.
520	SHM	Shared Memory Objects
521		The functionality described is optional. The functionality described is also an extension to the

522	ISO C standard.
523	Where applicable, functions are marked with the SHM margin legend in the SYNOPSIS section.
524	Where additional semantics apply to a function, the material is identified by use of the SHM
525	margin legend.
526	SIO Synchronized Input and Output
527	The functionality described is optional. The functionality described is also an extension to the
528	ISO C standard.
529	Where applicable, functions are marked with the SIO margin legend in the SYNOPSIS section.
530	Where additional semantics apply to a function, the material is identified by use of the SIO
531	margin legend.
532	SPI Spin Locks
533	The functionality described is optional. The functionality described is also an extension to the
534	ISO C standard.
535	Where applicable, functions are marked with the SPI margin legend in the SYNOPSIS section.
536	Where additional semantics apply to a function, the material is identified by use of the SPI
537	margin legend.
538	SPN Spawn
539	The functionality described is optional. The functionality described is also an extension to the
540	ISO C standard.
541	Where applicable, functions are marked with the SPN margin legend in the SYNOPSIS section.
542	Where additional semantics apply to a function, the material is identified by use of the SPN
543	margin legend.
544	SS Process Sporadic Server
545	The functionality described is optional. The functionality described is also an extension to the
546	ISO C standard.
547	Where applicable, functions are marked with the SS margin legend in the SYNOPSIS section.
548	Where additional semantics apply to a function, the material is identified by use of the SS
549	margin legend.
550	TCT Thread CPU-Time Clocks
551	The functionality described is optional. The functionality described is also an extension to the
552	ISO C standard.
553	Where applicable, functions are marked with the TCT margin legend in the SYNOPSIS section.
554	Where additional semantics apply to a function, the material is identified by use of the TCT
555	margin legend.
556	TEF Trace Event Filter
557	The functionality described is optional. The functionality described is also an extension to the
558	ISO C standard.
559	Where applicable, functions are marked with the TEF margin legend in the SYNOPSIS section.
560	Where additional semantics apply to a function, the material is identified by use of the TEF
561	margin legend.
562	THR Threads
563	The functionality described is optional. The functionality described is also an extension to the
564	ISO C standard.
565	Where applicable, functions are marked with the THR margin legend in the SYNOPSIS section.
566	Where additional semantics apply to a function, the material is identified by use of the THR

567		margin legend.
568	TMO	Timeouts The functionality described is optional. The functionality described is also an extension to the ISO C standard.
571		Where applicable, functions are marked with the TMO margin legend in the SYNOPSIS section.
572		Where additional semantics apply to a function, the material is identified by use of the TMO margin legend.
574	TMR	Timers The functionality described is optional. The functionality described is also an extension to the ISO C standard.
577		Where applicable, functions are marked with the TMR margin legend in the SYNOPSIS section.
578		Where additional semantics apply to a function, the material is identified by use of the TMR margin legend.
580	TPI	Thread Priority Inheritance The functionality described is optional. The functionality described is also an extension to the ISO C standard.
583		Where applicable, functions are marked with the TPI margin legend in the SYNOPSIS section.
584		Where additional semantics apply to a function, the material is identified by use of the TPI margin legend.
586	TPP	Thread Priority Protection The functionality described is optional. The functionality described is also an extension to the ISO C standard.
589		Where applicable, functions are marked with the TPP margin legend in the SYNOPSIS section.
590		Where additional semantics apply to a function, the material is identified by use of the TPP margin legend.
592	TPS	Thread Execution Scheduling The functionality described is optional. The functionality described is also an extension to the ISO C standard.
595		Where applicable, functions are marked with the TPS margin legend for the SYNOPSIS section.
596		Where additional semantics apply to a function, the material is identified by use of the TPS margin legend.
598	TRC	Trace The functionality described is optional. The functionality described is also an extension to the ISO C standard.
601		Where applicable, functions are marked with the TRC margin legend in the SYNOPSIS section.
602		Where additional semantics apply to a function, the material is identified by use of the TRC margin legend.
604	TRI	Trace Inherit The functionality described is optional. The functionality described is also an extension to the ISO C standard.
607		Where applicable, functions are marked with the TRI margin legend in the SYNOPSIS section.
608		Where additional semantics apply to a function, the material is identified by use of the TRI margin legend.
610	TRL	Trace Log The functionality described is optional. The functionality described is also an extension to the

612	ISO C standard.
613	Where applicable, functions are marked with the TRL margin legend in the SYNOPSIS section.
614	Where additional semantics apply to a function, the material is identified by use of the TRL margin legend.
615	
616	TSA
617	Thread Stack Address Attribute
618	The functionality described is optional. The functionality described is also an extension to the ISO C standard.
619	Where applicable, functions are marked with the TSA margin legend for the SYNOPSIS section.
620	Where additional semantics apply to a function, the material is identified by use of the TSA margin legend.
621	
622	TSF
623	Thread-Safe Functions
624	The functionality described is optional. The functionality described is also an extension to the ISO C standard.
625	Where applicable, functions are marked with the TSF margin legend in the SYNOPSIS section.
626	Where additional semantics apply to a function, the material is identified by use of the TSF margin legend.
627	
628	TSH
629	Thread Process-Shared Synchronization
630	The functionality described is optional. The functionality described is also an extension to the ISO C standard.
631	Where applicable, functions are marked with the TSH margin legend in the SYNOPSIS section.
632	Where additional semantics apply to a function, the material is identified by use of the TSH margin legend.
633	
634	TSP
635	Thread Sporadic Server
636	The functionality described is optional. The functionality described is also an extension to the ISO C standard.
637	Where applicable, functions are marked with the TSP margin legend in the SYNOPSIS section.
638	Where additional semantics apply to a function, the material is identified by use of the TSP margin legend.
639	
640	TSS
641	Thread Stack Size Attribute
642	The functionality described is optional. The functionality described is also an extension to the ISO C standard.
643	Where applicable, functions are marked with the TSS margin legend in the SYNOPSIS section.
644	Where additional semantics apply to a function, the material is identified by use of the TSS margin legend.
645	
646	TYM
647	Typed Memory Objects
648	The functionality described is optional. The functionality described is also an extension to the ISO C standard.
649	Where applicable, functions are marked with the TYM margin legend in the SYNOPSIS section.
650	Where additional semantics apply to a function, the material is identified by use of the TYM margin legend.
651	
652	UP
653	User Portability Utilities
654	The functionality described is optional.
655	Where applicable, utilities are marked with the UP margin legend in the SYNOPSIS section.
656	Where additional semantics apply to a utility, the material is identified by use of the UP margin legend.

657	XSI	Extension
658		The functionality described is an XSI extension. Functionality marked XSI is also an extension to the ISO C standard. Application writers may confidently make use of an extension on all systems supporting the X/Open System Interfaces Extension.
661		If an entire SYNOPSIS section is shaded and marked XSI, all the functionality described in that reference page is an extension. See the Base Definitions volume of IEEE Std 1003.1-2001, Section 3.439, XSI.
664	XSR	XSI STREAMS
665		The functionality described is optional. The functionality described is also an extension to the ISO C standard.
667		Where applicable, functions are marked with the XSR margin legend in the SYNOPSIS section. Where additional semantics apply to a function, the material is identified by use of the XSR margin legend.
669		

670 1.9 Utility Limits

671 This section lists magnitude limitations imposed by a specific implementation. The braces
 672 notation, {LIMIT}, is used in this volume of IEEE Std 1003.1-2001 to indicate these values, but the
 673 braces are not part of the name.

674 **Table 1-3 Utility Limit Minimum Values**

676 Name	677 Description	678 Value
677 {POSIX2_BC_BASE_MAX}	678 The maximum <i>obase</i> value allowed by the <i>bc</i> utility.	99
679 {POSIX2_BC_DIM_MAX}	680 The maximum number of elements permitted in an array by the <i>bc</i> utility.	2 048
681 {POSIX2_BC_SCALE_MAX}	682 The maximum <i>scale</i> value allowed by the <i>bc</i> utility.	99
683 {POSIX2_BC_STRING_MAX}	684 The maximum length of a string constant accepted by the <i>bc</i> utility.	1 000
685 {POSIX2_COLL_WEIGHTS_MAX}	686 The maximum number of weights that can be assigned to an entry of the <i>LC_COLLATE</i> order keyword in the locale definition file; see the border_start keyword in the Base Definitions volume of IEEE Std 1003.1-2001, Section 7.3.2, <i>LC_COLLATE</i> .	2
691 {POSIX2_EXPR_NEST_MAX}	692 The maximum number of expressions that can be nested within parentheses by the <i>expr</i> utility.	32
693 {POSIX2_LINE_MAX}	694 Unless otherwise noted, the maximum length, in bytes, of the input line of a utility (either standard input or another file), when the utility is described as processing text files. The length includes room for the trailing <newline>.	2 048
695		
696		
697		

Name	Description	Value
{POSIX2_RE_DUP_MAX}	The maximum number of repeated occurrences of a BRE permitted when using the interval notation $\{m,n\}$; see the Base Definitions volume of IEEE Std 1003.1-2001, Section 9.3.6, BREs Matching Multiple Characters.	255

The values specified in Table 1-3 (on page 17) represent the lowest values conforming implementations shall provide and, consequently, the largest values on which an application can rely without further enquiries, as described below. These values shall be accessible to applications via the *getconf* utility (see *getconf* (on page 484)).

1

Implementations may provide more liberal, or less restrictive, values than shown in Table 1-3 (on page 17). These possibly more liberal values are accessible using the symbols in Table 1-4.

The *sysconf()* function defined in the System Interfaces volume of IEEE Std 1003.1-2001 or the *getconf* utility return the value of each symbol on each specific implementation. The value so retrieved is the largest, or most liberal, value that is available throughout the session lifetime, as determined at session creation. The literal names shown in the table apply only to the *getconf* utility; the high-level language binding describes the exact form of each name to be used by the interfaces in that binding.

All numeric limits defined by the System Interfaces volume of IEEE Std 1003.1-2001, such as {PATH_MAX}, shall also apply to this volume of IEEE Std 1003.1-2001. All the utilities defined by this volume of IEEE Std 1003.1-2001 are implicitly limited by these values, unless otherwise noted in the utility descriptions.

It is not guaranteed that the application can actually reach the specified limit of an implementation in any given case, or at all, as a lack of virtual memory or other resources may prevent this. The limit value indicates only that the implementation does not specifically impose any arbitrary, more restrictive limit.

Table 1-4 Symbolic Utility Limits

Name	Description	Minimum Value
{BC_BASE_MAX}	The maximum <i>obase</i> value allowed by the <i>bc</i> utility.	{POSIX2_BC_BASE_MAX}
{BC_DIM_MAX}	The maximum number of elements permitted in an array by the <i>bc</i> utility.	{POSIX2_BC_DIM_MAX}
{BC_SCALE_MAX}	The maximum <i>scale</i> value allowed by the <i>bc</i> utility.	{POSIX2_BC_SCALE_MAX}
{BC_STRING_MAX}	The maximum length of a string constant accepted by the <i>bc</i> utility.	{POSIX2_BC_STRING_MAX}
{COLL_WEIGHTS_MAX}	The maximum number of weights that can be assigned to an entry of the <i>LC_COLLATE</i> order keyword in the locale definition file; see the	{POSIX2_COLL_WEIGHTS_MAX}

Name	Description	Minimum Value
{EXPR_NEST_MAX}	order_start keyword in the Base Definitions volume of IEEE Std 1003.1-2001, Section 7.3.2, LC_COLLATE. The maximum number of expressions that can be nested within parentheses by the <i>expr</i> utility.	{POSIX2_EXPR_NEST_MAX}
{LINE_MAX}	Unless otherwise noted, the maximum length, in bytes, of the input line of a utility (either standard input or another file), when the utility is described as processing text files. The length includes room for the trailing <newline>.	{POSIX2_LINE_MAX}
{RE_DUP_MAX}	The maximum number of repeated occurrences of a BRE permitted when using the interval notation $\backslash\{m,n\}$; see the Base Definitions volume of IEEE Std 1003.1-2001, Section 9.3.6, BREs Matching Multiple Characters.	{POSIX2_RE_DUP_MAX}

The following value may be a constant within an implementation or may vary from one pathname to another.

{POSIX2_SYMLINKS}

When referring to a directory, the system supports the creation of symbolic links within that directory; for non-directory files, the meaning of {POSIX2_SYMLINKS} is undefined.

1.10 Grammar Conventions

Portions of this volume of IEEE Std 1003.1-2001 are expressed in terms of a special grammar notation. It is used to portray the complex syntax of certain program input. The grammar is based on the syntax used by the *yacc* utility. However, it does not represent fully functional *yacc* input, suitable for program use; the lexical processing and all semantic requirements are described only in textual form. The grammar is not based on source used in any traditional implementation and has not been tested with the semantic code that would normally be required to accompany it. Furthermore, there is no implication that the partial *yacc* code presented represents the most efficient, or only, means of supporting the complex syntax within the utility. Implementations may use other programming languages or algorithms, as long as the syntax supported is the same as that represented by the grammar.

The following typographical conventions are used in the grammar; they have no significance except to aid in reading.

- 791 • The identifiers for the reserved words of the language are shown with a leading capital letter.
792 (These are terminals in the grammar; for example, **While**, **Case**.)
793 • The identifiers for terminals in the grammar are all named with uppercase letters and
794 underscores; for example, **NEWLINE**, **ASSIGN_OP**, **NAME**.
795 • The identifiers for non-terminals are all lowercase.

796 1.11 Utility Description Defaults

797 This section describes all of the subsections used within the utility descriptions, including:

- 798 • Intended usage of the section
799 • Global defaults that affect all the standard utilities
800 • The meanings of notations used in this volume of IEEE Std 1003.1-2001 that are specific to
801 individual utility sections

802 NAME

803 This section gives the name or names of the utility and briefly states its purpose.

804 SYNOPSIS

805 The SYNOPSIS section summarizes the syntax of the calling sequence for the utility,
806 including options, option-arguments, and operands. Standards for utility naming are
807 described in the Base Definitions volume of IEEE Std 1003.1-2001, Section 12.2, Utility
808 Syntax Guidelines; for describing the utility's arguments in the Base Definitions volume
809 of IEEE Std 1003.1-2001, Section 12.1, Utility Argument Syntax.

810 DESCRIPTION

811 The DESCRIPTION section describes the actions of the utility. If the utility has a very
812 complex set of subcommands or its own procedural language, an EXTENDED
813 DESCRIPTION section is also provided. Most explanations of optional functionality are
814 omitted here, as they are usually explained in the OPTIONS section.

815 As stated in Section 1.7.1.11 (on page 7), some functions are described in terms of
816 equivalent functionality. When specific functions are cited, the implementation shall
817 provide equivalent functionality including side effects associated with successful
818 execution of the function. The treatment of errors and intermediate results from the
819 individual functions cited is generally not specified by this volume of
820 IEEE Std 1003.1-2001. See the utility's EXIT STATUS and CONSEQUENCES OF
821 ERRORS sections for all actions associated with errors encountered by the utility.

822 OPTIONS

823 The OPTIONS section describes the utility options and option-arguments, and how
824 they modify the actions of the utility. Standard utilities that have options either fully
825 comply with the Base Definitions volume of IEEE Std 1003.1-2001, Section 12.2, Utility
826 Syntax Guidelines or describe all deviations. Apparent disagreements between
827 functionality descriptions in the OPTIONS and DESCRIPTION (or EXTENDED
828 DESCRIPTION) sections are always resolved in favor of the OPTIONS section.

829 Each OPTIONS section that uses the phrase "The ... utility shall conform to the Utility
830 Syntax Guidelines ..." refers only to the use of the utility as specified by this volume of
831 IEEE Std 1003.1-2001; implementation extensions should also conform to the
832 guidelines, but may allow exceptions for historical practice.

833 Unless otherwise stated in the utility description, when given an option unrecognized
834 by the implementation, or when a required option-argument is not provided, standard
835 utilities shall issue a diagnostic message to standard error and exit with a non-zero exit
836 status.

837 All utilities in this volume of IEEE Std 1003.1-2001 shall be capable of processing
838 arguments using eight-bit transparency.

839 **Default Behavior:** When this section is listed as “None.”, it means that the
840 implementation need not support any options. Standard utilities that do not accept
841 options, but that do accept operands, shall recognize “--” as a first argument to be
842 discarded.

843 The requirement for recognizing “--” is because conforming applications need a way
844 to shield their operands from any arbitrary options that the implementation may
845 provide as an extension. For example, if the standard utility *foo* is listed as taking no
846 options, and the application needed to give it a pathname with a leading hyphen, it
847 could safely do it as:

848 `foo -- myfile`

849 and avoid any problems with **-m** used as an extension.

850 OPERANDS

851 The OPERANDS section describes the utility operands, and how they affect the actions
852 of the utility. Apparent disagreements between functionality descriptions in the
853 OPERANDS and DESCRIPTION (or EXTENDED DESCRIPTION) sections shall be
854 resolved in favor of the OPERANDS section.

855 If an operand naming a file can be specified as ‘-’, which means to use the standard
856 input instead of a named file, this is explicitly stated in this section. Unless otherwise
857 stated, the use of multiple instances of ‘-’ to mean standard input in a single
858 command produces unspecified results.

859 Unless otherwise stated, the standard utilities that accept operands shall process those
860 operands in the order specified in the command line.

861 **Default Behavior:** When this section is listed as “None.”, it means that the
862 implementation need not support any operands.

863 STDIN

864 The STDIN section describes the standard input of the utility. This section is frequently
865 merely a reference to the following section, as many utilities treat standard input and
866 input files in the same manner. Unless otherwise stated, all restrictions described in the
867 INPUT FILES section shall apply to this section as well.

868 Use of a terminal for standard input can cause any of the standard utilities that read
869 standard input to stop when used in the background. For this reason, applications
870 should not use interactive features in scripts to be placed in the background.

871 The specified standard input format of the standard utilities shall not depend on the
872 existence or value of the environment variables defined in this volume of
873 IEEE Std 1003.1-2001, except as provided by this volume of IEEE Std 1003.1-2001.

874 **Default Behavior:** When this section is listed as “Not used.”, it means that the
875 standard input shall not be read when the utility is used as described by this volume of
876 IEEE Std 1003.1-2001.

877 **INPUT FILES**

878 The INPUT FILES section describes the files, other than the standard input, used as
 879 input by the utility. It includes files named as operands and option-arguments as well
 880 as other files that are referred to, such as start-up and initialization files, databases, and
 881 so on. Commonly-used files are generally described in one place and cross-referenced
 882 by other utilities.

883 All utilities in this volume of IEEE Std 1003.1-2001 shall be capable of processing input
 884 files using eight-bit transparency.

885 When a standard utility reads a seekable input file and terminates without an error
 886 before it reaches end-of-file, the utility shall ensure that the file offset in the open file
 887 description is properly positioned just past the last byte processed by the utility. For
 888 files that are not seekable, the state of the file offset in the open file description for that
 889 file is unspecified. A conforming application shall not assume that the following three
 890 commands are equivalent:

```
891           tail -n +2 file
  892           (sed -n 1q; cat) < file
  893           cat file | (sed -n 1q; cat)
```

894 The second command is equivalent to the first only when the file is seekable. The third
 895 command leaves the file offset in the open file description in an unspecified state. Other
 896 utilities, such as *head*, *read*, and *sh*, have similar properties.

897 Some of the standard utilities, such as filters, process input files a line or a block at a
 898 time and have no restrictions on the maximum input file size. Some utilities may have
 899 size limitations that are not as obvious as file space or memory limitations. Such
 900 limitations should reflect resource limitations of some sort, not arbitrary limits set by
 901 implementors. Implementations shall document those utilities that are limited by
 902 constraints other than file system space, available memory, and other limits specifically
 903 cited by this volume of IEEE Std 1003.1-2001, and identify what the constraint is and
 904 indicate a way of estimating when the constraint would be reached. Similarly, some
 905 utilities descend the directory tree (recursively). Implementations shall also document
 906 any limits that they may have in descending the directory tree that are beyond limits
 907 cited by this volume of IEEE Std 1003.1-2001.

908 When an input file is described as a “text file”, the utility produces undefined results if
 909 given input that is not from a text file, unless otherwise stated. Some utilities (for
 910 example, *make*, *read*, *sh*) allow for continued input lines using an escaped <newline>
 911 convention; unless otherwise stated, the utility need not be able to accumulate more
 912 than {LINE_MAX} bytes from a set of multiple, continued input lines. Thus, for a
 913 conforming application the total of all the continued lines in a set cannot exceed
 914 {LINE_MAX}. If a utility using the escaped <newline> convention detects an end-of-
 915 file condition immediately after an escaped <newline>, the results are unspecified.

916 Record formats are described in a notation similar to that used by the C-language
 917 function, *printf()*. See the Base Definitions volume of IEEE Std 1003.1-2001, Chapter 5,
 918 File Format Notation for a description of this notation. The format description is
 919 intended to be sufficiently rigorous to allow other applications to generate these input
 920 files. However, since <blank>s can legitimately be included in some of the fields
 921 described by the standard utilities, particularly in locales other than the POSIX locale,
 922 this intent is not always realized.

923 **Default Behavior:** When this section is listed as “None.”, it means that no input files
 924 are required to be supplied when the utility is used as described by this volume of

925 IEEE Std 1003.1-2001.

926 **ENVIRONMENT VARIABLES**

927 The ENVIRONMENT VARIABLES section lists what variables affect the utility's
928 execution.

929 The entire manner in which environment variables described in this volume of
930 IEEE Std 1003.1-2001 affect the behavior of each utility is described in the
931 ENVIRONMENT VARIABLES section for that utility, in conjunction with the global
932 XSI effects of the *LANG*, *LC_ALL*, and *NLSPATH* environment variables described in the
933 Base Definitions volume of IEEE Std 1003.1-2001, Chapter 8, Environment Variables.
934 The existence or value of environment variables described in this volume of
935 IEEE Std 1003.1-2001 shall not otherwise affect the specified behavior of the standard
936 utilities. Any effects of the existence or value of environment variables not described by
937 this volume of IEEE Std 1003.1-2001 upon the standard utilities are unspecified.

938 For those standard utilities that use environment variables as a means for selecting a
939 utility to execute (such as *CC* in *make*), the string provided to the utility is subjected to
940 the path search described for *PATH* in the Base Definitions volume of
941 IEEE Std 1003.1-2001, Chapter 8, Environment Variables.

942 All utilities in this volume of IEEE Std 1003.1-2001 shall be capable of processing
943 environment variable names and values using eight-bit transparency.

944 **Default Behavior:** When this section is listed as "None.", it means that the behavior of
945 the utility is not directly affected by environment variables described by this volume of
946 IEEE Std 1003.1-2001 when the utility is used as described by this volume of
947 IEEE Std 1003.1-2001.

948 **ASYNCHRONOUS EVENTS**

949 The ASYNCHRONOUS EVENTS section lists how the utility reacts to such events as
950 signals and what signals are caught.

951 **Default Behavior:** When this section is listed as "Default.", or it refers to "the standard
952 action for all other signals; see Section 1.11 (on page 20)" it means that the action taken
953 as a result of the signal shall be one of the following:

- 954 1. The action shall be that inherited from the parent according to the rules of
955 inheritance of signal actions defined in the System Interfaces volume of
956 IEEE Std 1003.1-2001.
- 957 2. When no action has been taken to change the default, the default action shall be
958 that specified by the System Interfaces volume of IEEE Std 1003.1-2001.
- 959 3. The result of the utility's execution is as if default actions had been taken.

960 A utility is permitted to catch a signal, perform some additional processing (such as
961 deleting temporary files), restore the default signal action (or action inherited from the
962 parent process), and resignal itself.

963 **STDOUT**

964 The STDOUT section completely describes the standard output of the utility. This
965 section is frequently merely a reference to the following section, OUTPUT FILES,
966 because many utilities treat standard output and output files in the same manner.

967 Use of a terminal for standard output may cause any of the standard utilities that write
968 standard output to stop when used in the background. For this reason, applications
969 should not use interactive features in scripts to be placed in the background.

970 Record formats are described in a notation similar to that used by the C-language
971 function, *printf()*. See the Base Definitions volume of IEEE Std 1003.1-2001, Chapter 5,
972 File Format Notation for a description of this notation.

973 The specified standard output of the standard utilities shall not depend on the
974 existence or value of the environment variables defined in this volume of
975 IEEE Std 1003.1-2001, except as provided by this volume of IEEE Std 1003.1-2001.

976 Some of the standard utilities describe their output using the verb *display*, defined in
977 the Base Definitions volume of IEEE Std 1003.1-2001, Section 3.132, Display. Output
978 described in the STDOUT sections of such utilities may be produced using means other
979 than standard output. When standard output is directed to a terminal, the output
980 described shall be written directly to the terminal. Otherwise, the results are undefined.

981 **Default Behavior:** When this section is listed as “Not used.”, it means that the
982 standard output shall not be written when the utility is used as described by this
983 volume of IEEE Std 1003.1-2001.

984 STDERR

985 The STDERR section describes the standard error output of the utility. Only those
986 messages that are purposely sent by the utility are described.

987 Use of a terminal for standard error may cause any of the standard utilities that write
988 standard error output to stop when used in the background. For this reason,
989 applications should not use interactive features in scripts to be placed in the
990 background.

991 The format of diagnostic messages for most utilities is unspecified, but the language
992 and cultural conventions of diagnostic and informative messages whose format is
993 unspecified by this volume of IEEE Std 1003.1-2001 should be affected by the setting of
994 XSI *LC_MESSAGES* and *NLSPATH*.

995 The specified standard error output of standard utilities shall not depend on the
996 existence or value of the environment variables defined in this volume of
997 IEEE Std 1003.1-2001, except as provided by this volume of IEEE Std 1003.1-2001.

998 **Default Behavior:** When this section is listed as “The standard error shall be used only
999 for diagnostic messages.”, it means that, unless otherwise stated, the diagnostic
1000 messages shall be sent to the standard error only when the exit status is non-zero and
1001 the utility is used as described by this volume of IEEE Std 1003.1-2001.

1002 When this section is listed as “Not used.”, it means that the standard error shall not be
1003 used when the utility is used as described in this volume of IEEE Std 1003.1-2001.

1004 OUTPUT FILES

1005 The OUTPUT FILES section completely describes the files created or modified by the
1006 utility. Temporary or system files that are created for internal usage by this utility or
1007 other parts of the implementation (for example, spool, log, and audit files) are not
1008 described in this, or any, section. The utilities creating such files and the names of such
1009 files are unspecified. If applications are written to use temporary or intermediate files,
1010 they should use the *TMPDIR* environment variable, if it is set and represents an
1011 accessible directory, to select the location of temporary files.

1012 Implementations shall ensure that temporary files, when used by the standard utilities,
1013 are named so that different utilities or multiple instances of the same utility can operate
1014 simultaneously without regard to their working directories, or any other process
1015 characteristic other than process ID. There are two exceptions to this rule:

- 1016 1. Resources for temporary files other than the name space (for example, disk space,
1017 available directory entries, or number of processes allowed) are not guaranteed.
1018 2. Certain standard utilities generate output files that are intended as input for other
1019 utilities (for example, *lex* generates *lex.yy.c*), and these cannot have unique
1020 names. These cases are explicitly identified in the descriptions of the respective
1021 utilities.

1022 Any temporary file created by the implementation shall be removed by the
1023 implementation upon a utility's successful exit, exit because of errors, or before
1024 termination by any of the SIGHUP, SIGINT, or SIGTERM signals, unless specified
1025 otherwise by the utility description.

1026 Receipt of the SIGQUIT signal should generally cause termination (unless in some
1027 debugging mode) that would bypass any attempted recovery actions.

1028 Record formats are described in a notation similar to that used by the C-language
1029 function, *printf()*; see the Base Definitions volume of IEEE Std 1003.1-2001, Chapter 5,
1030 File Format Notation for a description of this notation.

1031 **Default Behavior:** When this section is listed as "None.", it means that no files are
1032 created or modified as a consequence of direct action on the part of the utility when the
1033 utility is used as described by this volume of IEEE Std 1003.1-2001. However, the
1034 utility may create or modify system files, such as log files, that are outside the utility's
1035 normal execution environment.

1036 EXTENDED DESCRIPTION

1037 The EXTENDED DESCRIPTION section provides a place for describing the actions of
1038 very complicated utilities, such as text editors or language processors, which typically
1039 have elaborate command languages.

1040 **Default Behavior:** When this section is listed as "None.", no further description is
1041 necessary.

1042 EXIT STATUS

1043 The EXIT STATUS section describes the values the utility shall return to the calling
1044 program, or shell, and the conditions that cause these values to be returned. Usually,
1045 utilities return zero for successful completion and values greater than zero for various
1046 error conditions. If specific numeric values are listed in this section, the system shall
1047 use those values for the errors described. In some cases, status values are listed more
1048 loosely, such as >0. A strictly conforming application shall not rely on any specific
1049 value in the range shown and shall be prepared to receive any value in the range.

1050 For example, a utility may list zero as a successful return, 1 as a failure for a specific
1051 reason, and >1 as "an error occurred". In this case, unspecified conditions may cause a
1052 2 or 3, or other value, to be returned. A conforming application should be written so
1053 that it tests for successful exit status values (zero in this case), rather than relying upon
1054 the single specific error value listed in this volume of IEEE Std 1003.1-2001. In that
1055 way, it has maximum portability, even on implementations with extensions.

1056 Unspecified error conditions may be represented by specific values not listed in this
1057 volume of IEEE Std 1003.1-2001.

1058 CONSEQUENCES OF ERRORS

1059 The CONSEQUENCES OF ERRORS section describes the effects on the environment,
1060 file systems, process state, and so on, when error conditions occur. It does not describe
1061 error messages produced or exit status values used.

1062 The many reasons for failure of a utility are generally not specified by the utility
1063 descriptions. Utilities may terminate prematurely if they encounter: invalid usage of
1064 options, arguments, or environment variables; invalid usage of the complex syntaxes
1065 expressed in EXTENDED DESCRIPTION sections; difficulties accessing, creating,
1066 reading, or writing files; or difficulties associated with the privileges of the process.

1067 The following shall apply to each utility, unless otherwise stated:

- 1068 • If the requested action cannot be performed on an operand representing a file,
1069 directory, user, process, and so on, the utility shall issue a diagnostic message to
1070 standard error and continue processing the next operand in sequence, but the final
1071 exit status shall be returned as non-zero.

1072 For a utility that recursively traverses a file hierarchy (such as *find* or *chown -R*), if
1073 the requested action cannot be performed on a file or directory encountered in the
1074 hierarchy, the utility shall issue a diagnostic message to standard error and continue
1075 processing the remaining files in the hierarchy, but the final exit status shall be
1076 returned as non-zero.

- 1077 • If the requested action characterized by an option or option-argument cannot be
1078 performed, the utility shall issue a diagnostic message to standard error and the exit
1079 status returned shall be non-zero.
- 1080 • When an unrecoverable error condition is encountered, the utility shall exit with a
1081 non-zero exit status.
- 1082 • A diagnostic message shall be written to standard error whenever an error
1083 condition occurs.

1084 When a utility encounters an error condition several actions are possible, depending on
1085 the severity of the error and the state of the utility. Included in the possible actions of
1086 various utilities are: deletion of temporary or intermediate work files; deletion of
1087 incomplete files; validity checking of the file system or directory.

1088 **Default Behavior:** When this section is listed as “Default.”, it means that any changes
1089 to the environment are unspecified.

1090 APPLICATION USAGE

1091 This section is informative.

1092 The APPLICATION USAGE section gives advice to the application programmer or user
1093 about the way the utility should be used.

1094 EXAMPLES

1095 This section is informative.

1096 The EXAMPLES section gives one or more examples of usage, where appropriate. In
1097 the event of conflict between an example and a normative part of the specification, the
1098 normative material is to be taken as correct.

1099 In all examples, quoting has been used, showing how sample commands (utility names
1100 combined with arguments) could be passed correctly to a shell (see *sh*) or as a string to
1101 the *system()* function defined in the System Interfaces volume of IEEE Std 1003.1-2001.
1102 Such quoting would not be used if the utility is invoked using one of the *exec* functions
1103 defined in the System Interfaces volume of IEEE Std 1003.1-2001.

1104 RATIONALE

1105 This section is informative.

1106 This section contains historical information concerning the contents of this volume of
1107 IEEE Std 1003.1-2001 and why features were included or discarded by the standard
1108 developers.

1109 **FUTURE DIRECTIONS**

1110 This section is informative.

1111 The FUTURE DIRECTIONS section should be used as a guide to current thinking; there
1112 is not necessarily a commitment to implement all of these future directions in their
1113 entirety.

1114 **SEE ALSO**

1115 This section is informative.

1116 The SEE ALSO section lists related entries.

1117 **CHANGE HISTORY**

1118 This section is informative.

1119 This section shows the derivation of the entry and any significant changes that have
1120 been made to it.

1121 Certain of the standard utilities describe how they can invoke other utilities or applications, such
1122 as by passing a command string to the command interpreter. The external influences (STDIN,
1123 ENVIRONMENT VARIABLES, and so on) and external effects (STDOUT, CONSEQUENCES OF
1124 ERRORS, and so on) of such invoked utilities are not described in the section concerning the
1125 standard utility that invokes them.

1126 **1.12 Considerations for Utilities in Support of Files of Arbitrary Size**

1127 The following utilities support files of any size up to the maximum that can be created by the
1128 implementation. This support includes correct writing of file size-related values (such as file
1129 sizes and offsets, line numbers, and block counts) and correct interpretation of command line
1130 arguments that contain such values.

- 1131 *basename* Return non-directory portion of pathname.
- 1132 *cat* Concatenate and print files.
- 1133 *cd* Change working directory.
- 1134 *chgrp* Change file group ownership.
- 1135 *chmod* Change file modes.
- 1136 *chown* Change file ownership.
- 1137 *cksum* Write file checksums and sizes.
- 1138 *cmp* Compare two files.
- 1139 *cp* Copy files.
- 1140 *dd* Convert and copy a file.
- 1141 *df* Report free disk space.
- 1142 *dirname* Return directory portion of pathname.
- 1143 *du* Estimate file space usage.

1144	<i>find</i>	Find files.
1145	<i>ln</i>	Link files.
1146	<i>ls</i>	List directory contents.
1147	<i>mkdir</i>	Make directories.
1148	<i>mv</i>	Move files.
1149	<i>pathchk</i>	Check pathnames.
1150	<i>pwd</i>	Return working directory name.
1151	<i>rm</i>	Remove directory entries.
1152	<i>rmdir</i>	Remove directories.
1153	<i>sh</i>	Shell, the standard command language interpreter.
1154	<i>sum</i>	Print checksum and block or byte count of a file.
1155	<i>test</i>	Evaluate expression.
1156	<i>touch</i>	Change file access and modification times.
1157	<i>ulimit</i>	Set or report file size limit.
1158	Exceptions to the requirement that utilities support files of any size up to the maximum are as follows:	
1159		
1160	1.	Uses of files as command scripts, or for configuration or control, are exempt. For example, it is not required that <i>sh</i> be able to read an arbitrarily large .profile.
1161	2.	Shell input and output redirection are exempt. For example, it is not required that the redirections <i>sum < file</i> or <i>echo foo > file</i> succeed for an arbitrarily large existing file.
1162		
1163		

1164 1.13 Built-In Utilities

1165 Any of the standard utilities may be implemented as regular built-in utilities within the
 1166 command language interpreter. This is usually done to increase the performance of frequently
 1167 used utilities or to achieve functionality that would be more difficult in a separate environment.
 1168 The utilities named in Table 1-5 are frequently provided in built-in form. All of the utilities
 1169 named in the table have special properties in terms of command search order within the shell, as
 1170 described in Section 2.9.1.1 (on page 48).

1171 **Table 1-5** Regular Built-In Utilities

1172	<i>alias</i>	<i>false</i>	<i>jobs</i>	<i>read</i>	<i>wait</i>
1173	<i>bg</i>	<i>fc</i>	<i>kill</i>	<i>true</i>	
1174	<i>cd</i>	<i>fg</i>	<i>newgrp</i>	<i>umask</i>	
1175	<i>command</i>	<i>getopts</i>	<i>pwd</i>	<i>unalias</i>	

1176 However, all of the standard utilities, including the regular built-ins in the table, but not the
 1177 special built-ins described in Section 2.14 (on page 64), shall be implemented in a manner so that
 1178 they can be accessed via the *exec* family of functions as defined in the System Interfaces volume
 1179 of IEEE Std 1003.1-2001 and can be invoked directly by those standard utilities that require it
 1180 (*env*, *find*, *nice*, *nohup*, *time*, *xargs*).

Shell Command Language

1182

This chapter contains the definition of the Shell Command Language.

1183

2.1 Shell Introduction

1184
1185
1186

The shell is a command language interpreter. This chapter describes the syntax of that command language as it is used by the *sh* utility and the *system()* and *popen()* functions defined in the System Interfaces volume of IEEE Std 1003.1-2001.

1187
1188

The shell operates according to the following general overview of operations. The specific details are included in the cited sections of this chapter.

1189
1190
1191
1192

1. The shell reads its input from a file (see *sh*), from the *-c* option or from the *system()* and *popen()* functions defined in the System Interfaces volume of IEEE Std 1003.1-2001. If the first line of a file of shell commands starts with the characters "#!", the results are unspecified.

1193
1194
1195

2. The shell breaks the input into tokens: words and operators; see Section 2.3 (on page 31).

1196
1197
1198

3. The shell parses the input into simple commands (see Section 2.9.1 (on page 47)) and compound commands (see Section 2.9.4 (on page 52)).

1199
1200
1201
1202
1203
1204
1205

4. The shell performs various expansions (separately) on different parts of each command, resulting in a list of pathnames and fields to be treated as a command and arguments; see Section 2.6 (on page 36).

1206
1207

5. The shell performs redirection (see Section 2.7 (on page 43)) and removes redirection operators and their operands from the parameter list.

6. The shell executes a function (see Section 2.9.5 (on page 54)), built-in (see Section 2.14 (on page 64)), executable file, or script, giving the names of the arguments as positional parameters numbered 1 to *n*, and the name of the command (or in the case of a function within a script, the name of the script) as the positional parameter numbered 0 (see Section 2.9.1.1 (on page 48)).

7. The shell optionally waits for the command to complete and collects the exit status (see Section 2.8.2 (on page 46)).

1208 **2.2 Quoting**

1209 Quoting is used to remove the special meaning of certain characters or words to the shell.
 1210 Quoting can be used to preserve the literal meaning of the special characters in the next
 1211 paragraph, prevent reserved words from being recognized as such, and prevent parameter
 1212 expansion and command substitution within here-document processing (see Section 2.7.4 (on
 1213 page 44)).

1214 The application shall quote the following characters if they are to represent themselves:

1215 | & ; < > () \$ ` \ " ' <space> <tab> <newline>

1216 and the following may need to be quoted under certain circumstances. That is, these characters
 1217 may be special depending on conditions described elsewhere in this volume of
 1218 IEEE Std 1003.1-2001:

1219 * ? [# ~ = %

1220 The various quoting mechanisms are the escape character, single-quotes, and double-quotes.
 1221 The here-document represents another form of quoting; see Section 2.7.4 (on page 44).

1222 **2.2.1 Escape Character (Backslash)**

1223 A backslash that is not quoted shall preserve the literal value of the following character, with the
 1224 exception of a <newline>. If a <newline> follows the backslash, the shell shall interpret this as
 1225 line continuation. The backslash and <newline>s shall be removed before splitting the input into
 1226 tokens. Since the escaped <newline> is removed entirely from the input and is not replaced by
 1227 any white space, it cannot serve as a token separator.

1228 **2.2.2 Single-Quotes**

1229 Enclosing characters in single-quotes (‘ ’) shall preserve the literal value of each character
 1230 within the single-quotes. A single-quote cannot occur within single-quotes.

1231 **2.2.3 Double-Quotes**

1232 Enclosing characters in double-quotes (" ") shall preserve the literal value of all characters
 1233 within the double-quotes, with the exception of the characters dollar sign, backquote, and
 1234 backslash, as follows:

1235 \$ The dollar sign shall retain its special meaning introducing parameter expansion (see
 1236 Section 2.6.2 (on page 37)), a form of command substitution (see Section 2.6.3 (on page 40)),
 1237 and arithmetic expansion (see Section 2.6.4 (on page 41)).

1238 The input characters within the quoted string that are also enclosed between "\$(" and the
 1239 matching ")" shall not be affected by the double-quotes, but rather shall define that
 1240 command whose output replaces the "\$(...)" when the word is expanded. The
 1241 tokenizing rules in Section 2.3 (on page 31), not including the alias substitutions in Section
 1242 2.3.1 (on page 32), shall be applied recursively to find the matching ') '.

1243 Within the string of characters from an enclosed "\${" to the matching ' } ', an even number
 1244 of unescaped double-quotes or single-quotes, if any, shall occur. A preceding backslash
 1245 character shall be used to escape a literal '{' or '} '. The rule in Section 2.6.2 (on page 37)
 1246 shall be used to determine the matching ' } '.

1247 ' The backquote shall retain its special meaning introducing the other form of command
 1248 substitution (see Section 2.6.3 (on page 40)). The portion of the quoted string from the initial
 1249 backquote and the characters up to the next backquote that is not preceded by a backslash,

1250 having escape characters removed, defines that command whose output replaces ``...''
 1251 when the word is expanded. Either of the following cases produces undefined results:

- 1252 • A single-quoted or double-quoted string that begins, but does not end, within the
 1253 ``...'' sequence
 - 1254 • A ``...'' sequence that begins, but does not end, within the same double-quoted
 1255 string
- 1256 \ The backslash shall retain its special meaning as an escape character (see Section 2.2.1 (on
 1257 page 30)) only when followed by one of the following characters when considered special:

1258 \$ ` " \ <newline>

1259 The application shall ensure that a double-quote is preceded by a backslash to be included
 1260 within double-quotes. The parameter '@' has special meaning inside double-quotes and is
 1261 described in Section 2.5.2 (on page 34).

1262 2.3 Token Recognition

1263 The shell shall read its input in terms of lines from a file, from a terminal in the case of an
 1264 interactive shell, or from a string in the case of *sh -c* or *system()*. The input lines can be of
 1265 unlimited length. These lines shall be parsed using two major modes: ordinary token recognition
 1266 and processing of here-documents.

1267 When an **io_here** token has been recognized by the grammar (see Section 2.10 (on page 55)), one
 1268 or more of the subsequent lines immediately following the next **NEWLINE** token form the body
 1269 of one or more here-documents and shall be parsed according to the rules of Section 2.7.4 (on
 1270 page 44).

1271 When it is not processing an **io_here**, the shell shall break its input into tokens by applying the
 1272 first applicable rule below to the next character in its input. The token shall be from the current
 1273 position in the input until a token is delimited according to one of the rules below; the characters
 1274 forming the token are exactly those in the input, including any quoting characters. If it is
 1275 indicated that a token is delimited, and no characters have been included in a token, processing
 1276 shall continue until an actual token is delimited.

- 1277 1. If the end of input is recognized, the current token shall be delimited. If there is no current
 1278 token, the end-of-input indicator shall be returned as the token.
- 1279 2. If the previous character was used as part of an operator and the current character is not
 1280 quoted and can be used with the current characters to form an operator, it shall be used as
 1281 part of that (operator) token.
- 1282 3. If the previous character was used as part of an operator and the current character cannot
 1283 be used with the current characters to form an operator, the operator containing the
 1284 previous character shall be delimited.
- 1285 4. If the current character is backslash, single-quote, or double-quote ('\'', ''', or '') and
 1286 it is not quoted, it shall affect quoting for subsequent characters up to the end of the quoted
 1287 text. The rules for quoting are as described in Section 2.2 (on page 30). During token
 1288 recognition no substitutions shall be actually performed, and the result token shall contain
 1289 exactly the characters that appear in the input (except for <newline> joining), unmodified,
 1290 including any embedded or enclosing quotes or substitution operators, between the quote
 1291 mark and the end of the quoted text. The token shall not be delimited by the end of the
 1292 quoted field.

- 1293 5. If the current character is an unquoted '\$' or ' ', the shell shall identify the start of any
1294 candidates for parameter expansion (Section 2.6.2 (on page 37)), command substitution
1295 (Section 2.6.3 (on page 40)), or arithmetic expansion (Section 2.6.4 (on page 41)) from their
1296 introductory unquoted character sequences: '\$' or "\${", "\$(" or ' ', and "\$((", respectively. The shell shall read sufficient input to determine the end of the unit to be
1297 expanded (as explained in the cited sections). While processing the characters, if instances
1298 of expansions or quoting are found nested within the substitution, the shell shall
1299 recursively process them in the manner specified for the construct that is found. The
1300 characters found from the beginning of the substitution to its end, allowing for any
1301 recursion necessary to recognize embedded constructs, shall be included unmodified in the
1302 result token, including any embedded or enclosing substitution operators or quotes. The
1303 token shall not be delimited by the end of the substitution.
1304
- 1305 6. If the current character is not quoted and can be used as the first character of a new
1306 operator, the current token (if any) shall be delimited. The current character shall be used
1307 as the beginning of the next (operator) token.
1308 7. If the current character is an unquoted <newline>, the current token shall be delimited.
1309 8. If the current character is an unquoted <blank>, any token containing the previous
1310 character is delimited and the current character shall be discarded.
1311 9. If the previous character was part of a word, the current character shall be appended to
1312 that word.
1313 10. If the current character is a '#', it and all subsequent characters up to, but excluding, the
1314 next <newline> shall be discarded as a comment. The <newline> that ends the line is not
1315 considered part of the comment.
1316 11. The current character is used as the start of a new word.

1317 Once a token is delimited, it is categorized as required by the grammar in Section 2.10 (on page
1318 55).

1319 2.3.1 Alias Substitution

1320 UP XSI The processing of aliases shall be supported on all XSI-conformant systems or if the system
1321 supports the User Portability Utilities option (and the rest of this section is not further shaded for
1322 these options).

1323 After a token has been delimited, but before applying the grammatical rules in Section 2.10 (on
1324 page 55), a resulting word that is identified to be the command name word of a simple
1325 command shall be examined to determine whether it is an unquoted, valid alias name. However,
1326 reserved words in correct grammatical context shall not be candidates for alias substitution. A
1327 valid alias name (see the Base Definitions volume of IEEE Std 1003.1-2001, Section 3.10, Alias
1328 Name) shall be one that has been defined by the *alias* utility and not subsequently undefined
1329 using *unalias*. Implementations also may provide predefined valid aliases that are in effect when
1330 the shell is invoked. To prevent infinite loops in recursive aliasing, if the shell is not currently
1331 processing an alias of the same name, the word shall be replaced by the value of the alias;
1332 otherwise, it shall not be replaced.

1333 If the value of the alias replacing the word ends in a <blank>, the shell shall check the next
1334 command word for alias substitution; this process shall continue until a word is found that is not
1335 a valid alias or an alias value does not end in a <blank>.

1336 When used as specified by this volume of IEEE Std 1003.1-2001, alias definitions shall not be
1337 inherited by separate invocations of the shell or by the utility execution environments invoked
1338 by the shell; see Section 2.12 (on page 61).

1339 **2.4 Reserved Words**

1340 Reserved words are words that have special meaning to the shell; see Section 2.9 (on page 47).
 1341 The following words shall be recognized as reserved words:

1342 !	do	esac	in
1343 {	done	fi	then
1344 }	elif	for	until
1345 case	else	if	while

1346 This recognition shall only occur when none of the characters is quoted and when the word is
 1347 used as:

- 1348 • The first word of a command
- 1349 • The first word following one of the reserved words other than **case**, **for**, or **in**
- 1350 • The third word in a **case** command (only **in** is valid in this case)
- 1351 • The third word in a **for** command (only **in** and **do** are valid in this case)

1352 See the grammar in Section 2.10 (on page 55).

1353 The following words may be recognized as reserved words on some implementations (when
 1354 none of the characters are quoted), causing unspecified results:

1355 [[]]	function	select
--------------	----	----------	--------

1356 Words that are the concatenation of a name and a colon (':') are reserved; their use produces
 1357 unspecified results.

1358 **2.5 Parameters and Variables**

1359 A parameter can be denoted by a name, a number, or one of the special characters listed in
 1360 Section 2.5.2 (on page 34). A variable is a parameter denoted by a name.

1361 A parameter is set if it has an assigned value (null is a valid value). Once a variable is set, it can
 1362 only be unset by using the *unset* special built-in command.

1363 **2.5.1 Positional Parameters**

1364 A positional parameter is a parameter denoted by the decimal value represented by one or more
 1365 digits, other than the single digit 0. The digits denoting the positional parameters shall always be
 1366 interpreted as a decimal value, even if there is a leading zero. When a positional parameter with
 1367 more than one digit is specified, the application shall enclose the digits in braces (see Section
 1368 2.6.2 (on page 37)). Positional parameters are initially assigned when the shell is invoked (see
 1369 *sh*), temporarily replaced when a shell function is invoked (see Section 2.9.5 (on page 54)), and
 1370 can be reassigned with the *set* special built-in command.

1371 **2.5.2 Special Parameters**

1372 Listed below are the special parameters and the values to which they shall expand. Only the
1373 values of the special parameters are listed; see Section 2.6 (on page 36) for a detailed summary of
1374 all the stages involved in expanding words.

- 1375 @ Expands to the positional parameters, starting from one. When the expansion occurs within
1376 double-quotes, and where field splitting (see Section 2.6.5 (on page 42)) is performed, each
1377 positional parameter shall expand as a separate field, with the provision that the expansion
1378 of the first parameter shall still be joined with the beginning part of the original word
1379 (assuming that the expanded parameter was embedded within a word), and the expansion
1380 of the last parameter shall still be joined with the last part of the original word. If there are
1381 no positional parameters, the expansion of '@' shall generate zero fields, even when '@' is
1382 double-quoted.
- 1383 * Expands to the positional parameters, starting from one. When the expansion occurs within
1384 a double-quoted string (see Section 2.2.3 (on page 30)), it shall expand to a single field with
1385 the value of each parameter separated by the first character of the *IFS* variable, or by a
1386 <space> if *IFS* is unset. If *IFS* is set to a null string, this is not equivalent to unsetting it; its
1387 first character does not exist, so the parameter values are concatenated.
- 1388 # Expands to the decimal number of positional parameters. The command name (parameter
1389 0) shall not be counted in the number given by '#' because it is a special parameter, not a
1390 positional parameter.
- 1391 ? Expands to the decimal exit status of the most recent pipeline (see Section 2.9.2 (on page
1392 49)).
- 1393 - (Hyphen.) Expands to the current option flags (the single-letter option names concatenated
1394 into a string) as specified on invocation, by the *set* special built-in command, or implicitly by
1395 the shell.
- 1396 \$ Expands to the decimal process ID of the invoked shell. In a subshell (see Section 2.12 (on
1397 page 61)), '\$' shall expand to the same value as that of the current shell.
- 1398 ! Expands to the decimal process ID of the most recent background command (see Section
1399 2.9.3 (on page 50)) executed from the current shell. (For example, background commands
1400 executed from subshells do not affect the value of "\$!" in the current shell environment.)
1401 For a pipeline, the process ID is that of the last command in the pipeline.
- 1402 0 (Zero.) Expands to the name of the shell or shell script. See *sh* (on page 851) for a detailed
1403 description of how this name is derived.

1404 See the description of the *IFS* variable in Section 2.5.3.

1405 **2.5.3 Shell Variables**

1406 Variables shall be initialized from the environment (as defined by the Base Definitions volume of
1407 IEEE Std 1003.1-2001, Chapter 8, Environment Variables and the *exec* function in the System
1408 Interfaces volume of IEEE Std 1003.1-2001) and can be given new values with variable
1409 assignment commands. If a variable is initialized from the environment, it shall be marked for
1410 export immediately; see the *export* special built-in. New variables can be defined and initialized
1411 with variable assignments, with the *read* or *getopts* utilities, with the *name* parameter in a *for*
1412 loop, with the \${name=word} expansion, or with other mechanisms provided as implementation
1413 extensions.

1414 The following variables shall affect the execution of the shell:

1415	UP XSI	<i>ENV</i>	The processing of the <i>ENV</i> shell variable shall be supported on all XSI-conformant systems or if the system supports the User Portability Utilities option.
1416			
1417			
1418			This variable, when and only when an interactive shell is invoked, shall be subjected to parameter expansion (see Section 2.6.2 (on page 37)) by the shell and the resulting value shall be used as a pathname of a file containing shell commands to execute in the current environment. The file need not be executable. If the expanded value of <i>ENV</i> is not an absolute pathname, the results are unspecified. <i>ENV</i> shall be ignored if the user's real and effective user IDs or real and effective group IDs are different.
1419			
1420			
1421			
1422			
1423			
1424			
1425		<i>HOME</i>	The pathname of the user's home directory. The contents of <i>HOME</i> are used in tilde expansion (see Section 2.6.1 (on page 37)).
1426			
1427		<i>IFS</i>	(Input Field Separators.) A string treated as a list of characters that is used for field splitting and to split lines into fields with the <i>read</i> command. If <i>IFS</i> is not set, the shell shall behave as if the value of <i>IFS</i> is <space>, <tab>, and <newline>; see Section 2.6.5 (on page 42). Implementations may ignore the value of <i>IFS</i> in the environment at the time the shell is invoked, treating <i>IFS</i> as if it were not set.
1428			
1429			
1430			
1431			
1432			
1433		<i>LANG</i>	Provide a default value for the internationalization variables that are unset or null. (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2, Internationalization Variables for the precedence of internationalization variables used to determine the values of locale categories.)
1434			
1435			
1436			
1437		<i>LC_ALL</i>	The value of this variable overrides the <i>LC_*</i> variables and <i>LANG</i> , as described in the Base Definitions volume of IEEE Std 1003.1-2001, Chapter 8, Environment Variables.
1438			
1439			
1440		<i>LC_COLLATE</i>	Determine the behavior of range expressions, equivalence classes, and multi-character collating elements within pattern matching.
1441			
1442		<i>LC_CTYPE</i>	Determine the interpretation of sequences of bytes of text data as characters (for example, single-byte as opposed to multi-byte characters), which characters are defined as letters (character class alpha) and <blank>s (character class blank), and the behavior of character classes within pattern matching. Changing the value of <i>LC_CTYPE</i> after the shell has started shall not affect the lexical processing of shell commands in the current shell execution environment or its subshells. Invoking a shell script or performing <i>exec sh</i> subjects the new shell to the changes in <i>LC_CTYPE</i> .
1443			
1444			
1445			
1446			
1447			
1448			
1449			
1450		<i>LC_MESSAGES</i>	Determine the language in which messages should be written.
1451		<i>LINENO</i>	Set by the shell to a decimal number representing the current sequential line number (numbered starting with 1) within a script or function before it executes each command. If the user unsets or resets <i>LINENO</i> , the variable may lose its special meaning for the life of the shell. If the shell is not currently executing a script or function, the value of <i>LINENO</i> is unspecified. This volume of IEEE Std 1003.1-2001 specifies the effects of the variable only for systems supporting the User Portability Utilities option.
1452			
1453			
1454			
1455			
1456			
1457			
1458	XSI	<i>NLSPATH</i>	Determine the location of message catalogs for the processing of <i>LC_MESSAGES</i> .
1459			
1460		<i>PATH</i>	A string formatted as described in the Base Definitions volume of IEEE Std 1003.1-2001, Chapter 8, Environment Variables, used to effect
1461			

1462		command interpretation; see Section 2.9.1.1 (on page 48).
1463	<i>PPID</i>	Set by the shell to the decimal process ID of the process that invoked this shell. In a subshell (see Section 2.12 (on page 61)), <i>PPID</i> shall be set to the same value as that of the parent of the current shell. For example, <i>echo \$PPID</i> and (<i>echo \$PPID</i>) would produce the same value. This volume of IEEE Std 1003.1-2001 specifies the effects of the variable only for systems supporting the User Portability Utilities option.
1469	<i>PS1</i>	Each time an interactive shell is ready to read a command, the value of this variable shall be subjected to parameter expansion and written to standard error. The default value shall be " \$. For users who have specific additional implementation-defined privileges, the default may be another, implementation-defined value. The shell shall replace each instance of the character ' ! ' in <i>PS1</i> with the history file number of the next command to be typed. Escaping the ' ! ' with another ' ! ' (that is, " ! ! ") shall place the literal character ' ! ' in the prompt. This volume of IEEE Std 1003.1-2001 specifies the effects of the variable only for systems supporting the User Portability Utilities option.
1479	<i>PS2</i>	Each time the user enters a <newline> prior to completing a command line in an interactive shell, the value of this variable shall be subjected to parameter expansion and written to standard error. The default value is "> ". This volume of IEEE Std 1003.1-2001 specifies the effects of the variable only for systems supporting the User Portability Utilities option.
1484	<i>PS4</i>	When an execution trace (<i>set -x</i>) is being performed in an interactive shell, before each line in the execution trace, the value of this variable shall be subjected to parameter expansion and written to standard error. The default value is "+ ". This volume of IEEE Std 1003.1-2001 specifies the effects of the variable only for systems supporting the User Portability Utilities option.
1489	<i>PWD</i>	Set by the shell to be an absolute pathname of the current working directory, containing no components of type symbolic link, no components that are dot, and no components that are dot-dot when the shell is initialized. If an application sets or unsets the value of <i>PWD</i> , the behaviors of the <i>cd</i> and <i>pwd</i> utilities are unspecified.

1494 2.6 Word Expansions

1495 This section describes the various expansions that are performed on words. Not all expansions
 1496 are performed on every word, as explained in the following sections.

1497 Tilde expansions, parameter expansions, command substitutions, arithmetic expansions, and
 1498 quote removals that occur within a single word expand to a single field. It is only field splitting
 1499 or pathname expansion that can create multiple fields from a single word. The single exception
 1500 to this rule is the expansion of the special parameter '@' within double-quotes, as described in
 1501 Section 2.5.2 (on page 34).

1502 The order of word expansion shall be as follows:

- 1503 1. Tilde expansion (see Section 2.6.1 (on page 37)), parameter expansion (see Section 2.6.2 (on
 1504 page 37)), command substitution (see Section 2.6.3 (on page 40)), and arithmetic expansion
 1505 (see Section 2.6.4 (on page 41)) shall be performed, beginning to end. See item 5 in Section
 1506 2.3 (on page 31).

- 1507 2. Field splitting (see Section 2.6.5 (on page 42)) shall be performed on the portions of the
 1508 fields generated by step 1, unless *IFS* is null.
- 1509 3. Pathname expansion (see Section 2.6.6 (on page 42)) shall be performed, unless *set -f* is in
 1510 effect.
- 1511 4. Quote removal (see Section 2.6.7 (on page 42)) shall always be performed last.

1512 The expansions described in this section shall occur in the same shell environment as that in
 1513 which the command is executed.

1514 If the complete expansion appropriate for a word results in an empty field, that empty field shall
 1515 be deleted from the list of fields that form the completely expanded command, unless the
 1516 original word contained single-quote or double-quote characters.

1517 The '\$' character is used to introduce parameter expansion, command substitution, or
 1518 arithmetic evaluation. If an unquoted '\$' is followed by a character that is either not numeric,
 1519 the name of one of the special parameters (see Section 2.5.2 (on page 34)), a valid first character
 1520 of a variable name, a left curly brace ('{') or a left parenthesis, the result is unspecified.

1521 2.6.1 Tilde Expansion

1522 A "tilde-prefix" consists of an unquoted tilde character at the beginning of a word, followed by
 1523 all of the characters preceding the first unquoted slash in the word, or all the characters in the
 1524 word if there is no slash. In an assignment (see the Base Definitions volume of
 1525 IEEE Std 1003.1-2001, Section 4.21, Variable Assignment), multiple tilde-prefixes can be used: at
 1526 the beginning of the word (that is, following the equal sign of the assignment), following any
 1527 unquoted colon, or both. A tilde-prefix in an assignment is terminated by the first unquoted
 1528 colon or slash. If none of the characters in the tilde-prefix are quoted, the characters in the tilde-
 1529 prefix following the tilde are treated as a possible login name from the user database. A portable
 1530 login name cannot contain characters outside the set given in the description of the *LOGNAME*
 1531 environment variable in the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.3, Other
 1532 Environment Variables. If the login name is null (that is, the tilde-prefix contains only the tilde),
 1533 the tilde-prefix is replaced by the value of the variable *HOME*. If *HOME* is unset, the results are
 1534 unspecified. Otherwise, the tilde-prefix shall be replaced by a pathname of the initial working
 1535 directory associated with the login name obtained using the *getpwnam()* function as defined in
 1536 the System Interfaces volume of IEEE Std 1003.1-2001. If the system does not recognize the login
 1537 name, the results are undefined.

1538 2.6.2 Parameter Expansion

1539 The format for parameter expansion is as follows:

1540 `${expression}`

1541 where *expression* consists of all characters until the matching '}'. Any '}' escaped by a
 1542 backslash or within a quoted string, and characters in embedded arithmetic expansions,
 1543 command substitutions, and variable expansions, shall not be examined in determining the
 1544 matching '}'.

1545 The simplest form for parameter expansion is:

1546 `${parameter}`

1547 The value, if any, of *parameter* shall be substituted.

1548 The parameter name or symbol can be enclosed in braces, which are optional except for
 1549 positional parameters with more than one digit or when *parameter* is followed by a character that
 1550 could be interpreted as part of the name. The matching closing brace shall be determined by

1551	counting brace levels, skipping over enclosed quoted strings, and command substitutions.
1552	If the parameter name or symbol is not enclosed in braces, the expansion shall use the longest valid name (see the Base Definitions volume of IEEE Std 1003.1-2001, Section 3.230, Name), whether or not the symbol represented by that name exists.
1553	
1554	
1555	If a parameter expansion occurs inside double-quotes:
1556	• Pathname expansion shall not be performed on the results of the expansion.
1557	• Field splitting shall not be performed on the results of the expansion, with the exception of
1558	'@'; see Section 2.5.2 (on page 34).
1559	In addition, a parameter expansion can be modified by using one of the following formats. In
1560	each case that a value of <i>word</i> is needed (based on the state of <i>parameter</i> , as described below),
1561	<i>word</i> shall be subjected to tilde expansion, parameter expansion, command substitution, and
1562	arithmetic expansion. If <i>word</i> is not needed, it shall not be expanded. The '}' character that
1563	delimits the following parameter expansion modifications shall be determined as described
1564	previously in this section and in Section 2.2.3 (on page 30). (For example, \${foo-bar}xyz} would
1565	result in the expansion of foo followed by the string xyz} if foo is set, else the string
1566	"barxyz}").
1567	<code>\${parameter:-word}</code> Use Default Values. If <i>parameter</i> is unset or null, the expansion of <i>word</i>
1568	shall be substituted; otherwise, the value of <i>parameter</i> shall be substituted.
1569	<code>\${parameter:=word}</code> Assign Default Values. If <i>parameter</i> is unset or null, the expansion of
1570	<i>word</i> shall be assigned to <i>parameter</i> . In all cases, the final value of
1571	<i>parameter</i> shall be substituted. Only variables, not positional parameters
1572	or special parameters, can be assigned in this way.
1573	<code>\${parameter?:[word]}</code> Indicate Error if Null or Unset. If <i>parameter</i> is unset or null, the
1574	expansion of <i>word</i> (or a message indicating it is unset if <i>word</i> is omitted) shall be written to standard error and the shell exits with a non-zero exit
1575	status. Otherwise, the value of <i>parameter</i> shall be substituted. An
1576	interactive shell need not exit.
1577	
1578	<code>\${parameter:+word}</code> Use Alternative Value. If <i>parameter</i> is unset or null, null shall be
1579	substituted; otherwise, the expansion of <i>word</i> shall be substituted.
1580	In the parameter expansions shown previously, use of the colon in the format shall result in a
1581	test for a parameter that is unset or null; omission of the colon shall result in a test for a
1582	parameter that is only unset. The following table summarizes the effect of the colon:
1583	
1584	
1585	
1586	
1587	
1588	
1589	
1590	
1591	
1592	

	<i>parameter</i> Set and Not Null	<i>parameter</i> Set But Null	<i>parameter</i> Unset
<code>\${parameter:-word}</code>	substitute <i>parameter</i>	substitute <i>word</i>	substitute <i>word</i>
<code>\${parameter=word}</code>	substitute <i>parameter</i>	substitute <i>null</i>	substitute <i>word</i>
<code>\${parameter:=word}</code>	substitute <i>parameter</i>	assign <i>word</i>	assign <i>word</i>
<code>\${parameter=word}</code>	substitute <i>parameter</i>	substitute <i>null</i>	assign <i>word</i>
<code>\${parameter?:word}</code>	substitute <i>parameter</i>	error, exit	error, exit
<code>\${parameter?word}</code>	substitute <i>parameter</i>	substitute <i>null</i>	error, exit
<code>\${parameter:+word}</code>	substitute <i>word</i>	substitute <i>null</i>	substitute <i>null</i>
<code>\${parameter+word}</code>	substitute <i>word</i>	substitute <i>word</i>	substitute <i>null</i>

In all cases shown with “substitute”, the expression is replaced with the value shown. In all cases shown with “assign”, *parameter* is assigned that value, which also replaces the expression.

1595	<code>\${#parameter}</code>	String Length. The length in characters of the value of <i>parameter</i> shall be substituted. If <i>parameter</i> is '*' or '@', the result of the expansion is unspecified.
1598		The following four varieties of parameter expansion provide for substring processing. In each case, pattern matching notation (see Section 2.13 (on page 62)), rather than regular expression notation, shall be used to evaluate the patterns. If <i>parameter</i> is '*' or '@', the result of the expansion is unspecified. Enclosing the full parameter expansion string in double-quotes shall not cause the following four varieties of pattern characters to be quoted, whereas quoting characters within the braces shall have this effect.
1604	<code>\${parameter%word}</code>	Remove Smallest Suffix Pattern. The <i>word</i> shall be expanded to produce a pattern. The parameter expansion shall then result in <i>parameter</i> , with the smallest portion of the suffix matched by the <i>pattern</i> deleted.
1607	<code>\${parameter%%word}</code>	Remove Largest Suffix Pattern. The <i>word</i> shall be expanded to produce a pattern. The parameter expansion shall then result in <i>parameter</i> , with the largest portion of the suffix matched by the <i>pattern</i> deleted.
1610	<code>\${parameter#word}</code>	Remove Smallest Prefix Pattern. The <i>word</i> shall be expanded to produce a pattern. The parameter expansion shall then result in <i>parameter</i> , with the smallest portion of the prefix matched by the <i>pattern</i> deleted.
1613	<code>\${parameter##word}</code>	Remove Largest Prefix Pattern. The <i>word</i> shall be expanded to produce a pattern. The parameter expansion shall then result in <i>parameter</i> , with the largest portion of the prefix matched by the <i>pattern</i> deleted.

1616 Examples

1617 `${parameter:-word}`

1618 In this example, *ls* is executed only if *x* is null or unset. (The `$!(ls)` command substitution
1619 notation is explained in Section 2.6.3 (on page 40).)

1620 `$ {x:-$(ls)}`

1621 `${parameter:=word}`

1622 `unset X`
1623 `echo ${X:=abc}`
1624 `abc`

1625 `${parameter?:word}`

1626 `unset posix`
1627 `echo ${posix?:?}`
1628 `sh: posix: parameter null or not set`

1629 `${parameter:+word}`

1630 `set a b c`
1631 `echo ${3:+posix}`
1632 `posix`

1633 `$#{parameter}`

1634 `HOME=/usr posix`
1635 `echo ${#HOME}`
1636 `10`

1637 `${parameter%word}`

1638 `x=file.c`
1639 `echo ${x%.c}.o`

```

1640      file.o
1641      ${parameter%%word}
1642          x=posix/src/std
1643          echo ${x%/*}
1644          posix
1645      ${parameter#word}
1646          x=$HOME/src/cmd
1647          echo ${x#$HOME}
1648          /src/cmd
1649      ${parameter##word}
1650          x=/one/two/three
1651          echo ${x##*/}
1652          three
1653 The double-quoting of patterns is different depending on where the double-quotes are placed:
1654 " ${x##*}"    The asterisk is a pattern character.
1655 ${x#*"}       The literal asterisk is quoted and not special.

```

1656 2.6.3 Command Substitution

1657 Command substitution allows the output of a command to be substituted in place of the
 1658 command name itself. Command substitution shall occur when the command is enclosed as
 1659 follows:

1660 \$(*command*)

1661 or (backquoted version):

1662 `*command*`

1663 The shell shall expand the command substitution by executing *command* in a subshell
 1664 environment (see Section 2.12 (on page 61)) and replacing the command substitution (the text of
 1665 *command* plus the enclosing "\$()" or backquotes) with the standard output of the command,
 1666 removing sequences of one or more <newline>s at the end of the substitution. Embedded
 1667 <newline>s before the end of the output shall not be removed; however, they may be treated as
 1668 field delimiters and eliminated during field splitting, depending on the value of IFS and quoting
 1669 that is in effect.

1670 Within the backquoted style of command substitution, backslash shall retain its literal meaning,
 1671 except when followed by: '\$', ``', or '\` (dollar sign, backquote, backslash). The search for
 1672 the matching backquote shall be satisfied by the first backquote found without a preceding
 1673 backslash; during this search, if a non-escaped backquote is encountered within a shell
 1674 comment, a here-document, an embedded command substitution of the \$(*command*) form, or a
 1675 quoted string, undefined results occur. A single-quoted or double-quoted string that begins, but
 1676 does not end, within the ``...`` sequence produces undefined results.

1677 With the \$(*command*) form, all characters following the open parenthesis to the matching closing
 1678 parenthesis constitute the *command*. Any valid shell script can be used for *command*, except a
 1679 script consisting solely of redirections which produces unspecified results.

1680 The results of command substitution shall not be processed for further tilde expansion,
 1681 parameter expansion, command substitution, or arithmetic expansion. If a command
 1682 substitution occurs inside double-quotes, field splitting and pathname expansion shall not be
 1683 performed on the results of the substitution.

1
1

1684 Command substitution can be nested. To specify nesting within the backquoted version, the
 1685 application shall precede the inner backquotes with backslashes, for example:

1686 `\`command\``

1687 If the command substitution consists of a single subshell, such as:

1688 `$((command))`

1689 a conforming application shall separate the "`$()`" and '`()`' into two tokens (that is, separate
 1690 them with white space). This is required to avoid any ambiguities with arithmetic expansion.

1691 2.6.4 Arithmetic Expansion

1692 Arithmetic expansion provides a mechanism for evaluating an arithmetic expression and
 1693 substituting its value. The format for arithmetic expansion shall be as follows:

1694 `$((expression))`

1695 The expression shall be treated as if it were in double-quotes, except that a double-quote inside
 1696 the expression is not treated specially. The shell shall expand all tokens in the expression for
 1697 parameter expansion, command substitution, and quote removal.

1698 Next, the shell shall treat this as an arithmetic expression and substitute the value of the
 1699 expression. The arithmetic expression shall be processed according to the rules given in Section
 1700 1.7.2.1 (on page 7), with the following exceptions:

- 1701 • Only signed long integer arithmetic is required.
- 1702 • Only the decimal-constant, octal-constant, and hexadecimal-constant constants specified in
 the ISO C standard, Section 6.4.4.1 are required to be recognized as constants.
- 1704 • The `sizeof()` operator and the prefix and postfix "`++`" and "`--`" operators are not required.
- 1705 • Selection, iteration, and jump statements are not supported.

1706 All changes to variables in an arithmetic expression shall be in effect after the arithmetic 2
 1707 expansion, as in the parameter expansion "`$(x=value)`". 2

1708 If the shell variable `x` contains a value that forms a valid integer constant, then the arithmetic 2
 1709 expansions "`$(x)`" and "`$(($x))`" shall return the same value. 2

1710 As an extension, the shell may recognize arithmetic expressions beyond those listed. The shell
 1711 may use a signed integer type with a rank larger than the rank of **signed long**. The shell may use
 1712 a real-floating type instead of **signed long** as long as it does not affect the results in cases where
 1713 there is no overflow. If the expression is invalid, the expansion fails and the shell shall write a
 1714 message to standard error indicating the failure.

1715 Examples

1716 A simple example using arithmetic expansion:

```
1717       # repeat a command 100 times
1718       x=100
1719       while [ $x -gt 0 ]
1720       do
1721             command
1722             x=$((x-1))
1723       done
```

1724 2.6.5 Field Splitting

1725 After parameter expansion (Section 2.6.2 (on page 37)), command substitution (Section 2.6.3 (on
1726 page 40)), and arithmetic expansion (Section 2.6.4 (on page 41)), the shell shall scan the results of
1727 expansions and substitutions that did not occur in double-quotes for field splitting and multiple
1728 fields can result.

1729 The shell shall treat each character of the *IFS* as a delimiter and use the delimiters to split the
1730 results of parameter expansion and command substitution into fields.

- 1731 1. If the value of *IFS* is a <space>, <tab>, and <newline>, or if it is unset, any sequence of
1732 <space>s, <tab>s, or <newline>s at the beginning or end of the input shall be ignored and
1733 any sequence of those characters within the input shall delimit a field. For example, the
1734 input:

1735 <newline><space><tab>foo<tab><tab>bar<space>

1736 yields two fields, **foo** and **bar**.

- 1737 2. If the value of *IFS* is null, no field splitting shall be performed.
- 1738 3. Otherwise, the following rules shall be applied in sequence. The term “*IFS* white space” is
1739 used to mean any sequence (zero or more instances) of white space characters that are in
1740 the *IFS* value (for example, if *IFS* contains <space>/<comma>/<tab>, any sequence of
1741 <space>s and <tab>s is considered *IFS* white space).
- 1742 a. *IFS* white space shall be ignored at the beginning and end of the input.
- 1743 b. Each occurrence in the input of an *IFS* character that is not *IFS* white space, along
1744 with any adjacent *IFS* white space, shall delimit a field, as described previously.
- 1745 c. Non-zero-length *IFS* white space shall delimit a field.

1746 2.6.6 Pathname Expansion

1747 After field splitting, if *set -f* is not in effect, each field in the resulting command line shall be
1748 expanded using the algorithm described in Section 2.13 (on page 62), qualified by the rules in
1749 Section 2.13.3 (on page 63).

1750 2.6.7 Quote Removal

1751 The quote characters: ‘\’, ‘ ’, and ‘ ”’ (backslash, single-quote, double-quote) that were
1752 present in the original word shall be removed unless they have themselves been quoted.

1753 **2.7 Redirection**

1754 Redirection is used to open and close files for the current shell execution environment (see
1755 Section 2.12 (on page 61)) or for any command. Redirection operators can be used with numbers
1756 representing file descriptors (see the Base Definitions volume of IEEE Std 1003.1-2001, Section
1757 3.165, File Descriptor) as described below.

1758 The overall format used for redirection is:

1759 `[n]redir-op word`

1760 The number *n* is an optional decimal number designating the file descriptor number; the
1761 application shall ensure it is delimited from any preceding text and immediately precede the
1762 redirection operator *redir-op*. If *n* is quoted, the number shall not be recognized as part of the
1763 redirection expression. For example:

1764 `echo \2>a`

1765 writes the character 2 into file a. If any part of *redir-op* is quoted, no redirection expression is
1766 recognized. For example:

1767 `echo 2\>a`

1768 writes the characters 2>a to standard output. The optional number, redirection operator, and
1769 word shall not appear in the arguments provided to the command to be executed (if any).

1770 Open files are represented by decimal numbers starting with zero. The largest possible value is
1771 implementation-defined; however, all implementations shall support at least 0 to 9, inclusive, for
1772 use by the application. These numbers are called “file descriptors”. The values 0, 1, and 2 have
1773 special meaning and conventional uses and are implied by certain redirection operations; they
1774 are referred to as *standard input*, *standard output*, and *standard error*, respectively. Programs
1775 usually take their input from standard input, and write output on standard output. Error
1776 messages are usually written on standard error. The redirection operators can be preceded by
1777 one or more digits (with no intervening <blank>s allowed) to designate the file descriptor
1778 number.

1779 If the redirection operator is "<<" or "<<-", the word that follows the redirection operator shall
1780 be subjected to quote removal; it is unspecified whether any of the other expansions occur. For
1781 the other redirection operators, the word that follows the redirection operator shall be subjected
1782 to tilde expansion, parameter expansion, command substitution, arithmetic expansion, and
1783 quote removal. Pathname expansion shall not be performed on the word by a non-interactive
1784 shell; an interactive shell may perform it, but shall do so only when the expansion would result
1785 in one word.

1786 If more than one redirection operator is specified with a command, the order of evaluation is
1787 from beginning to end.

1788 A failure to open or create a file shall cause a redirection to fail.

1789 **2.7.1 Redirecting Input**

1790 Input redirection shall cause the file whose name results from the expansion of *word* to be
 1791 opened for reading on the designated file descriptor, or standard input if the file descriptor is not
 1792 specified.

1793 The general format for redirecting input is:

1794 `[n]<word`

1795 where the optional *n* represents the file descriptor number. If the number is omitted, the
 1796 redirection shall refer to standard input (file descriptor 0).

1797 **2.7.2 Redirecting Output**

1798 The two general formats for redirecting output are:

1799 `[n]>word`

1800 `[n]>|word`

1801 where the optional *n* represents the file descriptor number. If the number is omitted, the
 1802 redirection shall refer to standard output (file descriptor 1).

1803 Output redirection using the '`>`' format shall fail if the *noclobber* option is set (see the
 1804 description of *set -C*) and the file named by the expansion of *word* exists and is a regular file.
 1805 Otherwise, redirection using the '`>`' or "`>|`" formats shall cause the file whose name results
 1806 from the expansion of *word* to be created and opened for output on the designated file
 1807 descriptor, or standard output if none is specified. If the file does not exist, it shall be created;
 1808 otherwise, it shall be truncated to be an empty file after being opened.

1809 **2.7.3 Appending Redirected Output**

1810 Appended output redirection shall cause the file whose name results from the expansion of
 1811 *word* to be opened for output on the designated file descriptor. The file is opened as if the *open()*
 1812 function as defined in the System Interfaces volume of IEEE Std 1003.1-2001 was called with the
 1813 *O_APPEND* flag. If the file does not exist, it shall be created.

1814 The general format for appending redirected output is as follows:

1815 `[n]>>word`

1816 where the optional *n* represents the file descriptor number. If the number is omitted, the
 1817 redirection refers to standard output (file descriptor 1).

1818 **2.7.4 Here-Document**

1819 The redirection operators "`<<`" and "`<<-`" both allow redirection of lines contained in a shell
 1820 input file, known as a "here-document", to the input of a command.

1821 The here-document shall be treated as a single word that begins after the next *<newline>* and
 1822 continues until there is a line containing only the delimiter and a *<newline>*, with no *<blank>*s in
 1823 between. Then the next here-document starts, if there is one. The format is as follows:

1824 `[n]<<word`
 1825 *here-document*
 1826 *delimiter*

1827 where the optional *n* represents the file descriptor number. If the number is omitted, the here-
 1828 document refers to standard input (file descriptor 0).

1829 If any character in *word* is quoted, the delimiter shall be formed by performing quote removal on
1830 *word*, and the here-document lines shall not be expanded. Otherwise, the delimiter shall be the
1831 *word* itself.

1832 If no characters in *word* are quoted, all lines of the here-document shall be expanded for
1833 parameter expansion, command substitution, and arithmetic expansion. In this case, the
1834 backslash in the input behaves as the backslash inside double-quotes (see Section 2.2.3 (on page
1835 30)). However, the double-quote character ('"') shall not be treated specially within a here-
1836 document, except when the double-quote appears within "\$()", "''", or "\${}".

1837 If the redirection symbol is "<<-", all leading <tab>s shall be stripped from input lines and the
1838 line containing the trailing delimiter. If more than one "<<" or "<<-" operator is specified on a
1839 line, the here-document associated with the first operator shall be supplied first by the
1840 application and shall be read first by the shell.

1841 Examples

1842 An example of a here-document follows:

```
1843 cat <<eof1; cat <<eof2
1844     Hi,
1845     eof1
1846     Helene.
1847     eof2
```

1848 2.7.5 Duplicating an Input File Descriptor

1849 The redirection operator:

```
1850 [n]<&word
```

1851 shall duplicate one input file descriptor from another, or shall close one. If *word* evaluates to one
1852 or more digits, the file descriptor denoted by *n*, or standard input if *n* is not specified, shall be
1853 made to be a copy of the file descriptor denoted by *word*; if the digits in *word* do not represent a
1854 file descriptor already open for input, a redirection error shall result; see Section 2.8.1 (on page
1855 46). If *word* evaluates to '-', file descriptor *n*, or standard input if *n* is not specified, shall be
1856 closed. Attempts to close a file descriptor that is not open shall not constitute an error. If *word*
1857 evaluates to something else, the behavior is unspecified.

1858 2.7.6 Duplicating an Output File Descriptor

1859 The redirection operator:

```
1860 [n]>&word
```

1861 shall duplicate one output file descriptor from another, or shall close one. If *word* evaluates to
1862 one or more digits, the file descriptor denoted by *n*, or standard output if *n* is not specified, shall
1863 be made to be a copy of the file descriptor denoted by *word*; if the digits in *word* do not represent
1864 a file descriptor already open for output, a redirection error shall result; see Section 2.8.1 (on page
1865 46). If *word* evaluates to '-', file descriptor *n*, or standard output if *n* is not specified, is
1866 closed. Attempts to close a file descriptor that is not open shall not constitute an error. If *word*
1867 evaluates to something else, the behavior is unspecified.

1868 **2.7.7 Open File Descriptors for Reading and Writing**

1869 The redirection operator:

1870 `[n]<>word`

1871 shall cause the file whose name is the expansion of *word* to be opened for both reading and
 1872 writing on the file descriptor denoted by *n*, or standard input if *n* is not specified. If the file does
 1873 not exist, it shall be created.

1874 **2.8 Exit Status and Errors**

1875 **2.8.1 Consequences of Shell Errors**

1876 For a non-interactive shell, an error condition encountered by a special built-in (see Section 2.14
 1877 (on page 64)) or other type of utility shall cause the shell to write a diagnostic message to
 1878 standard error and exit as shown in the following table:

Error	Special Built-In	Other Utilities
Shell language syntax error	Shall exit	Shall exit
Utility syntax error (option or operand error)	Shall exit	Shall not exit
Redirection error	Shall exit	Shall not exit
Variable assignment error	Shall exit	Shall not exit
Expansion error	Shall exit	Shall exit
Command not found	N/A	May exit
Dot script not found	Shall exit	N/A

1887 An expansion error is one that occurs when the shell expansions defined in Section 2.6 (on page
 1888 36) are carried out (for example, "`$(x!y)`", because '`!`' is not a valid operator); an
 1889 implementation may treat these as syntax errors if it is able to detect them during tokenization,
 1890 rather than during expansion.

1891 If any of the errors shown as "shall exit" or "(may) exit" occur in a subshell, the subshell shall
 1892 (respectively may) exit with a non-zero status, but the script containing the subshell shall not
 1893 exit because of the error.

1894 In all of the cases shown in the table, an interactive shell shall write a diagnostic message to
 1895 standard error without exiting.

1896 **2.8.2 Exit Status for Commands**

1897 Each command has an exit status that can influence the behavior of other shell commands. The
 1898 exit status of commands that are not utilities is documented in this section. The exit status of the
 1899 standard utilities is documented in their respective sections.

1900 If a command is not found, the exit status shall be 127. If the command name is found, but it is
 1901 not an executable utility, the exit status shall be 126. Applications that invoke utilities without
 1902 using the shell should use these exit status values to report similar errors.

1903 If a command fails during word expansion or redirection, its exit status shall be greater than
 1904 zero.

1905 Internally, for purposes of deciding whether a command exits with a non-zero exit status, the
 1906 shell shall recognize the entire status value retrieved for the command by the equivalent of the
 1907 `wait()` function `WEXITSTATUS` macro (as defined in the System Interfaces volume of
 1908 IEEE Std 1003.1-2001). When reporting the exit status with the special parameter '?', the shell

1909 shall report the full eight bits of exit status available. The exit status of a command that
1910 terminated because it received a signal shall be reported as greater than 128.

1911 2.9 Shell Commands

1912 This section describes the basic structure of shell commands. The following command
1913 descriptions each describe a format of the command that is only used to aid the reader in
1914 recognizing the command type, and does not formally represent the syntax. Each description
1915 discusses the semantics of the command; for a formal definition of the command language,
1916 consult Section 2.10 (on page 55).

1917 A *command* is one of the following:

- 1918 • Simple command (see Section 2.9.1)
- 1919 • Pipeline (see Section 2.9.2 (on page 49))
- 1920 • List compound-list (see Section 2.9.3 (on page 50))
- 1921 • Compound command (see Section 2.9.4 (on page 52))
- 1922 • Function definition (see Section 2.9.5 (on page 54))

1923 Unless otherwise stated, the exit status of a command shall be that of the last simple command
1924 executed by the command. There shall be no limit on the size of any shell command other than
1925 that imposed by the underlying system (memory constraints, {ARG_MAX}, and so on).

1926 2.9.1 Simple Commands

1927 A “simple command” is a sequence of optional variable assignments and redirections, in any
1928 sequence, optionally followed by words and redirections, terminated by a control operator.

1929 When a given simple command is required to be executed (that is, when any conditional
1930 construct such as an AND-OR list or a *case* statement has not bypassed the simple command),
1931 the following expansions, assignments, and redirections shall all be performed from the
1932 beginning of the command text to the end:

- 1933 1. The words that are recognized as variable assignments or redirections according to Section
1934 2.10.2 (on page 56) are saved for processing in steps 3 and 4.
- 1935 2. The words that are not variable assignments or redirections shall be expanded. If any fields
1936 remain following their expansion, the first field shall be considered the command name
1937 and remaining fields are the arguments for the command.
- 1938 3. Redirections shall be performed as described in Section 2.7 (on page 43).
- 1939 4. Each variable assignment shall be expanded for tilde expansion, parameter expansion,
1940 command substitution, arithmetic expansion, and quote removal prior to assigning the
1941 value.

1942 In the preceding list, the order of steps 3 and 4 may be reversed for the processing of special
1943 built-in utilities; see Section 2.14 (on page 64).

1944 If no command name results, variable assignments shall affect the current execution
1945 environment. Otherwise, the variable assignments shall be exported for the execution
1946 environment of the command and shall not affect the current execution environment (except for
1947 special built-ins). If any of the variable assignments attempt to assign a value to a read-only
1948 variable, a variable assignment error shall occur. See Section 2.8.1 (on page 46) for the
1949 consequences of these errors.

If there is no command name, any redirections shall be performed in a subshell environment; it is unspecified whether this subshell environment is the same one as that used for a command substitution within the command. (To affect the current execution environment, see the `exec` special built-in.) If any of the redirections performed in the current shell execution environment fail, the command shall immediately fail with an exit status greater than zero, and the shell shall write an error message indicating the failure. See Section 2.8.1 (on page 46) for the consequences of these failures on interactive and non-interactive shells.

If there is a command name, execution shall continue as described in Section 2.9.1.1. If there is no command name, but the command contained a command substitution, the command shall complete with the exit status of the last command substitution performed. Otherwise, the command shall complete with a zero exit status.

2.9.1.1 Command Search and Execution

If a simple command results in a command name and an optional list of arguments, the following actions shall be performed:

1. If the command name does not contain any slashes, the first successful step in the following sequence shall occur:
 - a. If the command name matches the name of a special built-in utility, that special built-in utility shall be invoked.
 - b. If the command name matches the name of a function known to this shell, the function shall be invoked as described in Section 2.9.5 (on page 54). If the implementation has provided a standard utility in the form of a function, it shall not be recognized at this point. It shall be invoked in conjunction with the path search in step 1d.
 - c. If the command name matches the name of a utility listed in the following table, that utility shall be invoked.

<code>alias</code>	<code>false</code>	<code>jobs</code>	<code>read</code>	<code>wait</code>
<code>bg</code>	<code>fc</code>	<code>kill</code>	<code>true</code>	
<code>cd</code>	<code>fg</code>	<code>newgrp</code>	<code>umask</code>	
<code>command</code>	<code>getopts</code>	<code>pwd</code>	<code>unalias</code>	

- d. Otherwise, the command shall be searched for using the `PATH` environment variable as described in the Base Definitions volume of IEEE Std 1003.1-2001, Chapter 8, Environment Variables:

- i. If the search is successful:

- a. If the system has implemented the utility as a regular built-in or as a shell function, it shall be invoked at this point in the path search.
- b. Otherwise, the shell executes the utility in a separate utility environment (see Section 2.12 (on page 61)) with actions equivalent to calling the `execve()` function as defined in the System Interfaces volume of IEEE Std 1003.1-2001 with the `path` argument set to the pathname resulting from the search, `arg0` set to the command name, and the remaining arguments set to the operands, if any.

If the `execve()` function fails due to an error equivalent to the [ENOEXEC] error defined in the System Interfaces volume of IEEE Std 1003.1-2001, the shell shall execute a command equivalent to having a shell invoked with the pathname resulting from the search as its first operand, with any

remaining arguments passed to the new shell, except that the value of "\$0" in the new shell may be set to the command name. If the executable file is not a text file, the shell may bypass this command execution. In this case, it shall write an error message, and shall return an exit status of 126.

Once a utility has been searched for and found (either as a result of this specific search or as part of an unspecified shell start-up activity), an implementation may remember its location and need not search for the utility again unless the *PATH* variable has been the subject of an assignment. If the remembered location fails for a subsequent invocation, the shell shall repeat the search to find the new location for the utility, if any.

- 2005 ii. If the search is unsuccessful, the command shall fail with an exit status of 127
2006 and the shell shall write an error message.

2007 2. If the command name contains at least one slash, the shell shall execute the utility in a
2008 separate utility environment with actions equivalent to calling the *execve()* function
2009 defined in the System Interfaces volume of IEEE Std 1003.1-2001 with the *path* and *arg0*
2010 arguments set to the command name, and the remaining arguments set to the operands, if
2011 any.

If the `execve()` function fails due to an error equivalent to the [ENOEXEC] error, the shell shall execute a command equivalent to having a shell invoked with the command name as its first operand, with any remaining arguments passed to the new shell. If the executable file is not a text file, the shell may bypass this command execution. In this case, it shall write an error message and shall return an exit status of 126.

2017 2.9.2 Pipelines

2018 A *pipeline* is a sequence of one or more commands separated by the control operator ‘|’. The standard output of all but the last command shall be connected to the standard input of the next command.

2021 The format for a pipeline is:

2022 [!] *command1* [| *command2* ...]

The standard output of *command1* shall be connected to the standard input of *command2*. The standard input, standard output, or both of a command shall be considered to be assigned by the pipeline before any redirection specified by redirection operators that are part of the command (see Section 2.7 (on page 43)).

If the pipeline is not in the background (see Section 2.9.3.1 (on page 50)), the shell shall wait for the last command specified in the pipeline to complete, and may also wait for all commands to complete.

2030 Exit Status

2031 If the reserved word ! does not precede the pipeline, the exit status shall be the exit status of the
2032 last command specified in the pipeline. Otherwise, the exit status shall be the logical NOT of the
2033 exit status of the last command. That is, if the last command returns zero, the exit status shall be
2034 1; if the last command returns greater than zero, the exit status shall be zero.

2035 **2.9.3 Lists**

2036 An *AND-OR list* is a sequence of one or more pipelines separated by the operators "`&&`" and
 2037 "`||`".

2038 A *list* is a sequence of one or more AND-OR lists separated by the operators '`;`' and '`&`' and
 2039 optionally terminated by '`;`', '`&`', or `<newline>`.

2040 The operators "`&&`" and "`||`" shall have equal precedence and shall be evaluated with left
 2041 associativity. For example, both of the following commands write solely **bar** to standard output:

```
2042     false && echo foo || echo bar
  2043     true || echo foo && echo bar
```

2044 A '`:`' or `<newline>` terminator shall cause the preceding AND-OR list to be executed
 2045 sequentially; an '`&`' shall cause asynchronous execution of the preceding AND-OR list.

2046 The term "compound-list" is derived from the grammar in Section 2.10 (on page 55); it is
 2047 equivalent to a sequence of *lists*, separated by `<newline>`s, that can be preceded or followed by
 2048 an arbitrary number of `<newline>`s.

2049 **Examples**

2050 The following is an example that illustrates `<newline>`s in compound-lists:

```
2051     while
  2052         # a couple of <newline>s
  2053
  2054         # a list
  2055         date && who || ls; cat file
  2056         # a couple of <newline>s
  2057
  2058         # another list
  2059         wc file > output & true
  2060
  2061         do
  2062             # 2 lists
  2063             ls
  2064             cat file
  2065         done
```

2063 **2.9.3.1 Asynchronous Lists**

2064 If a command is terminated by the control operator ampersand ('`&`'), the shell shall execute the
 2065 command asynchronously in a subshell. This means that the shell shall not wait for the
 2066 command to finish before executing the next command.

2067 The format for running a command in the background is:

```
2068     command1 & [command2 & ... ]
```

2069 The standard input for an asynchronous list, before any explicit redirections are performed, shall
 2070 be considered to be assigned to a file that has the same properties as `/dev/null`. If it is an
 2071 interactive shell, this need not happen. In all cases, explicit redirection of standard input shall
 2072 override this activity.

2073 When an element of an asynchronous list (the portion of the list ended by an ampersand, such as
 2074 `command1`, above) is started by the shell, the process ID of the last command in the asynchronous
 2075 list element shall become known in the current shell execution environment; see Section 2.12 (on
 2076 page 61). This process ID shall remain known until:

- 2077 1. The command terminates and the application waits for the process ID.
2078 2. Another asynchronous list invoked before "\$!" (corresponding to the previous
2079 asynchronous list) is expanded in the current execution environment.

2080 The implementation need not retain more than the {CHILD_MAX} most recent entries in its list
2081 of known process IDs in the current shell execution environment.

2082 **Exit Status**

2083 The exit status of an asynchronous list shall be zero.

2084 **2.9.3.2 Sequential Lists**

2085 Commands that are separated by a semicolon (';') shall be executed sequentially.

2086 The format for executing commands sequentially shall be:

2087 *command1* [; *command2*] ...

2088 Each command shall be expanded and executed in the order specified.

2089 **Exit Status**

2090 The exit status of a sequential list shall be the exit status of the last command in the list.

2091 **2.9.3.3 AND Lists**

2092 The control operator "&&" denotes an AND list. The format shall be:

2093 *command1* [&& *command2*] ...

2094 First *command1* shall be executed. If its exit status is zero, *command2* shall be executed, and so on,
2095 until a command has a non-zero exit status or there are no more commands left to execute. The
2096 commands are expanded only if they are executed.

2097 **Exit Status**

2098 The exit status of an AND list shall be the exit status of the last command that is executed in the
2099 list.

2100 **2.9.3.4 OR Lists**

2101 The control operator "||" denotes an OR List. The format shall be:

2102 *command1* [|| *command2*] ...

2103 First, *command1* shall be executed. If its exit status is non-zero, *command2* shall be executed, and
2104 so on, until a command has a zero exit status or there are no more commands left to execute.

2105 **Exit Status**

2106 The exit status of an OR list shall be the exit status of the last command that is executed in the
2107 list.

2108 **2.9.4 Compound Commands**

2109 The shell has several programming constructs that are “compound commands”, which provide
 2110 control flow for commands. Each of these compound commands has a reserved word or control
 2111 operator at the beginning, and a corresponding terminator reserved word or operator at the end.
 2112 In addition, each can be followed by redirections on the same line as the terminator. Each
 2113 redirection shall apply to all the commands within the compound command that do not
 2114 explicitly override that redirection.

2115 **2.9.4.1 Grouping Commands**

2116 The format for grouping commands is as follows:

- 2117 (compound-list) Execute *compound-list* in a subshell environment; see Section 2.12 (on page
 2118 61). Variable assignments and built-in commands that affect the
 2119 environment shall not remain in effect after the list finishes.
- 2120 { compound-list;} Execute *compound-list* in the current process environment. The semicolon
 2121 shown here is an example of a control operator delimiting the } reserved
 2122 word. Other delimiters are possible, as shown in Section 2.10 (on page
 2123 55); a <newline> is frequently used.

2124 **Exit Status**

2125 The exit status of a grouping command shall be the exit status of *compound-list*.

2126 **2.9.4.2 The for Loop**

2127 The **for** loop shall execute a sequence of commands for each member in a list of *items*. The **for**
 2128 loop requires that the reserved words **do** and **done** be used to delimit the sequence of
 2129 commands.

2130 The format for the **for** loop is as follows:

```
2131     for name [ in [word ... ] ]
2132         do
2133             compound-list
2134         done
```

2135 First, the list of words following **in** shall be expanded to generate a list of items. Then, the
 2136 variable *name* shall be set to each item, in turn, and the *compound-list* executed each time. If no
 2137 items result from the expansion, the *compound-list* shall not be executed. Omitting:

2138 in word ...

2139 shall be equivalent to:

2140 in "\$@"

2141 **Exit Status**

2142 The exit status of a **for** command shall be the exit status of the last command that executes. If
 2143 there are no items, the exit status shall be zero.

2144 2.9.4.3 Case Conditional Construct

2145 The conditional construct **case** shall execute the *compound-list* corresponding to the first one of
 2146 several *patterns* (see Section 2.13 (on page 62)) that is matched by the string resulting from the
 2147 tilde expansion, parameter expansion, command substitution, arithmetic expansion, and quote
 2148 removal of the given word. The reserved word **in** shall denote the beginning of the patterns to be
 2149 matched. Multiple patterns with the same *compound-list* shall be delimited by the '|'
 2150 symbol. The control operator ')' terminates a list of patterns corresponding to a given action. The
 2151 *compound-list* for each list of patterns, with the possible exception of the last, shall be terminated
 2152 with "; ;". The **case** construct terminates with the reserved word **esac** (**case** reversed).

2153 The format for the **case** construct is as follows:

```
2154 case word in
2155   [( ]pattern1) compound-list;;
2156   [[( ]pattern[ | pattern] ...) compound-list;;] ...
2157   [[[( ]pattern[ | pattern] ...) compound-list]
2158   esac
```

2159 The " ; ; " is optional for the last *compound-list*.

2160 In order from the beginning to the end of the **case** statement, each *pattern* that labels a
 2161 *compound-list* shall be subjected to tilde expansion, parameter expansion, command substitution,
 2162 and arithmetic expansion, and the result of these expansions shall be compared against the
 2163 expansion of *word*, according to the rules described in Section 2.13 (on page 62) (which also
 2164 describes the effect of quoting parts of the pattern). After the first match, no more patterns shall
 2165 be expanded, and the *compound-list* shall be executed. The order of expansion and comparison of
 2166 multiple *patterns* that label a *compound-list* statement is unspecified.

2167 **Exit Status**

2168 The exit status of **case** shall be zero if no patterns are matched. Otherwise, the exit status shall be
 2169 the exit status of the last command executed in the *compound-list*.

2170 2.9.4.4 The if Conditional Construct

2171 The **if** command shall execute a *compound-list* and use its exit status to determine whether to
 2172 execute another *compound-list*.

2173 The format for the **if** construct is as follows:

```
2174 if compound-list
2175   then
2176     compound-list
2177   [elif compound-list
2178   then
2179     compound-list] ...
2180   [else
2181     compound-list]
2182   fi
```

2183 The **if** *compound-list* shall be executed; if its exit status is zero, the **then** *compound-list* shall be
 2184 executed and the command shall complete. Otherwise, each **elif** *compound-list* shall be executed,
 2185 in turn, and if its exit status is zero, the **then** *compound-list* shall be executed and the command
 2186 shall complete. Otherwise, the **else** *compound-list* shall be executed.

2187 **Exit Status**

2188 The exit status of the **if** command shall be the exit status of the **then** or **else compound-list** that
2189 was executed, or zero, if none was executed.

2190 **2.9.4.5 The while Loop**

2191 The **while** loop shall continuously execute one **compound-list** as long as another **compound-list** has
2192 a zero exit status.

2193 The format of the **while** loop is as follows:

```
2194           while compound-list-1
2195            do
2196              compound-list-2
2197            done
```

2198 The **compound-list-1** shall be executed, and if it has a non-zero exit status, the **while** command
2199 shall complete. Otherwise, the **compound-list-2** shall be executed, and the process shall repeat.

2200 **Exit Status**

2201 The exit status of the **while** loop shall be the exit status of the last **compound-list-2** executed, or
2202 zero if none was executed.

2203 **2.9.4.6 The until Loop**

2204 The **until** loop shall continuously execute one **compound-list** as long as another **compound-list** has
2205 a non-zero exit status.

2206 The format of the **until** loop is as follows:

```
2207           until compound-list-1
2208            do
2209              compound-list-2
2210            done
```

2211 The **compound-list-1** shall be executed, and if it has a zero exit status, the **until** command
2212 completes. Otherwise, the **compound-list-2** shall be executed, and the process repeats.

2213 **Exit Status**

2214 The exit status of the **until** loop shall be the exit status of the last **compound-list-2** executed, or
2215 zero if none was executed.

2216 **2.9.5 Function Definition Command**

2217 A function is a user-defined name that is used as a simple command to call a compound
2218 command with new positional parameters. A function is defined with a “function definition
2219 command”.

2220 The format of a function definition command is as follows:

```
2221           fname( ) compound-command[io-redirect ... ]
```

2222 The function is named *fname*; the application shall ensure that it is a name (see the Base
2223 Definitions volume of IEEE Std 1003.1-2001, Section 3.230, Name). An implementation may
2224 allow other characters in a function name as an extension. The implementation shall maintain
2225 separate name spaces for functions and variables.

2226 The argument *compound-command* represents a compound command, as described in Section
2227 2.9.4 (on page 52).

2228 When the function is declared, none of the expansions in Section 2.6 (on page 36) shall be
2229 performed on the text in *compound-command* or *io-redirect*; all expansions shall be performed as
2230 normal each time the function is called. Similarly, the optional *io-redirect* redirections and any
2231 variable assignments within *compound-command* shall be performed during the execution of the
2232 function itself, not the function definition. See Section 2.8.1 (on page 46) for the consequences of
2233 failures of these operations on interactive and non-interactive shells.

2234 When a function is executed, it shall have the syntax-error and variable-assignment properties
2235 described for special built-in utilities in the enumerated list at the beginning of Section 2.14 (on
2236 page 64).

2237 The *compound-command* shall be executed whenever the function name is specified as the name
2238 of a simple command (see Section 2.9.1.1 (on page 48)). The operands to the command
2239 temporarily shall become the positional parameters during the execution of the *compound-
2240 command*; the special parameter '#' also shall be changed to reflect the number of operands. The
2241 special parameter 0 shall be unchanged. When the function completes, the values of the
2242 positional parameters and the special parameter '#' shall be restored to the values they had
2243 before the function was executed. If the special built-in *return* is executed in the *compound-
2244 command*, the function completes and execution shall resume with the next command after the
2245 function call.

2246 **Exit Status**

2247 The exit status of a function definition shall be zero if the function was declared successfully;
2248 otherwise, it shall be greater than zero. The exit status of a function invocation shall be the exit
2249 status of the last command executed by the function.

2250 **2.10 Shell Grammar**

2251 The following grammar defines the Shell Command Language. This formal syntax shall take
2252 precedence over the preceding text syntax description.

2253 **2.10.1 Shell Grammar Lexical Conventions**

2254 The input language to the shell must be first recognized at the character level. The resulting
2255 tokens shall be classified by their immediate context according to the following rules (applied in
2256 order). These rules shall be used to determine what a "token" is that is subject to parsing at the
2257 token level. The rules for token recognition in Section 2.3 (on page 31) shall apply.

- 2258 1. A <newline> shall be returned as the token identifier **NEWLINE**.
- 2259 2. If the token is an operator, the token identifier for that operator shall result.
- 2260 3. If the string consists solely of digits and the delimiter character is one of '<' or '>', the
2261 token identifier **IO_NUMBER** shall be returned.
- 2262 4. Otherwise, the token identifier **TOKEN** results.

2263 Further distinction on **TOKEN** is context-dependent. It may be that the same **TOKEN** yields
2264 **WORD**, a **NAME**, an **ASSIGNMENT**, or one of the reserved words below, dependent upon the
2265 context. Some of the productions in the grammar below are annotated with a rule number from
2266 the following list. When a **TOKEN** is seen where one of those annotated productions could be
2267 used to reduce the symbol, the applicable rule shall be applied to convert the token identifier

2268 type of the **TOKEN** to a token identifier acceptable at that point in the grammar. The reduction
 2269 shall then proceed based upon the token identifier type yielded by the rule applied. When more
 2270 than one rule applies, the highest numbered rule shall apply (which in turn may refer to another
 2271 rule). (Note that except in rule 7, the presence of an '=' in the token has no effect.)

2272 The **WORD** tokens shall have the word expansion rules applied to them immediately before the
 2273 associated command is executed, not at the time the command is parsed.

2274 2.10.2 Shell Grammar Rules

2275 1. [Command Name]

2276 When the **TOKEN** is exactly a reserved word, the token identifier for that reserved word
 2277 shall result. Otherwise, the token **WORD** shall be returned. Also, if the parser is in any
 2278 state where only a reserved word could be the next correct token, proceed as above.

2279 **Note:** Because at this point quote marks are retained in the token, quoted strings cannot be
 2280 recognized as reserved words. This rule also implies that reserved words are not
 2281 recognized except in certain positions in the input, such as after a <newline> or
 2282 semicolon; the grammar presumes that if the reserved word is intended, it is properly
 2283 delimited by the user, and does not attempt to reflect that requirement directly. Also
 2284 note that line joining is done before tokenization, as described in Section 2.2.1 (on page
 2285 30), so escaped <newline>s are already removed at this point.

2286 Rule 1 is not directly referenced in the grammar, but is referred to by other rules, or applies
 2287 globally.

2288 2. [Redirection to or from filename]

2289 The expansions specified in Section 2.7 (on page 43) shall occur. As specified there, exactly
 2290 one field can result (or the result is unspecified), and there are additional requirements on
 2291 pathname expansion.

2292 3. [Redirection from here-document]

2293 Quote removal shall be applied to the word to determine the delimiter that is used to find
 2294 the end of the here-document that begins after the next <newline>.

2295 4. [Case statement termination]

2296 When the **TOKEN** is exactly the reserved word **esac**, the token identifier for **esac** shall
 2297 result. Otherwise, the token **WORD** shall be returned.

2298 5. [NAME in for]

2299 When the **TOKEN** meets the requirements for a name (see the Base Definitions volume of
 2300 IEEE Std 1003.1-2001, Section 3.230, Name), the token identifier **NAME** shall result.
 2301 Otherwise, the token **WORD** shall be returned.

2302 6. [Third word of for and case]

2303 a. [case only]

2304 When the **TOKEN** is exactly the reserved word **in**, the token identifier for **in** shall
 2305 result. Otherwise, the token **WORD** shall be returned.

2306 b. [for only]

2307 When the **TOKEN** is exactly the reserved word **in** or **do**, the token identifier for **in** or
 2308 **do** shall result, respectively. Otherwise, the token **WORD** shall be returned.

(For a. and b.: As indicated in the grammar, a *linebreak* precedes the tokens **in** and **do**. If
<newline>s are present at the indicated location, it is the token after them that is treated in
this fashion.)

2313 a. [When the first word]

If the TOKEN does not contain the character '=' , rule 1 is applied. Otherwise, 7b shall be applied.

2316 b. [Not the first word]

If the **TOKEN** contains the equal sign character:

— If it begins with '=' , the token **WORD** shall be returned.

— If all the characters preceding '=' form a valid name (see the Base Definitions volume of IEEE Std 1003.1-2001, Section 3.230, Name), the token **ASSIGNMENT_WORD** shall be returned. (Quoted characters cannot participate in forming a valid name.)

— Otherwise, it is unspecified whether it is **ASSIGNMENT_WORD** or **WORD** that is returned.

Assignment to the NAME shall occur as specified in Section 2.9.1 (on page 47).

2326 8. [NAME in function]

When the **TOKEN** is exactly a reserved word, the token identifier for that reserved word shall result. Otherwise, when the **TOKEN** meets the requirements for a name, the token identifier **NAME** shall result. Otherwise, rule 7 applies.

2330 9. [Body of function]

Word expansion and assignment shall never occur, even when required by the rules above, when this rule is being parsed. Each **TOKEN** that might either be expanded or have assignment applied to it shall instead be returned as a single **WORD** consisting only of characters that are exactly the token described in Section 2.3 (on page 31).

```

2335     /* -----
2336         The grammar symbols
2337         -----
2338     %token WORD
2339     %token ASSIGNMENT_WORD
2340     %token NAME
2341     %token NEWLINE
2342     %token IO_NUMBER

2343     /* The following are the operators mentioned above. */

2344     %token AND_IF      OR_IF      DSEMI
2345     /*          '&&'      '||'      ';' ;'      */
2346     %token DLESS      DGREAT     LESSAND    GREATAND   LESSGREAT  DLESSDASE
2347     /*          '<<'      '>'      '<&'      '>&'      '<>'      '<<-'
2348     %token CLOBBER
2349     /*          '>|'      */

```

```

2350      /* The following are the reserved words. */
2351      %token If Then Else Elif Fi Do Done
2352      /* 'if' 'then' 'else' 'elif' 'fi' 'do' 'done' */
2353      %token Case Esac While Until For
2354      /* 'case' 'esac' 'while' 'until' 'for' */
2355      /* These are reserved words, not operator tokens, and are
2356      recognized when reserved words are recognized. */
2357      %token Lbrace Rbrace Bang
2358      /* '{' '}' '!' */
2359      %token In
2360      /* 'in' */
2361      /* -----
2362      The Grammar
2363      ----- */
2364      %start complete_command
2365      %%
2366      complete_command : list separator
2367          | list
2368          ;
2369      list : list separator_op and_or
2370          | and_or
2371          ;
2372      and_or : pipeline
2373          | and_or AND_IF linebreak pipeline
2374          | and_or OR_IF linebreak pipeline
2375          ;
2376      pipeline : pipe_sequence
2377          | Bang pipe_sequence
2378          ;
2379      pipe_sequence : command
2380          | pipe_sequence '|'
2381          | linebreak command
2382      command : simple_command
2383          | compound_command
2384          | compound_command redirect_list
2385          | function_definition
2386          ;
2387      compound_command : brace_group
2388          | subshell
2389          | for_clause
2390          | case_clause
2391          | if_clause
2392          | while_clause
2393          | until_clause
2394          ;
2395      subshell : '(' compound_list ')'
2396          ;
2397      compound_list : term
2398          | newline_list term

```

```

2399          |           term separator
2400          |   newline_list term separator
2401          ;
2402      term       : term separator and_or
2403          |           and_or
2404          ;
2405      for_clause  : For name linebreak           do_group
2406          | For name linebreak in      sequential_sep do_group
2407          | For name linebreak in wordlist sequential_sep do_group
2408          ;
2409      name        : NAME                      /* Apply rule 5 */
2410          ;
2411      in          : In                       /* Apply rule 6 */
2412          ;
2413      wordlist    : wordlist WORD
2414          |           WORD
2415          ;
2416      case_clause  : Case WORD linebreak in linebreak case_list     Esac
2417          | Case WORD linebreak in linebreak case_list_ns Esac
2418          | Case WORD linebreak in linebreak             Esac
2419          ;
2420      case_list_ns : case_list case_item_ns
2421          |           case_item_ns
2422          ;
2423      case_list   : case_list case_item
2424          |           case_item
2425          ;
2426      case_item_ns : pattern ')'           linebreak
2427          | pattern ')' compound_list linebreak
2428          | '(' pattern ')'           linebreak
2429          | '(' pattern ')' compound_list linebreak
2430          ;
2431      case_item   : pattern ')' linebreak DSEMI linebreak
2432          | pattern ')' compound_list DSEMI linebreak
2433          | '(' pattern ')' linebreak DSEMI linebreak
2434          | '(' pattern ')' compound_list DSEMI linebreak
2435          ;
2436      pattern     : WORD                      /* Apply rule 4 */
2437          | pattern '|' WORD                  /* Do not apply rule 4 */
2438          ;
2439      if_clause   : If compound_list Then compound_list else_part Fi
2440          | If compound_list Then compound_list           Fi
2441          ;
2442      else_part   : Elif compound_list Then else_part
2443          | Else compound_list
2444          ;
2445      while_clause : While compound_list do_group
2446          ;
2447      until_clause : Until compound_list do_group
2448          ;
2449      function_definition : fname '(' ')' linebreak function_body
2450          ;

```

```

2451     function_body      : compound_command          /* Apply rule 9 */
2452             | compound_command redirect_list /* Apply rule 9 */
2453             ;
2454     fname            : NAME                  /* Apply rule 8 */
2455             ;
2456     brace_group       : Lbrace compound_list Rbrace
2457             ;
2458     do_group          : Do compound_list Done      /* Apply rule 6 */
2459             ;
2460     simple_command    : cmd_prefix cmd_word cmd_suffix
2461             | cmd_prefix cmd_word
2462             | cmd_prefix
2463             | cmd_name cmd_suffix
2464             | cmd_name
2465             ;
2466     cmd_name          : WORD                  /* Apply rule 7a */
2467             ;
2468     cmd_word          : WORD                  /* Apply rule 7b */
2469             ;
2470     cmd_prefix         : io_redirect
2471             | cmd_prefix io_redirect
2472             | ASSIGNMENT_WORD
2473             | cmd_prefix ASSIGNMENT_WORD
2474             ;
2475     cmd_suffix         : io_redirect
2476             | cmd_suffix io_redirect
2477             | WORD
2478             | cmd_suffix WORD
2479             ;
2480     redirect_list      : io_redirect
2481             | redirect_list io_redirect
2482             ;
2483     io_redirect        : io_file
2484             | IO_NUMBER io_file
2485             | io_here
2486             | IO_NUMBER io_here
2487             ;
2488     io_file            : '<'      filename
2489             | LESSAND   filename
2490             | '>'      filename
2491             | GREATAND  filename
2492             | DGREAT    filename
2493             | LESSGREAT filename
2494             | CLOBBER   filename
2495             ;
2496     filename           : WORD                  /* Apply rule 2 */
2497             ;
2498     io_here            : DLESS      here_end
2499             | DLESSDASH here_end
2500             ;
2501     here_end           : WORD                  /* Apply rule 3 */
2502             ;

```

```

2503     newline_list      :           NEWLINE
2504             | newline_list NEWLINE
2505             ;
2506     linebreak        : newline_list
2507             | /* empty */
2508             ;
2509     separator_op     : '&'
2510             | ';'
2511             ;
2512     separator         : separator_op linebreak
2513             | newline_list
2514             ;
2515     sequential_sep   : ';' linebreak
2516             | newline_list
2517             ;

```

2518 2.11 Signals and Error Handling

When a command is in an asynchronous list, the shell shall prevent SIGQUIT and SIGINT signals from the keyboard from interrupting the command. Otherwise, signals shall have the values inherited by the shell from its parent (see also the *trap* special built-in).

When a signal for which a trap has been set is received while the shell is waiting for the completion of a utility executing a foreground command, the trap associated with that signal shall not be executed until after the foreground command has completed. When the shell is waiting, by means of the *wait* utility, for asynchronous commands to complete, the reception of a signal for which a trap has been set shall cause the *wait* utility to return immediately with an exit status >128, immediately after which the trap associated with that signal shall be taken.

If multiple signals are pending for the shell for which there are associated trap actions, the order of execution of trap actions is unspecified.

2530 2.12 Shell Execution Environment

A shell execution environment consists of the following:

- Open files inherited upon invocation of the shell, plus open files controlled by *exec*
- Working directory as set by *cd*
- File creation mask set by *umask*
- Current traps set by *trap*
- Shell parameters that are set by variable assignment (see the *set* special built-in) or from the System Interfaces volume of IEEE Std 1003.1-2001 environment inherited by the shell when it begins (see the *export* special built-in)
- Shell functions; see Section 2.9.5 (on page 54)
- Options turned on at invocation or by *set*
- Process IDs of the last commands in asynchronous lists known to this shell environment; see Section 2.9.3.1 (on page 50)

- 2543 • Shell aliases; see Section 2.3.1 (on page 32)

2544 Utilities other than the special built-ins (see Section 2.14 (on page 64)) shall be invoked in a
2545 separate environment that consists of the following. The initial value of these objects shall be the
2546 same as that for the parent shell, except as noted below.

- 2547 • Open files inherited on invocation of the shell, open files controlled by the *exec* special built-
2548 in plus any modifications, and additions specified by any redirections to the utility
- 2549 • Current working directory
- 2550 • File creation mask
- 2551 • If the utility is a shell script, traps caught by the shell shall be set to the default values and
2552 traps ignored by the shell shall be set to be ignored by the utility; if the utility is not a shell
2553 script, the trap actions (default or ignore) shall be mapped into the appropriate signal
2554 handling actions for the utility
- 2555 • Variables with the *export* attribute, along with those explicitly exported for the duration of the
2556 command, shall be passed to the utility environment variables

2557 The environment of the shell process shall not be changed by the utility unless explicitly
2558 specified by the utility description (for example, *cd* and *umask*).

2559 A subshell environment shall be created as a duplicate of the shell environment, except that
2560 signal traps set by that shell environment shall be set to the default values. Changes made to the
2561 subshell environment shall not affect the shell environment. Command substitution, commands
2562 that are grouped with parentheses, and asynchronous lists shall be executed in a subshell
2563 environment. Additionally, each command of a multi-command pipeline is in a subshell
2564 environment; as an extension, however, any or all commands in a pipeline may be executed in
2565 the current environment. All other commands shall be executed in the current shell
2566 environment.

2567 **2.13 Pattern Matching Notation**

2568 The pattern matching notation described in this section is used to specify patterns for matching
2569 strings in the shell. Historically, pattern matching notation is related to, but slightly different
2570 from, the regular expression notation described in the Base Definitions volume of
2571 IEEE Std 1003.1-2001, Chapter 9, Regular Expressions. For this reason, the description of the
2572 rules for this pattern matching notation are based on the description of regular expression
2573 notation, modified to include backslash escape processing.

2574 **2.13.1 Patterns Matching a Single Character**

2575 The following patterns matching a single character shall match a single character: ordinary
2576 characters, special pattern characters, and pattern bracket expressions. The pattern bracket
2577 expression also shall match a single collating element. A backslash character shall escape the
2578 following character. The escaping backslash shall be discarded.

2579 An ordinary character is a pattern that shall match itself. It can be any character in the supported
2580 character set except for NUL, those special shell characters in Section 2.2 (on page 30) that
2581 require quoting, and the following three special pattern characters. Matching shall be based on
2582 the bit pattern used for encoding the character, not on the graphic representation of the
2583 character. If any character (ordinary, shell special, or pattern special) is quoted, that pattern shall
2584 match the character itself. The shell special characters always require quoting.

When unquoted and outside a bracket expression, the following three characters shall have special meaning in the specification of patterns:

- ? A question-mark is a pattern that shall match any character.
- * An asterisk is a pattern that shall match multiple characters, as described in Section 2.13.2.
- [The open bracket shall introduce a pattern bracket expression.

The description of basic regular expression bracket expressions in the Base Definitions volume of IEEE Std 1003.1-2001, Section 9.3.5, RE Bracket Expression shall also apply to the pattern bracket expression, except that the exclamation mark character ('!') shall replace the circumflex character ('^') in its role in a "non-matching list" in the regular expression notation. A bracket expression starting with an unquoted circumflex character produces unspecified results.

When pattern matching is used where shell quote removal is not performed (such as in the argument to the *find -name* primary when *find* is being called using one of the *exec* functions as defined in the System Interfaces volume of IEEE Std 1003.1-2001, or in the *pattern* argument to the *fnmatch()* function), special characters can be escaped to remove their special meaning by preceding them with a backslash character. This escaping backslash is discarded. The sequence "\\" represents one literal backslash. All of the requirements and effects of quoting on ordinary, shell special, and special pattern characters shall apply to escaping in this context.

2.13.2 Patterns Matching Multiple Characters

The following rules are used to construct patterns matching multiple characters from patterns matching a single character:

1. The asterisk (' * ') is a pattern that shall match any string, including the null string.
2. The concatenation of patterns matching a single character is a valid pattern that shall match the concatenation of the single characters or collating elements matched by each of the concatenated patterns.
3. The concatenation of one or more patterns matching a single character with one or more asterisks is a valid pattern. In such patterns, each asterisk shall match a string of zero or more characters, matching the greatest possible number of characters that still allows the remainder of the pattern to match the string.

2.13.3 Patterns Used for Filename Expansion

The rules described so far in Section 2.13.1 (on page 62) and Section 2.13.2 are qualified by the following rules that apply when pattern matching notation is used for filename expansion:

1. The slash character in a pathname shall be explicitly matched by using one or more slashes in the pattern; it shall neither be matched by the asterisk or question-mark special characters nor by a bracket expression. Slashes in the pattern shall be identified before bracket expressions; thus, a slash cannot be included in a pattern bracket expression used for filename expansion. If a slash character is found following an unescaped open square bracket character before a corresponding closing square bracket is found, the open bracket shall be treated as an ordinary character. For example, the pattern "a[b/c]d" does not match such pathnames as **abd** or **a/d**. It only matches a pathname of literally **a[b/c]d**.
2. If a filename begins with a period (' . '), the period shall be explicitly matched by using a period as the first character of the pattern or immediately following a slash character. The leading period shall not be matched by:

- 2628 • The asterisk or question-mark special characters
- 2629 • A bracket expression containing a non-matching list, such as "[!a]", a range
2630 expression, such as "[%–0]", or a character class expression, such as "[[:punct:]]"
- 2631 It is unspecified whether an explicit period in a bracket expression matching list, such as
2632 "[.abc]", can match a leading period in a filename.
- 2633 3. Specified patterns shall be matched against existing filenames and pathnames, as
2634 appropriate. Each component that contains a pattern character shall require read
2635 permission in the directory containing that component. Any component, except the last,
2636 that does not contain a pattern character shall require search permission. For example,
2637 given the pattern:
- 2638

```
/foo/bar/x*/bam
```
- 2639 search permission is needed for directories / and **foo**, search and read permissions are
2640 needed for directory **bar**, and search permission is needed for each **x*** directory. If the
2641 pattern matches any existing filenames or pathnames, the pattern shall be replaced with
2642 those filenames and pathnames, sorted according to the collating sequence in effect in the
2643 current locale. If the pattern contains an invalid bracket expression or does not match any
2644 existing filenames or pathnames, the pattern string shall be left unchanged.

2645 2.14 Special Built-In Utilities

2646 The following "special built-in" utilities shall be supported in the shell command language. The
2647 output of each command, if any, shall be written to standard output, subject to the normal
2648 redirection and piping possible with all commands.

2649 The term "built-in" implies that the shell can execute the utility directly and does not need to
2650 search for it. An implementation may choose to make any utility a built-in; however, the special
2651 built-in utilities described here differ from regular built-in utilities in two respects:

- 2652 1. A syntax error in a special built-in utility may cause a shell executing that utility to abort,
2653 while a syntax error in a regular built-in utility shall not cause a shell executing that utility
2654 to abort. (See Section 2.8.1 (on page 46) for the consequences of errors on interactive and
2655 non-interactive shells.) If a special built-in utility encountering a syntax error does not
2656 abort the shell, its exit value shall be non-zero.
- 2657 2. Variable assignments specified with special built-in utilities remain in effect after the
2658 built-in completes; this shall not be the case with a regular built-in or other utility.

2659 The special built-in utilities in this section need not be provided in a manner accessible via the
2660 exec family of functions defined in the System Interfaces volume of IEEE Std 1003.1-2001.

2661 Some of the special built-ins are described as conforming to the Base Definitions volume of
2662 IEEE Std 1003.1-2001, Section 12.2, Utility Syntax Guidelines. For those that are not, the
2663 requirement in Section 1.11 (on page 20) that "--" be recognized as a first argument to be
2664 discarded does not apply and a conforming application shall not use that argument.

2665 NAME

2666 **break** — exit from for, while, or until loop

2667 SYNOPSIS

2668 **break** [*n*]

2669 DESCRIPTION

2670 The *break* utility shall exit from the smallest enclosing **for**, **while**, or **until** loop, if any; or from the
2671 *n*th enclosing loop if *n* is specified. The value of *n* is an unsigned decimal integer greater than or
2672 equal to 1. The default shall be equivalent to *n*=1. If *n* is greater than the number of enclosing
2673 loops, the outermost enclosing loop shall be exited. Execution shall continue with the command
2674 immediately following the loop.

2675 OPTIONS

2676 None.

2677 OPERANDS

2678 See the DESCRIPTION.

1

2679 STDIN

2680 Not used.

1

2681 INPUT FILES

2682 None.

2683 ENVIRONMENT VARIABLES

2684 None.

2685 ASYNCHRONOUS EVENTS

2686 Default.

1

2687 STDOUT

2688 Not used.

1

2689 STDERR

2690 The standard error shall be used only for diagnostic messages.

1

2691 OUTPUT FILES

2692 None.

2693 EXTENDED DESCRIPTION

2694 None.

2695 EXIT STATUS

2696 0 Successful completion.

2697 >0 The *n* value was not an unsigned decimal integer greater than or equal to 1.

2698 CONSEQUENCES OF ERRORS

2699 Default.

1

2700 APPLICATION USAGE

2701 None.

2702 EXAMPLES

```
2703     for i in * do
2704         if test -d "$i" then break fi done
```

2705 RATIONALE

2706 In early proposals, consideration was given to expanding the syntax of *break* and *continue* to refer
2707 to a label associated with the appropriate loop as a preferable alternative to the *n* method.
2708 However, this volume of IEEE Std 1003.1-2001 does reserve the name space of command names
2709 ending with a colon. It is anticipated that a future implementation could take advantage of this
2710 and provide something like:

```
2711     outoffloop: for i in a b c d e
2712         do
2713             for j in 0 1 2 3 4 5 6 7 8 9
2714                 do
2715                     if test -r "${i}${j}"
2716                         then break outoffloop
2717                     fi
2718                 done
2719             done
```

2720 and that this might be standardized after implementation experience is achieved.

2721 FUTURE DIRECTIONS

2722 None.

2723 SEE ALSO

2724 Section 2.14 (on page 64)

2725 CHANGE HISTORY**2726 Issue 6**

IEEE Std 1003.1-2001/Cor 1-2002, item XCU/TC1/D6/5 is applied so that the reference page 1
sections use terms as described in the Utility Description Defaults (Section 1.11). No change in 1
behavior is intended. 1

2730	NAME	
2731	colon — null utility	
2732	SYNOPSIS	
2733	: [argument ...]	
2734	DESCRIPTION	
2735	This utility shall only expand command <i>arguments</i> . It is used when a command is needed, as in	
2736	the then condition of an if command, but nothing is to be done by the command.	
2737	OPTIONS	
2738	None.	
2739	OPERANDS	
2740	See the DESCRIPTION.	1
2741	STDIN	
2742	Not used.	1
2743	INPUT FILES	
2744	None.	
2745	ENVIRONMENT VARIABLES	
2746	None.	
2747	ASYNCHRONOUS EVENTS	
2748	Default.	1
2749	STDOUT	
2750	Not used.	1
2751	STDERR	
2752	The standard error shall be used only for diagnostic messages.	1
2753	OUTPUT FILES	
2754	None.	
2755	EXTENDED DESCRIPTION	
2756	None.	
2757	EXIT STATUS	
2758	Zero.	
2759	CONSEQUENCES OF ERRORS	
2760	Default.	1
2761	APPLICATION USAGE	
2762	None.	
2763	EXAMPLES	
2764	: \${X=abc}	
2765	if false	
2766	then :	
2767	else echo \$X	
2768	fi	
2769	abc	
2770	As with any of the special built-ins, the null utility can also have variable assignments and	
2771	redirections associated with it, such as:	

2772 `x=y : > z`
2773 which sets variable **x** to the value **y** (so that it persists after the null utility completes) and creates
2774 or truncates file **z**.

2775 **RATIONALE**

2776 None.

2777 **FUTURE DIRECTIONS**

2778 None.

2779 **SEE ALSO**

2780 Section 2.14 (on page 64)

2781 **CHANGE HISTORY**

2782 Issue 6	1
2783 IEEE Std 1003.1-2001/Cor 1-2002, item XCU/TC1/D6/5 is applied so that the reference page	1
2784 sections use terms as described in the Utility Description Defaults (Section 1.11). No change in	1
2785 behavior is intended.	1

2786 NAME

2787 continue — continue for, while, or until loop

2788 SYNOPSIS

2789 continue [n]

2790 DESCRIPTION

2791 The *continue* utility shall return to the top of the smallest enclosing **for**, **while**, or **until** loop, or to
2792 the top of the *n*th enclosing loop, if *n* is specified. This involves repeating the condition list of a
2793 **while** or **until** loop or performing the next assignment of a **for** loop, and re-executing the loop if
2794 appropriate.

2795 The value of *n* is a decimal integer greater than or equal to 1. The default shall be equivalent to
2796 *n*=1. If *n* is greater than the number of enclosing loops, the outermost enclosing loop shall be
2797 used.

2798 OPTIONS

2799 None.

2800 OPERANDS

2801 See the DESCRIPTION.

1

2802 STDIN

2803 Not used.

1

2804 INPUT FILES

2805 None.

2806 ENVIRONMENT VARIABLES

2807 None.

2808 ASYNCHRONOUS EVENTS

2809 Default.

1

2810 STDOUT

2811 Not used.

1

2812 STDERR

2813 The standard error shall be used only for diagnostic messages.

1

2814 OUTPUT FILES

2815 None.

2816 EXTENDED DESCRIPTION

2817 None.

2818 EXIT STATUS

2819 0 Successful completion.

2820 >0 The *n* value was not an unsigned decimal integer greater than or equal to 1.

2821 CONSEQUENCES OF ERRORS

2822 Default.

1

2823 **APPLICATION USAGE**

2824 None.

2825 **EXAMPLES**

```
2826     for i in *
2827     do
2828         if test -d "$i"
2829             then continue
2830         fi
2831         echo "\"$i\" is not a directory.
2832     done
```

2833 **RATIONALE**

2834 None.

2835 **FUTURE DIRECTIONS**

2836 None.

2837 **SEE ALSO**

2838 Section 2.14 (on page 64)

2839 **CHANGE HISTORY**2840 **Issue 6**

2841	IEEE Std 1003.1-2001/Cor 1-2002, item XCU/TC1/D6/5 is applied so that the reference page	1
2842	sections use terms as described in the Utility Description Defaults (Section 1.11). No change in	1
2843	behavior is intended.	1

2844 NAME

2845 dot — execute commands in the current environment

2846 SYNOPSIS

2847 . *file*

2848 DESCRIPTION

2849 The shell shall execute commands from the *file* in the current environment.

2850 If *file* does not contain a slash, the shell shall use the search path specified by *PATH* to find the
2851 directory containing *file*. Unlike normal command search, however, the file searched for by the
2852 *dot* utility need not be executable. If no readable file is found, a non-interactive shell shall abort;
2853 an interactive shell shall write a diagnostic message to standard error, but this condition shall
2854 not be considered a syntax error.

2855 OPTIONS

2856 None.

2857 OPERANDS

2858 See the DESCRIPTION.

1

2859 STDIN

2860 Not used.

1

2861 INPUT FILES

2862 See the DESCRIPTION.

1

2863 ENVIRONMENT VARIABLES

2864 See the DESCRIPTION.

1

2865 ASYNCHRONOUS EVENTS

2866 Default.

1

2867 STDOUT

2868 Not used.

1

2869 STDERR

2870 The standard error shall be used only for diagnostic messages.

2871 OUTPUT FILES

2872 None.

2873 EXTENDED DESCRIPTION

2874 None.

2875 EXIT STATUS

2876 Returns the value of the last command executed, or a zero exit status if no command is executed.

2877 CONSEQUENCES OF ERRORS

2878 Default.

1

2879 **APPLICATION USAGE**

2880 None.

2881 **EXAMPLES**

```
2882     cat foobar
2883     foo=hello bar=world
2884     . foobar
2885     echo $foo $bar
2886     hello world
```

2887 **RATIONALE**

2888 Some older implementations searched the current directory for the *file*, even if the value of *PATH* disallowed it. This behavior was omitted from this volume of IEEE Std 1003.1-2001 due to concerns about introducing the susceptibility to trojan horses that the user might be trying to avoid by leaving **dot** out of *PATH*.

2892 The KornShell version of *dot* takes optional arguments that are set to the positional parameters.
2893 This is a valid extension that allows a *dot* script to behave identically to a function.

2894 **FUTURE DIRECTIONS**

2895 None.

2896 **SEE ALSO**

2897 Section 2.14 (on page 64)

2898 **CHANGE HISTORY**2899 **Issue 6**

2900 IEEE Std 1003.1-2001/Cor 1-2002, item XCU/TC1/D6/5 is applied so that the reference page
2901 sections use terms as described in the Utility Description Defaults (Section 1.11). No change in
2902 behavior is intended. 1 1 1

2903 NAME

2904 eval — construct command by concatenating arguments

2905 SYNOPSIS

2906 eval [argument ...]

2907 DESCRIPTION

2908 The *eval* utility shall construct a command by concatenating *arguments* together, separating each
2909 with a <space>. The constructed command shall be read and executed by the shell.

2910 OPTIONS

2911 None.

2912 OPERANDS

2913 See the DESCRIPTION.

1

2914 STDIN

2915 Not used.

1

2916 INPUT FILES

2917 None.

2918 ENVIRONMENT VARIABLES

2919 None.

2920 ASYNCHRONOUS EVENTS

2921 Default.

1

2922 STDOUT

2923 Not used.

1

2924 STDERR

2925 The standard error shall be used only for diagnostic messages.

1

2926 OUTPUT FILES

2927 None.

2928 EXTENDED DESCRIPTION

2929 None.

2930 EXIT STATUS

2931 If there are no *arguments*, or only null arguments, *eval* shall return a zero exit status; otherwise, it
2932 shall return the exit status of the command defined by the string of concatenated *arguments*
2933 separated by <space>s.

2934 CONSEQUENCES OF ERRORS

2935 Default.

1

2936 APPLICATION USAGE

2937 None.

2938 EXAMPLES

```
2939     foo=10 x=foo
2940     y='$'$x
2941     echo $y
2942     $foo
2943     eval y=''$'$x
2944     echo $y
2945     10
```

2946 **RATIONALE**

2947 None.

2948 **FUTURE DIRECTIONS**

2949 None.

2950 **SEE ALSO**

2951 Section 2.14 (on page 64)

2952 **CHANGE HISTORY**2953 **Issue 6**

2954	IEEE Std 1003.1-2001/Cor 1-2002, item XCU/TC1/D6/5 is applied so that the reference page	1
2955	sections use terms as described in the Utility Description Defaults (Section 1.11). No change in	1
2956	behavior is intended.	1

2957	NAME	
2958	exec — execute commands and open, close, or copy file descriptors	
2959	SYNOPSIS	
2960	exec [<i>command</i> [<i>argument</i> . . .]]	
2961	DESCRIPTION	
2962	The <i>exec</i> utility shall open, close, and/or copy file descriptors as specified by any redirections as part of the command.	
2963		
2964	If <i>exec</i> is specified without <i>command</i> or <i>arguments</i> , and any file descriptors with numbers greater than 2 are opened with associated redirection statements, it is unspecified whether those file descriptors remain open when the shell invokes another utility. Scripts concerned that child shells could misuse open file descriptors can always close them explicitly, as shown in one of the following examples.	
2965		
2966		
2967		
2968		
2969	If <i>exec</i> is specified with <i>command</i> , it shall replace the shell with <i>command</i> without creating a new process. If <i>arguments</i> are specified, they shall be arguments to <i>command</i> . Redirection affects the current shell execution environment.	
2970		
2971		
2972	OPTIONS	
2973	None.	
2974	OPERANDS	
2975	See the DESCRIPTION.	1
2976	STDIN	
2977	Not used.	1
2978	INPUT FILES	
2979	None.	
2980	ENVIRONMENT VARIABLES	
2981	None.	
2982	ASYNCHRONOUS EVENTS	
2983	Default.	1
2984	STDOUT	
2985	Not used.	1
2986	STDERR	
2987	The standard error shall be used only for diagnostic messages.	1
2988	OUTPUT FILES	
2989	None.	
2990	EXTENDED DESCRIPTION	
2991	None.	
2992	EXIT STATUS	
2993	If <i>command</i> is specified, <i>exec</i> shall not return to the shell; rather, the exit status of the process shall be the exit status of the program implementing <i>command</i> , which overlaid the shell. If <i>command</i> is not found, the exit status shall be 127. If <i>command</i> is found, but it is not an executable utility, the exit status shall be 126. If a redirection error occurs (see Section 2.8.1 (on page 46)), the shell shall exit with a value in the range 1–125. Otherwise, <i>exec</i> shall return a zero exit status.	
2994		
2995		
2996		
2997		

2998 **CONSEQUENCES OF ERRORS**

2999 Default.

1

3000 **APPLICATION USAGE**

3001 None.

3002 **EXAMPLES**3003 Open *readfile* as file descriptor 3 for reading:

3004 exec 3< readfile

3005 Open *writefile* as file descriptor 4 for writing:

3006 exec 4> writefile

3007 Make file descriptor 5 a copy of file descriptor 0:

3008 exec 5<&0

3009 Close file descriptor 3:

3010 exec 3<&-

3011 Cat the file **maggie** by replacing the current shell with the *cat* utility:

3012 exec cat maggie

3013 **RATIONALE**

3014 Most historical implementations were not conformant in that:

3015 foo=bar exec cmd

3016 did not pass **foo** to **cmd**.3017 **FUTURE DIRECTIONS**

3018 None.

3019 **SEE ALSO**

3020 Section 2.14 (on page 64)

3021 **CHANGE HISTORY**3022 **Issue 6**3023 IEEE Std 1003.1-2001/Cor 1-2002, item XCU/TC1/D6/5 is applied so that the reference page 1
3024 sections use terms as described in the Utility Description Defaults (Section 1.11). No change in 1
3025 behavior is intended. 1

3026	NAME	
3027	exit — cause the shell to exit	
3028	SYNOPSIS	
3029	exit [n]	
3030	DESCRIPTION	
3031	The <i>exit</i> utility shall cause the shell to exit with the exit status specified by the unsigned decimal integer <i>n</i> . If <i>n</i> is specified, but its value is not between 0 and 255 inclusively, the exit status is undefined.	
3034	A <i>trap</i> on EXIT shall be executed before the shell terminates, except when the <i>exit</i> utility is invoked in that <i>trap</i> itself, in which case the shell shall exit immediately.	
3036	OPTIONS	
3037	None.	
3038	OPERANDS	
3039	See the DESCRIPTION.	1
3040	STDIN	
3041	Not used.	1
3042	INPUT FILES	
3043	None.	
3044	ENVIRONMENT VARIABLES	
3045	None.	
3046	ASYNCHRONOUS EVENTS	
3047	Default.	1
3048	STDOUT	
3049	Not used.	1
3050	STDERR	
3051	The standard error shall be used only for diagnostic messages.	1
3052	OUTPUT FILES	
3053	None.	
3054	EXTENDED DESCRIPTION	
3055	None.	
3056	EXIT STATUS	
3057	The exit status shall be <i>n</i> , if specified. Otherwise, the value shall be the exit value of the last command executed, or zero if no command was executed. When <i>exit</i> is executed in a <i>trap</i> action, the last command is considered to be the command that executed immediately preceding the <i>trap</i> action.	
3061	CONSEQUENCES OF ERRORS	
3062	Default.	1

3063 APPLICATION USAGE

3064 None.

3065 EXAMPLES

3066 Exit with a *true* value:

3067 exit 0

3068 Exit with a *false* value:

3069 exit 1

3070 RATIONALE

3071 As explained in other sections, certain exit status values have been reserved for special uses and
3072 should be used by applications only for those purposes:

3073 126 A file to be executed was found, but it was not an executable utility.

3074 127 A utility to be executed was not found.

3075 >128 A command was interrupted by a signal.

3076 FUTURE DIRECTIONS

3077 None.

3078 SEE ALSO

3079 Section 2.14 (on page 64)

3080 CHANGE HISTORY**3081 Issue 6**

3082 IEEE Std 1003.1-2001/Cor 1-2002, item XCU/TC1/D6/5 is applied so that the reference page 1
3083 sections use terms as described in the Utility Description Defaults (Section 1.11). No change in 1
3084 behavior is intended. 1

3085 **NAME**

3086 *export* — set the export attribute for variables

3087 **SYNOPSIS**

3088 *export name[=word]...*

3089 *export -p*

3090 **DESCRIPTION**

3091 The shell shall give the *export* attribute to the variables corresponding to the specified *names*,
3092 which shall cause them to be in the environment of subsequently executed commands. If the
3093 name of a variable is followed by *=word*, then the value of that variable shall be set to *word*. 1

3094 The *export* special built-in shall support the Base Definitions volume of IEEE Std 1003.1-2001,
3095 Section 12.2, Utility Syntax Guidelines.

3096 When *-p* is specified, *export* shall write to the standard output the names and values of all
3097 exported variables, in the following format:

3098 "export %s=%s\n", <*name*>, <*value*>

3099 if *name* is set, and:

3100 "export %s\n", <*name*>

3101 if *name* is unset.

3102 The shell shall format the output, including the proper use of quoting, so that it is suitable for
3103 reinput to the shell as commands that achieve the same exporting results, except:

- 3104 1. Read-only variables with values cannot be reset.
- 3105 2. Variables that were unset at the time they were output need not be reset to the unset state
3106 if a value is assigned to the variable between the time the state was saved and the time at
3107 which the saved output is reinput to the shell.

3108 When no arguments are given, the results are unspecified.

3109 **OPTIONS**

3110 See the DESCRIPTION. 1

3111 **OPERANDS**

3112 See the DESCRIPTION. 1

3113 **STDIN**

3114 Not used. 1

3115 **INPUT FILES**

3116 None.

3117 **ENVIRONMENT VARIABLES**

3118 None.

3119 **ASYNCHRONOUS EVENTS**

3120 Default. 1

3121 **STDOUT**

3122 See the DESCRIPTION. 1

3123	STDERR	
3124	The standard error shall be used only for diagnostic messages.	1
3125	OUTPUT FILES	
3126	None.	
3127	EXTENDED DESCRIPTION	
3128	None.	
3129	EXIT STATUS	
3130	Zero.	
3131	CONSEQUENCES OF ERRORS	
3132	Default.	1
3133	APPLICATION USAGE	
3134	None.	
3135	EXAMPLES	
3136	Export <i>PWD</i> and <i>HOME</i> variables:	
3137	export PWD HOME	
3138	Set and export the <i>PATH</i> variable:	
3139	export PATH=/local/bin:\$PATH	
3140	Save and restore all exported variables:	
3141	export -p > temp-file	
3142	unset a lot of variables	
3143	... processing	
3144	. temp-file	
3145	RATIONALE	
3146	Some historical shells use the no-argument case as the functional equivalent of what is required here with -p . This feature was left unspecified because it is not historical practice in all shells, and some scripts may rely on the now-unspecified results on their implementations. Attempts to specify the -p output as the default case were unsuccessful in achieving consensus. The -p option was added to allow portable access to the values that can be saved and then later restored using; for example, a <i>dot</i> script.	
3152	FUTURE DIRECTIONS	
3153	None.	
3154	SEE ALSO	
3155	Section 2.14 (on page 64)	
3156	CHANGE HISTORY	
3157	Issue 6	
3158	IEEE PASC Interpretation 1003.2 #203 is applied, clarifying the format when a variable is unset.	
3159	IEEE Std 1003.1-2001/Cor 1-2002, item XCU/TC1/D6/5 is applied so that the reference page sections use terms as described in the Utility Description Defaults (Section 1.11). No change in behavior is intended.	1
3162	IEEE Std 1003.1-2001/Cor 1-2002, item XCU/TC1/D6/6 is applied, adding the following text to the end of the first paragraph of the DESCRIPTION: "If the name of a variable is followed by =word, then the value of that variable shall be set to word.". The reason for this change is that the SYNOPSIS for <i>export</i> includes:	1
3165		1

3166	export name[=word] . . .	1
3167	but the meaning of the optional “=word” is never explained in the text.	1

3168 **NAME**

3169 readonly — set the readonly attribute for variables

3170 **SYNOPSIS**3171 **readonly** name[=word]...3172 **readonly** -p3173 **DESCRIPTION**

3174 The variables whose *names* are specified shall be given the *readonly* attribute. The values of
3175 variables with the *readonly* attribute cannot be changed by subsequent assignment, nor can those
3176 variables be unset by the *unset* utility. If the name of a variable is followed by =*word*, then the
3177 value of that variable shall be set to *word*. 1

3178 The *readonly* special built-in shall support the Base Definitions volume of IEEE Std 1003.1-2001,
3179 Section 12.2, Utility Syntax Guidelines.

3180 When -p is specified, *readonly* writes to the standard output the names and values of all read-
3181 only variables, in the following format:

3182 "readonly %s=%s\n", <name>, <value>

3183 if *name* is set, and

3184 "readonly %s\n", <name>

3185 if *name* is unset.

3186 The shell shall format the output, including the proper use of quoting, so that it is suitable for
3187 reinput to the shell as commands that achieve the same value and *readonly* attribute-setting
3188 results in a shell execution environment in which:

- 3189 1. Variables with values at the time they were output do not have the *readonly* attribute set.
- 3190 2. Variables that were unset at the time they were output do not have a value at the time at
3191 which the saved output is reinput to the shell.

3192 When no arguments are given, the results are unspecified.

3193 **OPTIONS**

3194 See the DESCRIPTION. 1

3195 **OPERANDS**

3196 See the DESCRIPTION. 1

3197 **STDIN**

3198 Not used. 1

3199 **INPUT FILES**

3200 None.

3201 **ENVIRONMENT VARIABLES**

3202 None.

3203 **ASYNCHRONOUS EVENTS**

3204 Default. 1

3205 **STDOUT**

3206 See the DESCRIPTION. 1

3207	STDERR		
3208	The standard error shall be used only for diagnostic messages.		1
3209	OUTPUT FILES		
3210	None.		
3211	EXTENDED DESCRIPTION		
3212	None.		
3213	EXIT STATUS		
3214	Zero.		
3215	CONSEQUENCES OF ERRORS		
3216	Default.		1
3217	APPLICATION USAGE		
3218	None.		
3219	EXAMPLES		
3220	readonly HOME PWD		
3221	RATIONALE		
3222	Some historical shells preserve the <i>readonly</i> attribute across separate invocations. This volume of IEEE Std 1003.1-2001 allows this behavior, but does not require it.		
3224	The -p option allows portable access to the values that can be saved and then later restored using, for example, a <i>dot</i> script. Also see the RATIONALE for <i>export</i> (on page 79) for a description of the no-argument and -p output cases and a related example.		
3227	Read-only functions were considered, but they were omitted as not being historical practice or particularly useful. Furthermore, functions must not be read-only across invocations to preclude “spoofing” (spoofing is the term for the practice of creating a program that acts like a well-known utility with the intent of subverting the real intent of the user) of administrative or security-relevant (or security-conscious) shell scripts.		
3232	FUTURE DIRECTIONS		
3233	None.		
3234	SEE ALSO		
3235	Section 2.14 (on page 64)		
3236	CHANGE HISTORY		
3237	Issue 6		
3238	IEEE PASC Interpretation 1003.2 #203 is applied, clarifying the format when a variable is unset.		
3239	IEEE Std 1003.1-2001/Cor 1-2002, item XCU/TC1/D6/5 is applied so that the reference page sections use terms as described in the Utility Description Defaults (Section 1.11). No change in behavior is intended.	1	1
3242	IEEE Std 1003.1-2001/Cor 1-2002, item XCU/TC1/D6/7 is applied, adding the following text to the end of the first paragraph of the DESCRIPTION: “If the name of a variable is followed by =word , then the value of that variable shall be set to word .”. The reason for this change is that the SYNOPSIS for <i>readonly</i> includes:	1	1
3246	readonly name[=word]...	1	1
3247	but the meaning of the optional “ =word ” is never explained in the text.	1	1

3248	NAME	
3249	return — return from a function	
3250	SYNOPSIS	
3251	return [n]	
3252	DESCRIPTION	
3253	The <i>return</i> utility shall cause the shell to stop executing the current function or <i>dot</i> script. If the	
3254	shell is not currently executing a function or <i>dot</i> script, the results are unspecified.	
3255	OPTIONS	
3256	None.	
3257	OPERANDS	
3258	See the DESCRIPTION.	1
3259	STDIN	
3260	Not used.	1
3261	INPUT FILES	
3262	None.	
3263	ENVIRONMENT VARIABLES	
3264	None.	
3265	ASYNCHRONOUS EVENTS	
3266	Default.	1
3267	STDOUT	
3268	Not used.	1
3269	STDERR	
3270	The standard error shall be used only for diagnostic messages.	1
3271	OUTPUT FILES	
3272	None.	
3273	EXTENDED DESCRIPTION	
3274	None.	
3275	EXIT STATUS	
3276	The value of the special parameter '?' shall be set to <i>n</i> , an unsigned decimal integer, or to the	
3277	exit status of the last command executed if <i>n</i> is not specified. If the value of <i>n</i> is greater than 255,	
3278	the results are undefined. When <i>return</i> is executed in a <i>trap</i> action, the last command is	
3279	considered to be the command that executed immediately preceding the <i>trap</i> action.	
3280	CONSEQUENCES OF ERRORS	
3281	Default.	1
3282	APPLICATION USAGE	
3283	None.	
3284	EXAMPLES	
3285	None.	
3286	RATIONALE	
3287	The behavior of <i>return</i> when not in a function or <i>dot</i> script differs between the System V shell	
3288	and the KornShell. In the System V shell this is an error, whereas in the KornShell, the effect is	
3289	the same as <i>exit</i> .	

The results of returning a number greater than 255 are undefined because of differing practices in the various historical implementations. Some shells AND out all but the low-order 8 bits; others allow larger values, but not of unlimited size.

3293 See the discussion of appropriate exit status values under *exit* (on page 77).

3294 FUTURE DIRECTIONS

3295 None.

3296 SEE ALSO

3297 Section 2.14 (on page 64)

3298 CHANGE HISTORY

3299 Issue 6

IEEE Std 1003.1-2001/Cor 1-2002, item XCU/TC1/D6/5 is applied so that the reference page sections use terms as described in the Utility Description Defaults (Section 1.11). No change in behavior is intended.

3303 **NAME**

3304 set — set or unset options and positional parameters

3305 **SYNOPSIS**

3306 XSI set [-abCefmnuvx][-h][-o option][argument...]

3307 XSI set [+abCefmnuvx][+h][+o option][argument...]

3308 set -- [argument...]

3309 set -o

3310 set +o

3311 **DESCRIPTION**

3312 If no *options* or *arguments* are specified, *set* shall write the names and values of all shell variables
3313 in the collation sequence of the current locale. Each *name* shall start on a separate line, using the
3314 format:

3315 "%s=%s\n", <name>, <value>

3316 The *value* string shall be written with appropriate quoting; see the description of shell quoting in
3317 Section 2.2 (on page 30). The output shall be suitable for reinput to the shell, setting or resetting,
3318 as far as possible, the variables that are currently set; read-only variables cannot be reset.

3319 When options are specified, they shall set or unset attributes of the shell, as described below.
3320 When *arguments* are specified, they cause positional parameters to be set or unset, as described
3321 below. Setting or unsetting attributes and positional parameters are not necessarily related
3322 actions, but they can be combined in a single invocation of *set*.

3323 The *set* special built-in shall support the Base Definitions volume of IEEE Std 1003.1-2001,
3324 Section 12.2, Utility Syntax Guidelines except that options can be specified with either a leading
3325 hyphen (meaning enable the option) or plus sign (meaning disable it) unless otherwise specified.

3326 Implementations shall support the options in the following list in both their hyphen and plus-
3327 sign forms. These options can also be specified as options to *sh*.

3328 **-a** When this option is on, the *export* attribute shall be set for each variable to which an
3329 assignment is performed; see the Base Definitions volume of IEEE Std 1003.1-2001, Section
3330 4.21, Variable Assignment. If the assignment precedes a utility name in a command, the
3331 *export* attribute shall not persist in the current execution environment after the utility
3332 completes, with the exception that preceding one of the special built-in utilities causes the
3333 *export* attribute to persist after the built-in has completed. If the assignment does not
3334 precede a utility name in the command, or if the assignment is a result of the operation of
3335 the *getopts* or *read* utilities, the *export* attribute shall persist until the variable is unset.

3336 **-b** This option shall be supported if the implementation supports the User Portability Utilities
3337 option. It shall cause the shell to notify the user asynchronously of background job
3338 completions. The following message is written to standard error:

3339 "[%d]%c %s%s\n", <job-number>, <current>, <status>, <job-name>

3340 where the fields shall be as follows:

3341 <current> The character '+' identifies the job that would be used as a default for
3342 the *fg* or *bg* utilities; this job can also be specified using the *job_id* "%+" or
3343 "%%". The character '-' identifies the job that would become the default
3344 if the current default job were to exit; this job can also be specified using
3345 the *job_id* "%-". For other jobs, this field is a <space>. At most one job
3346 can be identified with '+' and at most one job can be identified with '-'.

3347		If there is any suspended job, then the current job shall be a suspended job. If there are at least two suspended jobs, then the previous job also shall be a suspended job.
3350	<job-number>	A number that can be used to identify the process group to the <i>wait</i> , <i>fg</i> , <i>bg</i> , and <i>kill</i> utilities. Using these utilities, the job can be identified by prefixing the job number with '%'.
3353	<status>	Unspecified.
3354	<job-name>	Unspecified.
3355		When the shell notifies the user a job has been completed, it may remove the job's process ID from the list of those known in the current shell execution environment; see Section 2.9.3.1 (on page 50). Asynchronous notification shall not be enabled by default.
3358	-C	(Uppercase C.) Prevent existing files from being overwritten by the shell's '>' redirection operator (see Section 2.7.2 (on page 44)); the "> " redirection operator shall override this <i>noclobber</i> option for an individual file.
3361	-e	When this option is on, if a simple command fails for any of the reasons listed in Section 2.8.1 (on page 46) or returns an exit status value >0, and is not part of the compound list following a while , until , or if keyword, and is not a part of an AND or OR list, and is not a pipeline preceded by the ! reserved word, then the shell shall immediately exit.
3365	-f	The shell shall disable pathname expansion.
3366 XSI	-h	Locate and remember utilities invoked by functions as those functions are defined (the utilities are normally located when the function is executed).
3367		
3368	-m	This option shall be supported if the implementation supports the User Portability Utilities option. All jobs shall be run in their own process groups. Immediately before the shell issues a prompt after completion of the background job, a message reporting the exit status of the background job shall be written to standard error. If a foreground job stops, the shell shall write a message to standard error to that effect, formatted as described by the <i>jobs</i> utility. In addition, if a job changes status other than exiting (for example, if it stops for input or output or is stopped by a SIGSTOP signal), the shell shall write a similar message immediately prior to writing the next prompt. This option is enabled by default for interactive shells.
3377	-n	The shell shall read commands but does not execute them; this can be used to check for shell script syntax errors. An interactive shell may ignore this option.
3378		
3379	-o	Write the current settings of the options to standard output in an unspecified format.
3380	+o	Write the current option settings to standard output in a format that is suitable for reinput to the shell as commands that achieve the same options settings.
3381		
3382	-o option	This option is supported if the system supports the User Portability Utilities option. It shall set various options, many of which shall be equivalent to the single option letters. The following values of <i>option</i> shall be supported:
3386	allexport	Equivalent to -a .
3387	errexit	Equivalent to -e .
3388	ignoreeof	Prevent an interactive shell from exiting on end-of-file. This setting prevents accidental logouts when <control>-D is entered. A user shall explicitly <i>exit</i> to leave the interactive shell.
3389		
3390		

3391	<i>monitor</i>	Equivalent to -m . This option is supported if the system supports the User Portability Utilities option.
3392		
3393	<i>noclobber</i>	Equivalent to -C (uppercase C).
3394	<i>noglob</i>	Equivalent to -f .
3395	<i>noexec</i>	Equivalent to -n .
3396	<i>nolog</i>	Prevent the entry of function definitions into the command history; see Command History List (on page 855).
3397		
3398	<i>notify</i>	Equivalent to -b .
3399	<i>nounset</i>	Equivalent to -u .
3400	<i>verbose</i>	Equivalent to -v .
3401	<i>vi</i>	Allow shell command line editing using the built-in <i>vi</i> editor. Enabling <i>vi</i> mode shall disable any other command line editing mode provided as an implementation extension.
3402		
3403		
3404		It need not be possible to set <i>vi</i> mode on for certain block-mode terminals.
3405	<i>xtrace</i>	Equivalent to -x .
3406	-u	The shell shall write a message to standard error when it tries to expand a variable that is not set and immediately exit. An interactive shell shall not exit.
3407		
3408	-v	The shell shall write its input to standard error as it is read.
3409	-x	The shell shall write to standard error a trace for each command after it expands the command and before it executes it. It is unspecified whether the command that turns tracing off is traced.
3410		
3411		
3412		The default for all these options shall be off (unset) unless stated otherwise in the description of the option or unless the shell was invoked with them on; see <i>sh</i> .
3413		
3414		The remaining arguments shall be assigned in order to the positional parameters. The special parameter ' # ' shall be set to reflect the number of positional parameters. All positional parameters shall be unset before any new values are assigned.
3415		
3416		
3417		The special argument " -- " immediately following the <i>set</i> command name can be used to delimit the arguments if the first argument begins with '+' or '-', or to prevent inadvertent listing of all shell variables when there are no arguments. The command <i>set --</i> without <i>argument</i> shall unset all positional parameters and set the special parameter ' # ' to zero.
3418		
3419		
3420		

3421 OPTIONS

3422 See the DESCRIPTION.

1

3423 OPERANDS

3424 See the DESCRIPTION.

1

3425 STDIN

3426 Not used.

1

3427 INPUT FILES

3428 None.

3429	ENVIRONMENT VARIABLES	
3430	None.	
3431	ASYNCHRONOUS EVENTS	
3432	Default.	1
3433	STDOUT	
3434	See the DESCRIPTION.	1
3435	STDERR	
3436	The standard error shall be used only for diagnostic messages.	1
3437	OUTPUT FILES	
3438	None.	
3439	EXTENDED DESCRIPTION	
3440	None.	
3441	EXIT STATUS	
3442	Zero.	
3443	CONSEQUENCES OF ERRORS	
3444	Default.	1
3445	APPLICATION USAGE	
3446	None.	
3447	EXAMPLES	
3448	Write out all variables and their values:	
3449	set	
3450	Set \$1, \$2, and \$3 and set "\$#" to 3:	
3451	set c a b	
3452	Turn on the -x and -v options:	
3453	set -xv	
3454	Unset all positional parameters:	
3455	set --	
3456	Set \$1 to the value of x, even if it begins with '-' or '+':	
3457	set -- "\$x"	
3458	Set the positional parameters to the expansion of x, even if x expands with a leading '-' or '+':	
3459	set -- \$x	
3460	RATIONALE	
3461	The <i>set --</i> form is listed specifically in the SYNOPSIS even though this usage is implied by the Utility Syntax Guidelines. The explanation of this feature removes any ambiguity about whether the <i>set --</i> form might be misinterpreted as being equivalent to <i>set</i> without any options or arguments. The functionality of this form has been adopted from the KornShell. In System V, <i>set --</i> only unsets parameters if there is at least one argument; the only way to unset all parameters is to use <i>shift</i> . Using the KornShell version should not affect System V scripts because there should be no reason to issue it without arguments deliberately; if it were issued as, for example:	
3468	set -- "\$@"	

3469 and there were in fact no arguments resulting from "\$@", unsetting the parameters would have
3470 no result.

3471 The *set +* form in early proposals was omitted as being an unnecessary duplication of *set* alone
3472 and not widespread historical practice.

3473 The *noclobber* option was changed to allow *set -C* as well as the *set -o noclobber* option. The
3474 single-letter version was added so that the historical "\$-" paradigm would not be broken; see
3475 Section 2.5.2 (on page 34).

3476 The **-h** flag is related to command name hashing and is only required on XSI-conformant
3477 systems.

3478 The following *set* flags were omitted intentionally with the following rationale:

- 3479 **-k** The **-k** flag was originally added by the author of the Bourne shell to make it easier for
3480 users of pre-release versions of the shell. In early versions of the Bourne shell the construct
3481 *set name=value* had to be used to assign values to shell variables. The problem with **-k** is
3482 that the behavior affects parsing, virtually precluding writing any compilers. To explain the
3483 behavior of **-k**, it is necessary to describe the parsing algorithm, which is implementation-
3484 defined. For example:

3485 `set -k; echo name=value`

3486 and:

3487 `set -k`
3488 `echo name=value`

3489 behave differently. The interaction with functions is even more complex. What is more, the
3490 **-k** flag is never needed, since the command line could have been reordered.

- 3491 **-t** The **-t** flag is hard to specify and almost never used. The only known use could be done
3492 with here-documents. Moreover, the behavior with *ksh* and *sh* differs. The reference page
3493 says that it exits after reading and executing one command. What is one command? If the
3494 input is *date;date*, *sh* executes both *date* commands while *ksh* does only the first.

3495 Consideration was given to rewriting *set* to simplify its confusing syntax. A specific suggestion
3496 was that the *unset* utility should be used to unset options instead of using the non-*getopt()*-able
3497 *+option* syntax. However, the conclusion was reached that the historical practice of using *+option*
3498 was satisfactory and that there was no compelling reason to modify such widespread historical
3499 practice.

3500 The **-o** option was adopted from the KornShell to address user needs. In addition to its generally
3501 friendly interface, **-o** is needed to provide the *vi* command line editing mode, for which
3502 historical practice yields no single-letter option name. (Although it might have been possible to
3503 invent such a letter, it was recognized that other editing modes would be developed and **-o**
3504 provides ample name space for describing such extensions.)

3505 Historical implementations are inconsistent in the format used for **-o** option status reporting.
3506 The **+o** format without an option-argument was added to allow portable access to the options
3507 that can be saved and then later restored using, for instance, a dot script.

3508 Historically, *sh* did trace the command *set +x*, but *ksh* did not.

3509 The *ignoreeof* setting prevents accidental logouts when the end-of-file character (typically
3510 <control>-D) is entered. A user shall explicitly *exit* to leave the interactive shell.

3511 The *set -m* option was added to apply only to the UPE because it applies primarily to interactive
3512 use, not shell script applications.

3513 The ability to do asynchronous notification became available in the 1988 version of the
3514 KornShell. To have it occur, the user had to issue the command:

3515 trap "jobs -n" CLD

3516 The C shell provides two different levels of an asynchronous notification capability. The
3517 environment variable *notify* is analogous to what is done in *set -b* or *set -o notify*. When set, it
3518 notifies the user immediately of background job completions. When unset, this capability is
3519 turned off.

3520 The other notification ability comes through the built-in utility *notify*. The syntax is:

3521 *notify* [%job ...]

1

3522 By issuing *notify* with no operands, it causes the C shell to notify the user asynchronously when
3523 the state of the current job changes. If given operands, *notify* asynchronously informs the user of
3524 changes in the states of the specified jobs.

1

3525 To add asynchronous notification to the POSIX shell, neither the KornShell extensions to *trap*,
3526 nor the C shell *notify* environment variable seemed appropriate (*notify* is not a proper POSIX
3527 environment variable name).

3528 The *set -b* option was selected as a compromise.

3529 The *notify* built-in was considered to have more functionality than was required for simple
3530 asynchronous notification.

3531 FUTURE DIRECTIONS

3532 None.

3533 SEE ALSO

3534 Section 2.14 (on page 64)

3535 CHANGE HISTORY

3536 Issue 6

3537 The obsolescent *set* command name followed by '-' has been removed.

3538 The following new requirements on POSIX implementations derive from alignment with the
3539 Single UNIX Specification:

- The *nolog* option is added to *set -o*.

3541 IEEE PASC Interpretation 1003.2 #167 is applied, clarifying that the options default also takes
3542 into account the description of the option.

3543 IEEE Std 1003.1-2001/Cor 1-2002, item XCU/TC1/D6/5 is applied so that the reference page
3544 sections use terms as described in the Utility Description Defaults (Section 1.11). No change in
3545 behavior is intended.

1

3546 IEEE Std 1003.1-2001/Cor 1-2002, item XCU/TC1/D6/8 is applied, changing the square brackets
3547 in the example in RATIONALE to be in bold, which is the typeface used for optional items.

1

1

3548 **NAME**

3549 shift — shift positional parameters

3550 **SYNOPSIS**

3551 shift [n]

3552 **DESCRIPTION**

3553 The positional parameters shall be shifted. Positional parameter 1 shall be assigned the value of parameter $(1+n)$, parameter 2 shall be assigned the value of parameter $(2+n)$, and so on. The parameters represented by the numbers "\$#" down to "\$#-n+1" shall be unset, and the parameter '#' is updated to reflect the new number of positional parameters.

3557 The value n shall be an unsigned decimal integer less than or equal to the value of the special parameter '#'. If n is not given, it shall be assumed to be 1. If n is 0, the positional and special parameters are not changed.

3560 **OPTIONS**

3561 None.

3562 **OPERANDS**

3563 See the DESCRIPTION.

1

3564 **STDIN**

3565 Not used.

1

3566 **INPUT FILES**

3567 None.

3568 **ENVIRONMENT VARIABLES**

3569 None.

3570 **ASYNCHRONOUS EVENTS**

3571 Default.

1

3572 **STDOUT**

3573 Not used.

1

3574 **STDERR**

3575 The standard error shall be used only for diagnostic messages.

1

3576 **OUTPUT FILES**

3577 None.

3578 **EXTENDED DESCRIPTION**

3579 None.

3580 **EXIT STATUS**

3581 The exit status is >0 if $n > \$\#$; otherwise, it is zero.

3582 **CONSEQUENCES OF ERRORS**

3583 Default.

1

3584 **APPLICATION USAGE**

3585 None.

3586 **EXAMPLES**

```
3587     $ set a b c d e
3588     $ shift 2
3589     $ echo $*
3590     c d e
```

3591 **RATIONALE**

3592 None.

3593 **FUTURE DIRECTIONS**

3594 None.

3595 **SEE ALSO**

3596 Section 2.14 (on page 64)

3597 **CHANGE HISTORY**3598 **Issue 6**

3599 IEEE Std 1003.1-2001/Cor 1-2002, item XCU/TC1/D6/5 is applied so that the reference page	1
3600 sections use terms as described in the Utility Description Defaults (Section 1.11). No change in	1
3601 behavior is intended.	1

3602	NAME	
3603	times — write process times	
3604	SYNOPSIS	
3605	times	
3606	DESCRIPTION	
3607	The <i>times</i> utility shall write the accumulated user and system times for the shell and for all of its child processes, in the following POSIX locale format:	1
3609	"%dm%fs %dm%fs\n%dm%fs %dm%fs\n", <shell user minutes>,	
3610	<shell user seconds>, <shell system minutes>,	
3611	<shell system seconds>, <children user minutes>,	
3612	<children user seconds>, <children system minutes>,	
3613	<children system seconds>	
3614	The four pairs of times shall correspond to the members of the <sys/times.h> tms structure	
3615	(defined in the Base Definitions volume of IEEE Std 1003.1-2001, Chapter 13, Headers) as	
3616	returned by <i>times()</i> : <i>tms_utime</i> , <i>tms_stime</i> , <i>tms_cutime</i> , and <i>tms_cstime</i> , respectively.	
3617	OPTIONS	
3618	None.	
3619	OPERANDS	
3620	None.	
3621	STDIN	
3622	Not used.	1
3623	INPUT FILES	
3624	None.	
3625	ENVIRONMENT VARIABLES	
3626	None.	
3627	ASYNCHRONOUS EVENTS	
3628	Default.	1
3629	STDOUT	
3630	See the DESCRIPTION.	1
3631	STDERR	
3632	The standard error shall be used only for diagnostic messages.	1
3633	OUTPUT FILES	
3634	None.	
3635	EXTENDED DESCRIPTION	
3636	None.	
3637	EXIT STATUS	
3638	Zero.	
3639	CONSEQUENCES OF ERRORS	
3640	Default.	1

3641 **APPLICATION USAGE**

3642 None.

3643 **EXAMPLES**

```
3644 $ times
3645 0m0.43s 0m1.11s
3646 8m44.18s 1m43.23s
```

3647 **RATIONALE**

3648 The *times* special built-in from the Single UNIX Specification is now required for all conforming
3649 shells.

3650 **FUTURE DIRECTIONS**

3651 None.

3652 **SEE ALSO**

3653 Section 2.14 (on page 64)

3654 **CHANGE HISTORY**3655 **Issue 6**

3656 IEEE Std 1003.1-2001/Cor 1-2002, item XCU/TC1/D6/9 is applied, changing text in the	1
3657 DESCRIPTION from: "Write the accumulated user and system times for the shell and for all of	1
3658 its child processes ..." to: "The <i>times</i> utility shall write the accumulated user and system times	1
3659 for the shell and for all of its child processes ...".	1

3660 **NAME**

3661 trap — trap signals

3662 **SYNOPSIS**3663 trap [*action condition ...*]3664 **DESCRIPTION**

3665 If *action* is ‘-’, the shell shall reset each *condition* to the default value. If *action* is null (" "), the
3666 shell shall ignore each specified *condition* if it arises. Otherwise, the argument *action* shall be read
3667 and executed by the shell when one of the corresponding conditions arises. The action of *trap*
3668 shall override a previous action (either default action or one explicitly set). The value of "\$?"
3669 after the *trap* action completes shall be the value it had before *trap* was invoked.

3670 The condition can be EXIT, 0 (equivalent to EXIT), or a signal specified using a symbolic name,
3671 without the SIG prefix, as listed in the tables of signal names in the <signal.h> header defined in
3672 the Base Definitions volume of IEEE Std 1003.1-2001, Chapter 13, Headers; for example, HUP,
3673 INT, QUIT, TERM. Implementations may permit names with the SIG prefix or ignore case in
3674 signal names as an extension. Setting a trap for SIGKILL or SIGSTOP produces undefined
3675 results.

3676 The environment in which the shell executes a *trap* on EXIT shall be identical to the environment
3677 immediately after the last command executed before the *trap* on EXIT was taken.

3678 Each time *trap* is invoked, the *action* argument shall be processed in a manner equivalent to:

3679 eval *action*

3680 Signals that were ignored on entry to a non-interactive shell cannot be trapped or reset, although
3681 no error need be reported when attempting to do so. An interactive shell may reset or catch
3682 signals ignored on entry. Traps shall remain in place for a given shell until explicitly changed
3683 with another *trap* command.

3684 When a subshell is entered, traps that are not being ignored are set to the default actions. This
3685 does not imply that the *trap* command cannot be used within the subshell to set new traps.

3686 The *trap* command with no arguments shall write to standard output a list of commands
3687 associated with each condition. The format shall be:

3688 "trap -- %s %s ... \n", <*action*>, <*condition*> ...

3689 The shell shall format the output, including the proper use of quoting, so that it is suitable for
3690 reinput to the shell as commands that achieve the same trapping results. For example:

3691 save_traps=\$(trap)
3692 ...
3693 eval "\$save_traps"

3694 XSI-conformant systems also allow numeric signal numbers for the conditions corresponding to
3695 the following signal names: 1

3696 1	SIGHUP	1
3697 2	SIGINT	1
3698 3	SIGQUIT	1
3699 6	SIGABRT	1
3700 9	SIGKILL	1
3701 14	SIGALRM	1

3702	15 SIGTERM	1
3703		
3704	The <i>trap</i> special built-in shall conform to the Base Definitions volume of IEEE Std 1003.1-2001,	
3705	Section 12.2, Utility Syntax Guidelines.	
3706	OPTIONS	
3707	None.	
3708	OPERANDS	
3709	See the DESCRIPTION.	1
3710	STDIN	
3711	Not used.	1
3712	INPUT FILES	
3713	None.	
3714	ENVIRONMENT VARIABLES	
3715	None.	
3716	ASYNCHRONOUS EVENTS	
3717	Default.	1
3718	STDOUT	
3719	See the DESCRIPTION.	1
3720	STDERR	
3721	The standard error shall be used only for diagnostic messages.	1
3722	OUTPUT FILES	
3723	None.	
3724	EXTENDED DESCRIPTION	
3725	None.	
3726	EXIT STATUS	
3727	XSI If the trap name or number is invalid, a non-zero exit status shall be returned; otherwise, zero	
3728	XSI shall be returned. For both interactive and non-interactive shells, invalid signal names or	
3729	numbers shall not be considered a syntax error and do not cause the shell to abort.	
3730	CONSEQUENCES OF ERRORS	
3731	Default.	1
3732	APPLICATION USAGE	
3733	None.	
3734	EXAMPLES	
3735	Write out a list of all traps and actions:	
3736	trap	
3737	Set a trap so the <i>logout</i> utility in the directory referred to by the <i>HOME</i> environment variable	
3738	executes when the shell terminates:	
3739	trap '\$HOME/logout' EXIT	
3740	or:	
3741	trap '\$HOME/logout' 0	
3742	Unset traps on INT, QUIT, TERM, and EXIT:	

3743 trap - INT QUIT TERM EXIT

3744 **RATIONALE**

3745 Implementations may permit lowercase signal names as an extension. Implementations may
3746 also accept the names with the SIG prefix; no known historical shell does so. The *trap* and *kill*
3747 utilities in this volume of IEEE Std 1003.1-2001 are now consistent in their omission of the SIG
3748 prefix for signal names. Some *kill* implementations do not allow the prefix, and *kill -l* lists the
3749 signals without prefixes.

3750 Trapping SIGKILL or SIGSTOP is syntactically accepted by some historical implementations, but
3751 it has no effect. Portable POSIX applications cannot attempt to trap these signals.

3752 The output format is not historical practice. Since the output of historical *trap* commands is not
3753 portable (because numeric signal values are not portable) and had to change to become so, an
3754 opportunity was taken to format the output in a way that a shell script could use to save and
3755 then later reuse a trap if it wanted.

3756 The KornShell uses an **ERR** trap that is triggered whenever *set -e* would cause an exit. This is
3757 allowable as an extension, but was not mandated, as other shells have not used it.

3758 The text about the environment for the EXIT trap invalidates the behavior of some historical
3759 versions of interactive shells which, for example, close the standard input before executing a
3760 trap on 0. For example, in some historical interactive shell sessions the following trap on 0 would
3761 always print "--":

3762 trap 'read foo; echo "-\$foo-' 0

3763 **FUTURE DIRECTIONS**

3764 None.

3765 **SEE ALSO**

3766 Section 2.14 (on page 64)

3767 **CHANGE HISTORY**

3768 **Issue 6**

3769 XSI-conforming implementations provide the mapping of signal names to numbers given above
3770 (previously this had been marked obsolescent). Other implementations need not provide this
3771 optional mapping.

3772 IEEE Std 1003.1-2001/Cor 1-2002, item XCU/TC1/D6/5 is applied so that the reference page 1
3773 sections use terms as described in the Utility Description Defaults (Section 1.11). No change in 1
3774 behavior is intended. 1

3775 NAME

3776 unset — unset values and attributes of variables and functions

3777 SYNOPSIS

3778 `unset [-fv] name ...`

3779 DESCRIPTION

3780 Each variable or function specified by *name* shall be unset.

3781 If **-v** is specified, *name* refers to a variable name and the shell shall unset it and remove it from
3782 the environment. Read-only variables cannot be unset.

3783 If **-f** is specified, *name* refers to a function and the shell shall unset the function definition.

3784 If neither **-f** nor **-v** is specified, *name* refers to a variable; if a variable by that name does not
3785 exist, it is unspecified whether a function by that name, if any, shall be unset.

3786 Unsetting a variable or function that was not previously set shall not be considered an error and
3787 does not cause the shell to abort.

3788 The *unset* special built-in shall support the Base Definitions volume of IEEE Std 1003.1-2001,
3789 Section 12.2, Utility Syntax Guidelines.

3790 Note that:

3791 VARIABLE=

3792 is not equivalent to an *unset* of VARIABLE; in the example, VARIABLE is set to " ". Also, the
3793 variables that can be *unset* should not be misinterpreted to include the special parameters (see
3794 Section 2.5.2 (on page 34)).

3795 OPTIONS

3796 See the DESCRIPTION.

1

3797 OPERANDS

3798 See the DESCRIPTION.

1

3799 STDIN

3800 Not used.

1

3801 INPUT FILES

3802 None.

3803 ENVIRONMENT VARIABLES

3804 None.

3805 ASYNCHRONOUS EVENTS

3806 Default.

1

3807 STDOUT

3808 Not used.

1

3809 STDERR

3810 The standard error shall be used only for diagnostic messages.

1

3811 OUTPUT FILES

3812 None.

3813 EXTENDED DESCRIPTION

3814 None.

3815 EXIT STATUS

3816 0 All *name* operands were successfully unset.
3817 >0 At least one *name* could not be unset.

3818 CONSEQUENCES OF ERRORS

3819 Default.

1

3820 APPLICATION USAGE

3821 None.

3822 EXAMPLES

3823 Unset *VISUAL* variable:

3824 `unset -v VISUAL`

3825 Unset the functions **foo** and **bar**:

3826 `unset -f foo bar`

3827 RATIONALE

3828 Consideration was given to omitting the **-f** option in favor of an *unfunction* utility, but the
3829 standard developers decided to retain historical practice.

3830 The **-v** option was introduced because System V historically used one name space for both
3831 variables and functions. When *unset* is used without options, System V historically unset either a
3832 function or a variable, and there was no confusion about which one was intended. A portable
3833 POSIX application can use *unset* without an option to unset a variable, but not a function; the **-f**
3834 option must be used.

3835 FUTURE DIRECTIONS

3836 None.

3837 SEE ALSO

3838 Section 2.14 (on page 64)

3839 CHANGE HISTORY**3840 Issue 6**

3841 IEEE Std 1003.1-2001/Cor 1-2002, item XCU/TC1/D6/5 is applied so that the reference page
3842 sections use terms as described in the Utility Description Defaults (Section 1.11). No change in
3843 behavior is intended.

1

1

1

Batch Environment Services

3845 BE This chapter describes the services and utilities that shall be implemented on all systems that
 3846 claim conformance to the Batch Environment Services and Utilities option. This functionality is
 3847 dependent on support of this option (and the rest of this section is not further shaded for this
 3848 option). 1

3849 3.1 General Concepts

3850 3.1.1 Batch Client-Server Interaction

3851 Batch jobs are created and managed by batch servers. A batch client interacts with a batch server
 3852 to access batch services on behalf of the user. In order to use batch services, a user must have
 3853 access to a batch client.

3854 A batch server is a computational entity, such as a daemon process, that provides batch services.
 3855 Batch servers route, queue, modify, and execute batch jobs on behalf of batch clients.

3856 The batch utilities described in this volume of IEEE Std 1003.1-2001 (and listed in Table 3-1) are
 3857 clients of batch services; they allow users to perform actions on the job such as creating,
 3858 modifying, and deleting batch jobs from a shell command line. Although these batch utilities
 3859 may be said to accomplish certain services, they actually obtain services on behalf of a user by
 3860 means of requests to batch servers.

3861 **Table 3-1** Batch Utilities

<i>qalter</i>	<i>qmove</i>	<i>qrsl</i>	<i>qstat</i>
<i>qdel</i>	<i>qmsg</i>	<i>qselect</i>	<i>qsub</i>
<i>qhold</i>	<i>qrerun</i>	<i>qsig</i>	

3865 Client-server interaction takes place by means of the batch requests defined in this chapter.
 3866 Because direct access to batch jobs and queues is limited to batch servers, clients and servers of
 3867 different implementations can interoperate, since dependencies on private structures for batch
 3868 jobs and queues are limited to batch servers. Also, batch servers may be clients of other batch
 3869 servers.

3870 3.1.2 Batch Queues

3871 Two types of batch queue are described: routing queues and execution queues. When a batch job
 3872 is placed in a routing queue, it is a candidate for routing. A batch job is removed from routing
 3873 queues under the following conditions:

- 3874 • The batch job has been routed to another queue.
- 3875 • The batch job has been deleted from the batch queue.
- 3876 • The batch job has been aborted.

3877 When a batch job is placed in an execution queue, it is a candidate for execution.

3878 A batch job is removed from an execution queue under the following conditions:

- 3879 • The batch job has been executed and exited.
3880 • The batch job has been aborted.
3881 • The batch job has been deleted from the batch queue.
3882 • The batch job has been moved to another queue.

3883 Access to a batch queue is limited to the batch server that manages the batch queue. Clients
3884 never access a batch queue or a batch job directly, either to read or write information; all client
3885 access to batch queues or jobs takes place through batch servers.

3886 3.1.3 Batch Job Creation

3887 When a batch server creates a batch job on behalf of a client, it shall assign a batch job identifier
3888 to the job. A batch job identifier consists of both a sequence number that is unique among the
3889 sequence numbers issued by that server and the name of the server. Since the batch server name
3890 is unique within a name space, the job identifier is likewise unique within the name space.

3891 The batch server that creates a batch job shall return the batch server-assigned job identifier to
3892 the client that requested the job creation. If the batch server routes or moves the job to another
3893 server, it sends the job identifier with the job. Once assigned, the job identifier of a batch job shall
3894 never change.

3895 3.1.4 Batch Job Tracking

3896 Since a batch job may be moved after creation, the batch server name component of the job
3897 identifier need not indicate the location of the job. An implementation may provide a batch job
3898 tracking mechanism, in which case the user generally does not need to know the location of the
3899 job. However, an implementation need not provide a batch job tracking mechanism, in which
3900 case the user must find routed jobs by probing the possible destinations.

3901 3.1.5 Batch Job Routing

3902 To route a batch job, a batch server either moves the job to some other queue that is managed by
3903 the batch server, or requests that some other batch server accept the job.

3904 Each routing queue has one or more queues to which it can route batch jobs. The batch server
3905 administrator creates routing queues.

3906 A batch server may route a batch job from a routing queue to another routing queue. Batch
3907 servers shall prevent or otherwise handle cases of circular routing paths. As a deferred service, a
3908 batch server routes jobs from the routing queues that it manages. The algorithm by which a
3909 batch server selects a batch queue to which to route a batch job is implementation-defined.

3910 A batch job need not be eligible for routing to all the batch queues fed by the routing queue from
3911 which it is routed. A batch server that has been asked to accept the job may reject the request if
3912 the job requires resources that are unavailable to that batch server, or if the client is not
3913 authorized to access the batch server.

3914 Batch servers may route high-priority jobs before low-priority jobs, but, on other than
3915 overloaded systems, the effect may be imperceptible to the user. If all the batch servers fed by a
3916 routing queue reject requests to accept the job for reasons that are permanent, the batch server
3917 that manages the job shall abort the job. If all or some rejections are temporary, the batch server
3918 should try to route the job again at some later point.

3919 The reasons for rejecting a batch job are implementation-defined.

3920 The reasons for which the routing should be retried later and the reasons for which the job
3921 should be aborted are also implementation-defined.

3922 3.1.6 Batch Job Execution

3923 To execute a batch job is to create a session leader (a process) that runs the shell program
3924 indicated by the *Shell_Path* attribute of the job. The script shall be passed to the program as its
3925 standard input. An implementation may pass the script to the program by other
3926 implementation-defined means. At the time a batch job begins execution, it is defined to enter
3927 the RUNNING state. The primary program that is executed by a batch job is typically, though
3928 not necessarily, a shell program.

3929 A batch server shall execute eligible jobs as a deferred service—no client request is necessary
3930 once the batch job is created and eligible. However, the attributes of a batch job, such as the job
3931 hold type, may render the job ineligible. A batch server shall scan the execution queues that it
3932 manages for jobs that are eligible for execution. The algorithm by which the batch server selects
3933 eligible jobs for execution is implementation-defined.

3934 As part of creating the process for the batch job, the batch server shall open the standard output
3935 and standard error streams of the session.

3936 The attributes of a batch job may indicate that the batch server executing the job shall send mail
3937 to a list of users at the time it begins execution of the job.

3938 3.1.7 Batch Job Exit

3939 When the session leader of an executing job terminates, the job exits. As part of exiting a batch
3940 job, the batch server that manages the job shall remove the job from the batch queue in which it
3941 resides. The server shall transfer output files of the job to a location described by the attributes of
3942 the job.

3943 The attributes of a batch job may indicate that the batch server managing the job shall send mail
3944 to a list of users at the time the job exits.

3945 3.1.8 Batch Job Abort

3946 A batch server shall abort jobs for which a required deferred service cannot be performed. The
3947 attributes of a batch job may indicate that the batch server that aborts the job shall send mail to a
3948 list of users at the time it aborts the job.

3949 3.1.9 Batch Authorization

3950 Clients, such as the batch environment utilities (marked BE), access batch services by means of
3951 requests to one or more batch servers. To acquire the services of any given batch server, the user
3952 identifier under which the client runs must be authorized to use that batch server.

3953 The user with an associated user name that creates a batch job shall own the job and can perform
3954 actions such as read, modify, delete, and move.

3955 A user identifier of the same value at a different host need not be the same user. For example,
3956 user name *smith* at host **alpha** may or may not represent the same person as user name *smith* at
3957 host **beta**. Likewise, the same person may have access to different user names on different hosts.

3958 An implementation may optionally provide an authorization mechanism that permits one user
3959 name to access jobs under another user name.

3960 A process on a client host may be authorized to run processes under multiple user names at a
3961 batch server host. Where appropriate, the utilities defined in this volume of IEEE Std 1003.1-2001

3962 provide a means for a user to choose from among such user names when creating or modifying a
3963 batch job.

3964 **3.1.10 Batch Administration**

3965 The processing of a batch job by a batch server is affected by the attributes of the job. The
3966 processing of a batch job may also be affected by the attributes of the batch queue in which the
3967 job resides and by the status of the batch server that manages the job. See also the Base
3968 Definitions volume of IEEE Std 1003.1-2001, Chapter 3, Definitions for batch definitions.

3969 **3.1.11 Batch Notification**

3970 Whereas batch servers are persistent entities, clients are often transient. For example, the *qsub*
3971 utility creates a batch job and exits. For this reason, batch servers notify users of batch job events
3972 by sending mail to the user that owns the job, or to other designated users.

3973 **3.2 Batch Services**

3974 The presence of Batch Environment Services and Utilities option services is indicated by the 1
3975 configuration variable POSIX2_PBS. A conforming batch server provides services as defined in 1
3976 this section. 1

3977 A batch server shall provide batch services in two ways:

- 3978 1. The batch server provides a service at the request of a client.
3979 2. The batch server provides a deferred service as a result of a change in conditions
3980 monitored by the batch server.

3981 If a batch server cannot complete a request, it shall reject the request. If a batch server cannot
3982 complete a deferred service for a batch job, the batch server shall abort the batch job. Table 3-2
3983 (on page 105) is a summary of environment variables that shall be supported by an
3984 implementation of the batch server and utilities.

3985

Table 3-2 Environment Variable Summary

Variable	Description
<i>PBS_DPREFIX</i>	Defines the directive prefix (see <i>qsub</i>)
<i>PBS_ENVIRONMENT</i>	Batch Job is batch or interactive (see Section 3.2.2.1)
<i>PBS_JOBID</i>	The <i>job_identifier</i> attribute of job (see Section 3.2.3.8)
<i>PBS_JOBNAME</i>	The <i>job_name</i> attribute of job (see Section 3.2.3.8)
<i>PBS_O_HOME</i>	Defines the <i>HOME</i> of the batch client (see <i>qsub</i>)
<i>PBS_O_HOST</i>	Defines the host name of the batch client (see <i>qsub</i>)
<i>PBS_O_LANG</i>	Defines the <i>LANG</i> of the batch client (see <i>qsub</i>)
<i>PBS_O_LOGNAME</i>	Defines the <i>LOGNAME</i> of the batch client (see <i>qsub</i>)
<i>PBS_O_MAIL</i>	Defines the <i>MAIL</i> of the batch client (see <i>qsub</i>)
<i>PBS_O_PATH</i>	Defines the <i>PATH</i> of the batch client (see <i>qsub</i>)
<i>PBS_O_QUEUE</i>	Defines the submit queue of the batch client (see <i>qsub</i>)
<i>PBS_O_SHELL</i>	Defines the <i>SHELL</i> of the batch client (see <i>qsub</i>)
<i>PBS_O_TZ</i>	Defines the <i>TZ</i> of the batch client (see <i>qsub</i>)
<i>PBS_O_WORKDIR</i>	Defines the working directory of the batch client (see <i>qsub</i>)
<i>PBS_QUEUE</i>	Defines the initial execution queue (see Section 3.2.2.1)

3.2.1 Batch Job States

A batch job shall always be in one of the following states: QUEUED, RUNNING, HELD, WAITING, EXITING, or TRANSITING. The state of a batch job determines the types of requests that the batch server that manages the batch job can accept for the batch job. A batch server shall change the state of a batch job either in response to service requests from clients or as a result of deferred services, such as job execution or job routing.

A batch job that is in the QUEUED state resides in a queue but is still pending either execution or routing, depending on the queue type.

A batch server that queues a batch job in a routing queue shall put the batch job in the QUEUED state. A batch server that puts a batch job in an execution queue, but has not yet executed the batch job, shall put the batch job in the QUEUED state. A batch job that resides in an execution queue and is executing is defined to be in the RUNNING state. While a batch job is in the RUNNING state, a session leader is associated with the batch job.

A batch job that resides in an execution queue, but is ineligible to run because of a hold attribute, is defined to be in the HELD state.

A batch job that is not held, but must wait until a future date and time before executing, is defined to be in the WAITING state.

When the session leader associated with a running job exits, the batch job shall be placed in the EXITING state.

A batch job for which the session leader has terminated is defined to be in the EXITING state, and the batch server that manages such a batch job cannot accept job modification requests that affect the batch job. While a batch job is in the EXITING state, the batch server that manages the batch job is staging output files and notifying clients of job completion. Once a batch job has exited, it no longer exists as an object managed by a batch server.

A batch job that is being moved from a routing queue to another queue is defined to be in the TRANSITING state.

4028 When a batch job in a routing queue has been selected to be moved to a new destination, then
4029 the batch job shall be in either the QUEUED state or the TRANSITING state, depending on the
4030 batch server implementation.

4031 Batch jobs with either an *Execution_Time* attribute value set in the future or a *Hold_Types* attribute
4032 of value not equal to NO_HOLD, or both, may be routed or held in the routing queue. The
4033 treatment of jobs with the *Execution_Time* or *Hold_Types* attributes in a routing queue is
4034 implementation-defined.

4035 When a batch job in a routing queue has not been selected to be moved to a new destination and
4036 the batch job has a *Hold_Types* attribute value of other than NO_HOLD, then the job should be in
4037 the HELD state.

4038 **Note:** The effect of a hold upon a batch job in a routing queue is implementation-defined. The
4039 implementation should use the state that matches whether the batch job can route with a hold
4040 or not.

4041 When a batch job in a routing queue has not been selected to be moved to a new destination and
4042 the batch job has:

- 4043 • A *Hold_Types* attribute value of NO_HOLD
- 4044 • An *Execution_Time* attribute in the past

4045 then the batch job shall be in the QUEUED state.

4046 When a batch job in a routing queue has not been selected to be moved to a new destination and
4047 the batch job has:

- 4048 • A *Hold_Types* attribute value of NO_HOLD
- 4049 • An *Execution_Time* attribute in the future

4050 then the batch job may be in the WAITING state.

4051 **Note:** The effect of a future execution time upon a batch job in a routing queue is implementation-
4052 defined. The implementation should use the state that matches whether the batch job can route
4053 with a hold or not.

4054 Table 3-3 (on page 107) describes the next state of a batch job, given the current state of the batch
4055 job and the type of request. Table 3-4 (on page 108) describes the response of a batch server to a
4056 request, given the current state of the batch job and the type of request.

4057 3.2.2 Deferred Batch Services

4058 This section describes the deferred services performed by batch servers: job execution, job
4059 routing, job exit, job abort, and the rerunning of jobs after a restart.

4060 3.2.2.1 Batch Job Execution

4061 To execute a batch job is to create a session leader (a process) that runs the shell program
4062 indicated by the *Shell_Path_List* attribute of the batch job. The script is passed to the program as
4063 its standard input. An implementation may pass the script to the program by other
4064 implementation-defined means. At the time a batch job begins execution, it is defined to enter
4065 the RUNNING state.

4066

Table 3-3 Next State Table

Request Type	Current State						
	X	Q	R	H	W	E	T
Queue Batch Job Request	Q	e	e	e	e	e	e
Modify Batch Job Request	e	Q	R	H	W	e	T
Delete Batch Job Request	e	X	E	X	X	E	X
Batch Job Message Request	e	Q	R	H	W	E	T
Rerun Batch Job Request	e	e	Q	e	e	e	e
Signal Batch Job Request	e	e	R	H	W	e	e
Batch Job Status Request	e	Q	R	H	W	E	T
Batch Queue Status Request	X	Q	R	H	W	E	T
Server Status Request	X	Q	R	H	W	E	T
Select Batch Jobs Request	X	Q	R	H	W	E	T
Move Batch Job Request	e	Q	R	H	W	e	T
Hold Batch Job Request	e	H	R/H	H	H	e	T
Release Batch Job Request	e	Q	R	Q/W/H	W	e	T
Server Shutdown Request	X	Q	Q		W	E	T
Locate Batch Job Request	e	Q	R	H	W	E	T

4084 **Legend**

4085 X Nonexistent

4086 Q QUEUED

4087 R RUNNING

4088 H HELD

4089 W WAITING

4090 E EXITING

4091 T TRANSITING

4092 e Error

4093 A batch server that has an execution queue containing jobs is said to own the queue and manage
 4094 the batch jobs in that queue. A batch server that has been started shall execute the batch jobs in
 4095 the execution queues owned by the batch server. The batch server shall schedule for execution
 4096 those jobs in the execution queues that are in the QUEUED state. The algorithm for scheduling
 4097 jobs is implementation-defined.

4098 A batch server that executes a batch job shall create, in the environment of the session leader of
 4099 the batch job, an environment variable named *PBS_ENVIRONMENT*, the value of which is the
 4100 string PBS_BATCH encoded in the portable character set.

4101 A batch server that executes a batch job shall create, in the environment of the session leader of
 4102 the batch job, an environment variable named *PBS_QUEUE*, the value of which is the name of
 4103 the execution queue of the batch job encoded in the portable character set.

4104 To rerun a batch job is to requeue a batch job that is currently executing and then kill the session
 4105 leader of the executing job by sending a SIGKILL prior to completion; see Section 3.2.3.11 (on
 4106 page 120). A batch server that reruns a batch job shall append the standard output and standard
 4107 error files of the batch job to the corresponding files of the previous execution, if they exist, with
 4108 appropriate annotation. If either file does not exist, that file shall be created as in normal
 4109 execution.

4110

Table 3-4 Results/Output Table

4111

4112

4113

4114

4115

4116

4117

4118

4119

4120

4121

4122

4123

4124

4125

4126

4127

Request Type	Current State						
	X	Q	R	H	W	E	T
Queue Batch Job Request	O	e	e	e	e	e	e
Modify Batch Job Request	e	O	e	O	O	e	e
Delete Batch Job Request	e	O	O	O	O	e	O
Batch Job Message Request	e	e	O	e	e	e	e
Rerun Batch Job Request	e	e	O	e	e	e	e
Signal Batch Job Request	e	e	O	e	e	e	e
Batch Job Status Request	e	O	O	O	O	O	O
Batch Queue Status Request	O	O	O	O	O	O	O
Server Status Request	O	O	O	O	O	O	O
Select Batch Job Request	e	O	O	O	O	O	O
Move Batch Job Request	e	O	O	O	O	e	e
Hold Batch Job Request	e	O	O	O	O	e	e
Release Batch Job Request	e	O	e	O	O	e	e
Server Shutdown Request	O	O	e	O	O	e	e
Locate Batch Job Request	e	O	O	O	O	O	O

4128

Legend

4129

O OK

4130

e Error message

4131

4132

The execution of a batch job by a batch server shall be controlled by job, queue, and server attributes, as defined in this section.

4133

Account_Name Attribute

4134

4135

Batch accounting is an optional feature of batch servers. If a batch server implements accounting, the statements in this section apply and the configuration variable POSIX2_PBS_ACCOUNTING shall be set to 1.

4137

4138

A batch server that executes a batch job shall charge the account named in the *Account_Name* attribute of the batch job for resources consumed by the batch job.

4139

4140

If the *Account_Name* attribute of the batch job is absent from the batch job attribute list or is altered while the batch job is in execution, the batch server action is implementation-defined.

4141

Checkpoint Attribute

4142

4143

Batch checkpointing is an optional feature of batch servers. If a batch server implements checkpointing, the statements in this section apply and the configuration variable POSIX2_PBS_CHECKPOINT shall be set to 1.

4145

4146

4147

4148

4149

There are two attributes associated with the checkpointing feature: *Checkpoint* and *Minimum_Cpu_Interval*. *Checkpoint* is a batch job attribute, while *Minimum_Cpu_Interval* is a queue attribute. An implementation that does not support checkpointing shall support the *Checkpoint* job attribute to the extent that the batch server shall maintain and pass this attribute to other servers.

4150

4151

4152

The behavior of a batch server that executes a batch job for which the value of the *Checkpoint* attribute is CHECKPOINT_UNSPECIFIED is implementation-defined. A batch server that executes a batch job for which the value of the *Checkpoint* attribute is NO_CHECKPOINT shall

- 4153 not checkpoint the batch job.
- 4154 A batch server that executes a batch job for which the value of the *CHECKPOINT* attribute is
4155 CHECKPOINT_AT_SHUTDOWN shall checkpoint the batch job only when the batch server
4156 accepts a request to shut down during the time when the batch job is in the RUNNING state.
- 4157 A batch server that executes a batch job for which the value of the *CHECKPOINT* attribute is
4158 CHECKPOINT_AT_MIN_CPU_INTERVAL shall checkpoint the batch job at the interval
4159 specified by the *Minimum_Cpu_Interval* attribute of the queue for which the batch job has been
4160 selected. The *Minimum_Cpu_Interval* attribute shall be specified in units of CPU minutes.
- 4161 A batch server that executes a batch job for which the value of the *CHECKPOINT* attribute is an
4162 unsigned integer shall checkpoint the batch job at an interval that is the value of either the
4163 *CHECKPOINT* attribute, or the *Minimum_Cpu_Interval* attribute of the queue for which the batch job
4164 has been selected, whichever is greater. Both intervals shall be in units of CPU minutes. When
4165 the *Minimum_Cpu_Interval* attribute is greater than the *CHECKPOINT* attribute, the batch job shall
4166 write a warning message to the standard error stream of the batch job.
- 4167 **Error_Path Attribute**
- 4168 The *Error_Path* attribute of a running job cannot be changed by a *Modify Batch Job Request*. When
4169 the *Join_Path* attribute of the batch job is set to the value FALSE and the *Keep_Files* attribute of
4170 the batch job does not contain the value KEEP_STD_ERROR, a batch server that executes a batch
4171 job shall perform one of the following actions:
- 4172 • Set the standard error stream of the session leader of the batch job to the path described by
4173 the value of the *Error_Path* attribute of the batch job.
 - 4174 • Buffer the standard error of the session leader of the batch job until completion of the batch
4175 job, and when the batch job exits return the contents to the destination described by the value
4176 of the *Error_Path* attribute of the batch job.
- 4177 Applications shall not rely on having access to the standard error of a batch job prior to the
4178 completion of the batch job.
- 4179 When the *Error_Path* attribute does not specify a host name, then the batch server shall retain the
4180 standard error of the batch job on the host of execution.
- 4181 When the *Error_Path* attribute does specify a host name and the *Keep_Files* attribute does not
4182 contain the value KEEP_STD_ERROR, then the final destination of the standard error of the
4183 batch job shall be on the host whose host name is specified.
- 4184 If the path indicated by the value of the *Error_Path* attribute of the batch job is a relative path, the
4185 batch server shall expand the path relative to the home directory of the user on the host to which
4186 the file is being returned.
- 4187 When the batch server buffers the standard error of the batch job and the file cannot be opened
4188 for write upon completion of the batch job, then the server shall place the standard error in an
4189 implementation-defined location and notify the user of the location via mail. It shall be possible
4190 for the user to process this mail using the *mailx* utility.
- 4191 If a batch server that does not buffer the standard error cannot open the standard error path of
4192 the batch job for write access, then the batch server shall abort the batch job.

4193 **Execution_Time Attribute**

4194 A batch server shall not execute a batch job before the time represented by the value of the
4195 *Execution_Time* attribute of the batch job. The *Execution_Time* attribute is defined in seconds since
4196 the Epoch.

4197 **Hold_Types Attribute**

4198 A batch server shall support the following hold types:

- 4199 s Can be set or released by a user with at least a privilege level of batch administrator
4200 (SYSTEM).
- 4201 o Can be set or released by a user with at least a privilege level of batch operator
4202 (OPERATOR).
- 4203 u Can be set or released by the user with at least a privilege level of user, where the user is
4204 defined in the *Job_Owner* attribute (USER).
- 4205 n Indicates that none of the *Hold_Types* attributes are set (NO_HOLD).

4206 An implementation may define other hold types. Any additional hold types, how they are
4207 specified, their internal representation, their behavior, and how they affect the behavior of other
4208 utilities are implementation-defined.

4209 The value of the *Hold_Types* attribute shall be the union of the valid hold types ('s', 'o', 'u',
4210 and any implementation-defined hold types), or 'n'.

4211 A batch server shall not execute a batch job if the *Hold_Types* attribute of the batch job has a
4212 value other than NO_HOLD. If the *Hold_Types* attribute of the batch job has a value other than
4213 NO_HOLD, the batch job shall be in the HELD state.

4214 **Job_Owner Attribute**

4215 The *Job_Owner* attribute consists of a pair of user name and host name values of the form:

4216 username@hostname

4217 A batch server that accepts a *Queue Batch Job Request* shall set the *Job_Owner* attribute to a string
4218 that is the *username@hostname* of the user who submitted the job.

4219 **Join_Path Attribute**

4220 A batch server that executes a batch job for which the value of the *Join_Path* attribute is TRUE
4221 shall ignore the value of the *Error_Path* attribute and merge the standard error of the batch job
4222 with the standard output of the batch job.

4223 **Keep_Files Attribute**

4224 A batch server that executes a batch job for which the value of the *Keep_Files* attribute includes
4225 the value KEEP_STD_OUTPUT shall retain the standard output of the batch job on the host
4226 where execution occurs. The standard output shall be retained in the home directory of the user
4227 under whose user ID the batch job is executed and the filename shall be the default filename for
4228 the standard output as defined under the -o option of the *qsub* utility. The *Output_Path* attribute
4229 is not modified.

4230 A batch server that executes a batch job for which the value of the *Keep_Files* attribute includes
4231 the value KEEP_STD_ERROR shall retain the standard error of the batch job on the host where
4232 execution occurs. The standard error shall be retained in the home directory of the user under
4233 whose user ID the batch job is executed and the filename shall be the default filename for

4234 standard error as defined under the `-e` option of the *qsub* utility. The *Error_Path* attribute is not
4235 modified.

4236 A batch server that executes a batch job for which the value of the *Keep_Files* attribute includes
4237 values other than *KEEP_STD_OUTPUT* and *KEEP_STD_ERROR* shall retain these other files on
4238 the host where execution occurs. These files (with implementation-defined names) shall be
4239 retained in the home directory of the user under whose user identifier the batch job is executed.

4240 **Mail_Points and Mail_Users Attributes**

4241 A batch server that executes a batch job for which one of the values of the *Mail_Points* attribute is
4242 the value *MAIL_AT_BEGINNING* shall send a mail message to each user account listed in the
4243 *Mail_Users* attribute of the batch job.

4244 The mail message shall contain at least the batch job identifier, queue, and server at which the
4245 batch job currently resides, and the *Job_Owner* attribute.

4246 **Output_Path Attribute**

4247 The *Output_Path* attribute of a running job cannot be changed by a *Modify Batch Job Request*.
4248 When the *Keep_Files* attribute of the batch job does not contain the value *KEEP_STD_OUTPUT*, a
4249 batch server that executes a batch job shall either:

- 4250 • Set the standard output stream of the session leader of the batch job to the destination
4251 described by the value of the *Output_Path* attribute of the batch job.

4252 or:

4253 • Buffer the standard output of the session leader of the batch job until completion of the batch
4254 job, and when the batch job exits return the contents to the destination described by the value
4255 of the *Output_Path* attribute of the batch job.

4256 When the *Output_Path* attribute does not specify a host name, then the batch server shall retain
4257 the standard output of the batch job on the host of execution.

4258 When the *Keep_Files* attribute does not contain the value *KEEP_STD_OUTPUT* and the
4259 *Output_Path* attribute does specify a host name, then the final destination of the standard output
4260 of the batch job shall be on the host specified.

4261 If the path specified in the *Output_Path* attribute of the batch job is a relative path, the batch
4262 server shall expand the path relative to the home directory of the user on the host to which the
4263 file is being returned.

4264 Whether or not the batch server buffers the standard output of the batch job until completion of
4265 the batch job is implementation-defined. Applications shall not rely on having access to the
4266 standard output of a batch job prior to the completion of the batch job.

4267 When the batch server does buffer the standard output of the batch job and the file cannot be
4268 opened for write upon completion of the batch job, then the batch server shall place the standard
4269 output in an implementation-defined location and notify the user of the location via mail. It shall
4270 be possible for the user to process this mail using the *mailx* utility.

4271 If a batch server that does not buffer the standard output cannot open the standard output path
4272 of the batch job for write access, then the batch server shall abort the batch job.

4273	Priority Attribute
4274	A batch server implementation may choose to preferentially execute a batch job based on the <i>Priority</i> attribute. The interpretation of the batch job <i>Priority</i> attribute by a batch server is implementation-defined. If an implementation uses the <i>Priority</i> attribute, it shall interpret larger values of the <i>Priority</i> attribute to mean the batch job shall be preferentially selected for execution.
4275	
4276	
4277	
4278	Rerunable Attribute
4279	A batch job that began execution but did not complete, because the batch server either shut down or terminated abnormally, shall be requeued if the <i>Rerunable</i> attribute of the batch job has the value TRUE.
4280	
4281	
4282	If a batch job, which was requeued after beginning execution but prior to completion, has a valid checkpoint file and the batch server supports checkpointing, then the batch job shall be restarted from the last valid checkpoint.
4283	
4284	
4285	If the batch job cannot be restarted from a checkpoint, then when a batch job has a <i>Rerunable</i> attribute value of TRUE and was requeued after beginning execution but prior to completion, the batch server shall place the batch job into execution at the beginning of the job.
4286	
4287	
4288	When a batch job has a <i>Rerunable</i> attribute value other than TRUE and was requeued after beginning execution but prior to completion, and the batch job cannot be restarted from a checkpoint, then the batch server shall abort the batch job.
4289	
4290	
4291	Resource_List Attribute
4292	A batch server that executes a batch job shall establish the resource limits of the session leader of the batch job according to the values of the <i>Resource_List</i> attribute of the batch job. Resource limits shall be enforced by an implementation-defined method.
4293	
4294	
4295	Shell_Path_List Attribute
4296	The <i>Shell_Path_List</i> job attribute consists of a list of pairs of pathname and host name values. The host name component can be omitted, in which case the pathname serves as the default pathname when a batch server cannot find the name of the host on which it is running in the list.
4297	
4298	
4299	A batch server that executes a batch job shall select, from the value of the <i>Shell_Path_List</i> attribute of the batch job, a pathname where the shell to execute the batch job shall be found. The batch server shall select the pathname, in order of preference, according to the following methods:
4300	
4301	
4302	
4303	<ul style="list-style-type: none">• Select the pathname that contains the name of the host on which the batch server is running.
4304	<ul style="list-style-type: none">• Select the pathname for which the host name has been omitted.
4305	<ul style="list-style-type: none">• Select the pathname for the login shell of the user under which the batch job is to execute.
4306	If the shell path value selected is an invalid pathname, the batch server shall abort the batch job.
4307	
4308	
4309	
4310	If the value of the selected pathname from the <i>Shell_Path_List</i> attribute of the batch job represents a partial path, the batch server shall expand the path relative to a path that is implementation-defined.
4311	
4312	The batch server that executes the batch job shall execute the program that was selected from the <i>Shell_Path_List</i> attribute of the batch job. The batch server shall pass the path to the script of the batch job as the first argument to the shell program.

4313 **User_List Attribute**

4314 The *User_List* job attribute consists of a list of pairs of user name and host name values. The host
 4315 name component can be omitted, in which case the user name serves as a default when a batch
 4316 server cannot find the name of the host on which it is running in the list.

4317 A batch server that executes a batch job shall select, from the value of the *User_List* attribute of
 4318 the batch job, a user name under which to create the session leader. The server shall select the
 4319 user name, in order of preference, according to the following methods:

- 4320 • Select the user name of a value that contains the name of the host on which the batch server
 4321 executes.
- 4322 • Select the user name of a value for which the host name has been omitted.
- 4323 • Select the user name from the *Job_Owner* attribute of the batch job.

4324 **Variable_List Attribute**

4325 A batch server that executes a batch job shall create, in the environment of the session leader of
 4326 the batch job, each environment variable listed in the *Variable_List* attribute of the batch job, and
 4327 set the value of each such environment variable to that of the corresponding variable in the
 4328 variable list.

4329 **3.2.2.2 Batch Job Routing**

4330 To route a batch job is to select a queue from a list and move the batch job to that queue.

4331 A batch server that has routing queues, which have been started, shall route the jobs in the
 4332 routing queues owned by the batch server. A batch server may delay the routing of a batch job.
 4333 The algorithm for selecting a batch job and the queue to which it will be routed is
 4334 implementation-defined.

4335 When a routing queue has multiple possible destinations specified, then the precedence of the
 4336 destinations is implementation-defined.

4337 A batch server that routes a batch job to a queue at another server shall move the batch job into
 4338 the target queue with a *Queue Batch Job Request*.

4339 If the target server rejects the *Queue Batch Job Request*, the routing server shall retry routing the
 4340 batch job or abort the batch job. A batch server that retries failed routings shall provide a means
 4341 for the batch administrator to specify the number of retries and the minimum period of time
 4342 between retries. The means by which an administrator specifies the number of retries and the
 4343 delay between retries is implementation-defined. When the number of retries specified by the
 4344 batch administrator has been exhausted, the batch server shall abort the batch job and perform
 4345 the functions of *Batch Job Exit*; see Section 3.2.2.3.

4346 **3.2.2.3 Batch Job Exit**

4347 For each job in the EXITING state, the batch server that exited the batch job shall perform the
 4348 following deferred services in the order specified:

- 4349 1. If buffering standard error, move that file into the location specified by the *Error_Path*
 4350 attribute of the batch job.
- 4351 2. If buffering standard output, move that file into the location specified by the *Output_Path*
 4352 attribute of the batch job.
- 4353 3. If the *Mail_Points* attribute of the batch job includes MAIL_AT_EXIT, send mail to the users
 4354 listed in the *Mail_Users* attribute of the batch job. The mail message shall contain at least

4355 the batch job identifier, queue, and server at which the batch job currently resides, and the
4356 *Job_Owner* attribute.

4357 4. Remove the batch job from the queue.

4358 If a batch server that buffers the standard error output cannot return the standard error file to
4359 the standard error path at the time the batch job exits, the batch server shall do one of the
4360 following:

- 4361 • Mail the standard error file to the batch job owner.
- 4362 • Save the standard error file and mail the location and name of the file where the standard
4363 error is stored to the batch job owner.
- 4364 • Save the standard error file and notify the user by other implementation-defined means.

4365 If a batch server that buffers the standard output cannot return the standard output file to the
4366 standard output path at the time the batch job exits, the batch server shall do one of the
4367 following:

- 4368 • Mail the standard output file to the batch job owner.
- 4369 • Save the standard output file and mail the location and name of the file where the standard
4370 output is stored to the batch job owner.
- 4371 • Save the standard output file and notify the user by other implementation-defined means.

4372 At the conclusion of job exit processing, the batch job is no longer managed by a batch server.

4373 3.2.2.4 *Batch Server Restart*

4374 A batch server that has been either shutdown or terminated abnormally, and has returned to
4375 operation, is said to have “restarted”.

4376 Upon restarting, a batch server shall requeue those jobs managed by the batch server that were
4377 in the RUNNING state at the time the batch server shut down and for which the *Rerunable*
4378 attribute of the batch job has the value TRUE.

4379 Queues are defined to be non-volatile. A batch server shall store the content of queues that it
4380 controls in such a way that server and system shutdowns do not erase the content of the queues.

4381 3.2.2.5 *Batch Job Abort*

4382 A batch server that cannot perform a deferred service for a batch job shall abort the batch job.

4383 A batch server that aborts a batch job shall perform the following services:

- 4384 • Delete the batch job from the queue in which it resides.
- 4385 • If the *Mail_Points* attribute of the batch job includes the value MAIL_AT_ABORT, send mail
4386 to the users listed in the value of the *Mail_Users* attribute of the job. The mail message shall
4387 contain at least the batch job identifier, queue, and server at which the batch job currently
4388 resides, the *Job_Owner* attribute, and the reason for the abort.
- 4389 • If the batch job was in the RUNNING state, terminate the session leader of the executing job
4390 by sending the session leader a SIGKILL, place the batch job in the EXITING state, and
4391 perform the actions of *Batch Job Exit*.

4392 **3.2.3 Requested Batch Services**

4393 This section describes the services provided by batch servers in response to requests from
 4394 clients. Table 3-5 summarizes the current set of batch service requests and for each gives its type
 4395 (deferred or not) and whether it is an optional function.

4396 **Table 3-5 Batch Services Summary**

4397 Batch Service	4398 Deferred	4399 Optional
<i>Batch Job Execution</i>	Yes	No
<i>Batch Job Routing</i>	Yes	No
<i>Batch Job Exit</i>	Yes	No
<i>Batch Server Restart</i>	Yes	No
<i>Batch Job Abort</i>	Yes	No
<i>Delete Batch Job Request</i>	No	No
<i>Hold Batch Job Request</i>	No	No
<i>Batch Job Message Request</i>	No	Yes
<i>Batch Job Status Request</i>	No	No
<i>Locate Batch Job Request</i>	No	Yes
<i>Modify Batch Job Request</i>	No	No
<i>Move Batch Job Request</i>	No	No
<i>Queue Batch Job Request</i>	No	No
<i>Batch Queue Status Request</i>	No	No
<i>Release Batch Job Request</i>	No	No
<i>Rerun Batch Job Request</i>	No	No
<i>Select Batch Jobs Request</i>	No	No
<i>Server Shutdown Request</i>	No	No
<i>Server Status Request</i>	No	No
<i>Signal Batch Job Request</i>	No	No
<i>Track Batch Job Request</i>	No	Yes

4419 If a request is rejected because the batch client is not authorized to perform the action, the batch
 4420 server shall return the same status as when the batch job does not exist.

4421 **3.2.3.1 Delete Batch Job Request**

4422 A batch job is defined to have been deleted when it has been removed from the queue in which it
 4423 resides and not instantiated in another queue. A client requests that the server that manages a
 4424 batch job delete the batch job. Such a request is called a *Delete Batch Job Request*.

4425 A batch server shall reject a *Delete Batch Job Request* if any of the following statements are true:

- 4426 • The user of the batch client is not authorized to delete the designated job.
- 4427 • The designated job is not managed by the batch server.
- 4428 • The designated job is in a state inconsistent with the delete request.

4429 A batch server may reject a *Delete Batch Job Request* for other implementation-defined reasons.
 4430 The method used to determine whether the user of a client is authorized to perform the
 4431 requested action is implementation-defined.

4432 A batch server requested to delete a batch job shall delete the batch job if the batch job exists and
 4433 is not in the EXITING state.

4434 A batch server that deletes a batch job in the RUNNING state shall send a SIGKILL signal to the
 4435 session leader of the batch job. It is implementation-defined whether additional signals are sent

- 4436 to the session leader of the job prior to sending the SIGKILL signal.
- 4437 A batch server that deletes a batch job in the RUNNING state shall place the batch job in the
4438 EXITING state after it has killed the session leader of the batch job and shall perform the actions
4439 of *Batch Job Exit*.
- 4440 3.2.3.2 *Hold Batch Job Request*
- 4441 A batch client can request that the batch server add one or more holds to a batch job. Such a
4442 request is called a *Hold Batch Job Request*.
- 4443 A batch server shall reject a *Hold Batch Job Request* if any of the following statements are true:
- 4444 • The batch server does not support one or more of the requested holds to be added to the
4445 batch job.
- 4446 • The user of the batch client is not authorized to add one or more of the requested holds to the
4447 batch job.
- 4448 • The batch server does not manage the specified job.
- 4449 • The designated job is in the EXITING state.
- 4450 A batch server may reject a *Hold Batch Job Request* for other implementation-defined reasons. The
4451 method used to determine whether the user of a client is authorized to perform the requested
4452 action is implementation-defined.
- 4453 A batch server that accepts a *Hold Batch Job Request* for a batch job in the RUNNING state shall
4454 place a hold on the batch job. The effects, if any, the hold will have on a batch job in the
4455 RUNNING state are implementation-defined.
- 4456 A batch server that accepts a *Hold Batch Job Request* shall add each type of hold listed in the *Hold*
4457 *Batch Job Request*, that is not already present, to the value of the *Hold_Types* attribute of the batch
4458 job.
- 4459 3.2.3.3 *Batch Job Message Request*
- 4460 *Batch Job Message Request* is an optional feature of batch servers. If an implementation supports
4461 *Batch Job Message Request*, the statements in this section apply and the configuration variable
4462 *POSIX2_PBS_MESSAGE* shall be set to 1.
- 4463 A batch client can request that a batch server write a message into certain output files of a batch
4464 job. Such a request is called a *Batch Job Message Request*.
- 4465 A batch server shall reject a *Batch Job Message Request* if any of the following statements are true:
- 4466 • The batch server does not support sending messages to jobs.
- 4467 • The user of the batch client is not authorized to post a message to the designated job.
- 4468 • The designated job does not exist on the batch server.
- 4469 • The designated job is not in the RUNNING state.
- 4470 A batch server may reject a *Batch Job Message Request* for other implementation-defined reasons.
4471 The method used to determine whether the user of a client is authorized to perform the
4472 requested action is implementation-defined.
- 4473 A batch server that accepts a *Batch Job Message Request* shall write the message sent by the batch
4474 client into the files indicated by the batch client.

4475 3.2.3.4 *Batch Job Status Request*

4476 A batch client can request that a batch server respond with the status and attributes of a batch
4477 job. Such a request is called a *Batch Job Status Request*.

4478 A batch server shall reject a *Batch Job Status Request* if any of the following statements are true:

- 4479 • The user of the batch client is not authorized to query the status of the designated job.
4480 • The designated job is not managed by the batch server.

4481 A batch server may reject a *Batch Job Status Request* for other implementation-defined reasons.
4482 The method used to determine whether the user of a client is authorized to perform the
4483 requested action is implementation-defined.

4484 A batch server that accepts a *Batch Job Status Request* shall return a *Batch Job Status Message* to the
4485 batch client.

4486 A batch server may return other information in response to a *Batch Job Status Request*.

4487 3.2.3.5 *Locate Batch Job Request*

4488 *Locate Batch Job Request* is an optional feature of batch servers. If an implementation supports
4489 *Locate Batch Job Request*, the statements in this section apply and the configuration variable
4490 POSIX2_PBS_LOCATE shall be set to 1.

4491 A batch client can ask a batch server to respond with the location of a batch job that was created
4492 by the batch server. Such a request is called a *Locate Batch Job Request*.

4493 A batch server that accepts a *Locate Batch Job Request* shall return a *Batch Job Location Message* to
4494 the batch client.

4495 A batch server may reject a *Locate Batch Job Request* for a batch job that was not created by that
4496 server.

4497 A batch server may reject a *Locate Batch Job Request* for a batch job that is no longer managed by
4498 that server; that is, for a batch job that is not in a queue owned by that server.

4499 A batch server may reject a *Locate Batch Job Request* for other implementation-defined reasons.

4500 3.2.3.6 *Modify Batch Job Request*

4501 Batch clients modify (alter) the attributes of a batch job by making a request to the server that
4502 manages the batch job. Such a request is called a *Modify Batch Job Request*.

4503 A batch server shall reject a *Modify Batch Job Request* if any of the following statements are true:

- 4504 • The user of the batch client is not authorized to make the requested modification to the batch
4505 job.
4506 • The designated job is not managed by the batch server.
4507 • The requested modification is inconsistent with the state of the batch job.
4508 • An unrecognized resource is requested for a batch job in an execution queue.

4509 A batch server may reject a *Modify Batch Job Request* for other implementation-defined reasons.
4510 The method used to determine whether the user of a client is authorized to perform the
4511 requested action is implementation-defined.

4512 A batch server that accepts a *Modify Batch Job Request* shall modify all the specified attributes of
4513 the batch job. A batch server that rejects a *Modify Batch Job Request* shall modify none of the
4514 attributes of the batch job.

4515 If the servicing by a batch server of an otherwise valid request would result in no change, then
4516 the batch server shall indicate successful completion of the request.

4517 **3.2.3.7 Move Batch Job Request**

4518 A batch client can request that a batch server move a batch job to another destination. Such a
4519 request is called a *Move Batch Job Request*.

4520 A batch server shall reject a *Move Batch Job Request* if any of the following statements are true:

- 4521 • The user of the batch client is not authorized to remove the designated job from the queue in
4522 which the batch job resides.
- 4523 • The user of the batch client is not authorized to move the designated job to the destination.
- 4524 • The designated job is not managed by the batch server.
- 4525 • The designated job is in the EXITING state.
- 4526 • The destination is inaccessible.

4527 A batch server can reject a *Move Batch Job Request* for other implementation-defined reasons. The
4528 method used to determine whether the user of a client is authorized to perform the requested
4529 action is implementation-defined.

4530 A batch server that accepts a *Move Batch Job Request* shall perform the following services:

- 4531 • Queue the designated job at the destination.
- 4532 • Remove the designated job from the queue in which the batch job resides.

4533 If the destination resides on another batch server, the batch server shall queue the batch job at
4534 the destination by sending a *Queue Batch Job Request* to the other server. If the *Queue Batch Job*
4535 *Request* fails, the batch server shall reject the *Move Batch Job Request*. If the *Queue Batch Job Request*
4536 succeeds, the batch server shall remove the batch job from its queue.

4537 The batch server shall not modify any attributes of the batch job.

4538 **3.2.3.8 Queue Batch Job Request**

4539 A batch queue is controlled by one and only one batch server. A batch server is said to own the
4540 queues that it controls. Batch clients make requests of batch servers to have jobs queued. Such a
4541 request is called a *Queue Batch Job Request*.

4542 A batch server requested to queue a batch job for which the queue is not specified shall select an
4543 implementation-defined queue for the batch job. Such a queue is called the “default queue” of
4544 the batch server. The implementation shall provide the means for a batch administrator to
4545 specify the default queue. The queue, whether specified or defaulted, is called the “target
4546 queue”.

4547 A batch server shall reject a *Queue Batch Job Request* if any of the following statements are true:

- 4548 • The client is not authorized to create a batch job in the target queue.
- 4549 • The request specifies a queue that does not exist on the batch server.
- 4550 • The target queue is an execution queue and the batch server cannot satisfy a resource
4551 requirement of the batch job.
- 4552 • The target queue is an execution queue and an unrecognized resource is requested.
- 4553 • The target queue is an execution queue, the batch server does not support checkpointing, and
4554 the value of the *Checkpoint* attribute of the batch job is not NO_CHECKPOINT.

- 4555 • The job requires access to a user identifier that the batch client is not authorized to access.
- 4556 A batch server may reject a *Queue Batch Job Request* for other implementation-defined reasons.
- 4557 A batch server that accepts a *Queue Batch Job Request* for a batch job for which the
4558 PBS_O_QUEUE value is missing from the value of the *Variable_List* attribute of the batch job
4559 shall add that variable to the list and set the value to the name of the target queue. Once set, no
4560 server shall change the value of PBS_O_QUEUE, even if the batch job is moved to another
4561 queue.
- 4562 A batch server that accepts a *Queue Batch Job Request* for a batch job for which the PBS_JOBID
4563 value is missing from the value of the *Variable_List* attribute shall add that variable to the list and
4564 set the value to the batch job identifier assigned by the server in the format:
- 4565 sequence_number.server
- 4566 A batch server that accepts a *Queue Batch Job Request* for a batch job for which the
4567 PBS_JOBNAME value is missing from the value of the *Variable_List* attribute of the batch job
4568 shall add that variable to the list and set the value to the *Job_Name* attribute of the batch job.
- 4569 **3.2.3.9 Batch Queue Status Request**
- 4570 A batch client can request that a batch server respond with the status and attributes of a queue.
4571 Such a request is called a *Batch Queue Status Request*.
- 4572 A batch server shall reject a *Batch Queue Status Request* if any of the following statements are true:
- 4573 • The user of the batch client is not authorized to query the status of the designated queue.
- 4574 • The designated queue does not exist on the batch server.
- 4575 A batch server may reject a *Batch Queue Status Request* for other implementation-defined reasons.
4576 The method used to determine whether the user of a client is authorized to perform the
4577 requested action is implementation-defined.
- 4578 A batch server that accepts a *Batch Queue Status Request* shall return a *Batch Queue Status Reply* to
4579 the batch client.
- 4580 **3.2.3.10 Release Batch Job Request**
- 4581 A batch client can request that the server remove one or more holds from a batch job. Such a
4582 request is called a *Release Batch Job Request*.
- 4583 A batch server shall reject a *Release Batch Job Request* if any of the following statements are true:
- 4584 • The user of the batch client is not authorized to remove one or more of the requested holds
4585 from the batch job.
- 4586 • The batch server does not manage the specified job.
- 4587 A batch server may reject a *Release Batch Job Request* for other implementation-defined reasons.
4588 The method used to determine whether the user of a client is authorized to perform the
4589 requested action is implementation-defined.
- 4590 A batch server that accepts a *Release Batch Job Request* shall remove each type of hold listed in the
4591 *Release Batch Job Request*, that is present, from the value of the *Hold_Types* attribute of the batch
4592 job.

4593 3.2.3.11 *Rerun Batch Job Request*

4594 To rerun a batch job is to kill the session leader of the batch job and leave the batch job eligible
4595 for re-execution. A batch client can request that a batch server rerun a batch job. Such a request is
4596 called *Rerun Batch Job Request*.

4597 A batch server shall reject a *Rerun Batch Job Request* if any of the following statements are true:

- 4598 • The user of the batch client is not authorized to rerun the designated job.
- 4599 • The *Rerunable* attribute of the designated job has the value FALSE.
- 4600 • The designated job is not in the RUNNING state.
- 4601 • The batch server does not manage the designated job.

4602 A batch server may reject a *Rerun Batch Job Request* for other implementation-defined reasons.
4603 The method used to determine whether the user of a client is authorized to perform the
4604 requested action is implementation-defined.

4605 A batch server that rejects a *Rerun Batch Job Request* shall in no way modify the execution of the
4606 batch job.

4607 A batch server that accepts a request to rerun a batch job shall perform the following services:

- 4608 • Requeue the batch job in the execution queue in which it was executing.
- 4609 • Send a SIGKILL signal to the process group of the session leader of the batch job.

4610 An implementation may indicate to the batch job owner that the batch job has been rerun.
4611 Whether and how the batch job owner is notified that a batch job is rerun is implementation-
4612 defined.

4613 A batch server that reruns a batch job may send other implementation-defined signals to the
4614 session leader of the batch job prior to sending the SIGKILL signal.

4615 A batch server may preferentially select a rerun job for execution. Whether rerun jobs shall be
4616 selected for execution before other jobs is implementation-defined.

4617 3.2.3.12 *Select Batch Jobs Request*

4618 A batch client can request from a batch server a list of jobs managed by that server that match a
4619 list of selection criteria. Such a request is called a *Select Batch Jobs Request*. All the batch jobs
4620 managed by the batch server that receives the request are candidates for selection.

4621 A batch server that accepts a *Select Batch Jobs Request* shall return a list of zero or more job
4622 identifiers that correspond to jobs that meet the selection criteria.

4623 If the batch client is not authorized to query the status of a batch job, the batch server shall not
4624 select the batch job.

4625 3.2.3.13 *Server Shutdown Request*

4626 A batch server is defined to have shut down when it does not respond to requests from clients
4627 and does not perform deferred services for jobs. A batch client can request that a batch server
4628 shut down. Such a request is called a *Server Shutdown Request*.

4629 A batch server shall reject a *Server Shutdown Request* from a client that is not authorized to shut
4630 down the batch server. The method used to determine whether the user of a client is authorized
4631 to perform the requested action is implementation-defined.

4632 A batch server may reject a *Server Shutdown Request* for other implementation-defined reasons.
4633 The reasons for which a *Server Shutdown Request* may be rejected are implementation-defined.

4634 At server shutdown, a batch server shall do, in order of preference, one of the following:

- 4635 • If checkpointing is implemented and the batch job is checkpointable, then checkpoint the
4636 batch job and requeue it.
- 4637 • If the batch job is rerunnable, then requeue the batch job to be rerun (restarted from the
4638 beginning).
- 4639 • Abort the batch job.

4640 **3.2.3.14 Server Status Request**

4641 A batch client can request that a batch server respond with the status and attributes of the batch
4642 server. Such a request is called a *Server Status Request*.

4643 A batch server shall reject a *Server Status Request* if the following statement is true:

- 4644 • The user of the batch client is not authorized to query the status of the designated server.

4645 A batch server may reject a *Server Status Request* for other implementation-defined reasons. The
4646 method used to determine whether the user of a client is authorized to perform the requested
4647 action is implementation-defined.

4648 A batch server that accepts a *Server Status Request* shall return a *Server Status Reply* to the batch
4649 client.

4650 **3.2.3.15 Signal Batch Job Request**

4651 A batch client can request that a batch server signal the session leader of a batch job. Such a
4652 request is called a *Signal Batch Job Request*.

4653 A batch server shall reject a *Signal Batch Job Request* if any of the following statements are true:

- 4654 • The user of the batch client is not authorized to signal the batch job.
- 4655 • The job is not in the RUNNING state.
- 4656 • The batch server does not manage the designated job.
- 4657 • The requested signal is not supported by the implementation.

4658 A batch server may reject a *Signal Batch Job Request* for other implementation-defined reasons.
4659 The method used to determine whether the user of a client is authorized to perform the
4660 requested action is implementation-defined.

4661 A batch server that accepts a request to signal a batch job shall send the signal requested by the
4662 batch client to the process group of the session leader of the batch job.

4663 **3.2.3.16 Track Batch Job Request**

4664 *Track Batch Job Request* is an optional feature of batch servers. If an implementation supports
4665 *Track Batch Job Request*, the statements in this section apply and the configuration variable
4666 POSIX2_PBS_TRACK shall be set to 1.

4667 *Track Batch Job Request* provides a method for tracking the current location of a batch job. Clients
4668 may use the tracking information to determine the batch server that should receive a batch
4669 server request.

4670 If *Track Batch Job Request* is supported by a batch server, then when the batch server queues a
4671 batch job as a result of a *Queue Batch Job Request*, and the batch server is not the batch server that
4672 created the batch job, the batch server shall send a *Track Batch Job Request* to the batch server that
4673 created the job.

4674 If *Track Batch Job Request* is supported by a batch server, then the *Track Batch Job Request* may also
4675 be sent to other servers as a backup to the primary server. The method by which backup servers
4676 are specified is implementation-defined.

4677 If *Track Batch Job Request* is supported by a batch server that receives a *Track Batch Job Request*,
4678 then the batch server shall record the current location of the batch job as contained in the
4679 request.

4680 3.3 Common Behavior for Batch Environment Utilities

4681 3.3.1 Batch Job Identifier

4682 A utility shall recognize *job_identifiers* of the format:

4683 [sequence_number] [.server_name] [@server]

4684 where:

4685 sequence_number An integer that, when combined with *server_name*, provides a batch job
4686 identifier that is unique within the batch system.

4687 server_name The name of the batch server to which the batch job was originally submitted.

4688 server The name of the batch server that is currently managing the batch job.

4689 If the application omits the batch *server_name* portion of a batch job identifier, a utility shall use
4690 the name of a default batch server.

4691 If the application omits the batch *server* portion of a batch job identifier, a utility shall use:

- 4692 • The batch server indicated by *server_name*, if present
- 4693 • The name of the default batch server
- 4694 • The name of the batch server that is currently managing the batch job

4695 If only @server is specified, then the status of all jobs owned by the user on the requested server
4696 is listed.

4697 The means by which a utility determines the default batch server is implementation-defined.

4698 If the application presents the batch *server* portion of a batch job identifier to a utility, the utility
4699 shall send the request to the specified server.

4700 A strictly conforming application shall use the syntax described for the job identifier. Whenever
4701 a batch job identifier is specified whose syntax is not recognized by an implementation, then a
4702 message for each error that occurs shall be written to standard error and the utility shall exit
4703 with an exit status greater than zero.

4704 When a batch job identifier is supplied as an argument to a batch utility and the *server_name*
4705 portion of the batch job identifier is omitted, then the utility shall use the name of the default
4706 batch server.

4707 When a batch job identifier is supplied as an argument to a batch utility and the batch *server*
4708 portion of the batch job identifier is omitted, then the utility shall use either:

- 4709 • The name of the default batch server
 4710 or:
 4711 • The name of the batch server that is currently managing the batch job
 4712 When a batch job identifier is supplied as an argument to a batch utility and the batch *server*
 4713 portion of the batch job identifier is specified, then the utility shall send the required *Batch Server*
 4714 *Request* to the specified server.

4715 **3.3.2 Destination**

4716 The utility shall recognize a *destination* of the format:

4717 [*queue*] [*@server*]

4718 where:

4719 *queue* The name of a valid execution or routing queue at the batch server denoted by
 4720 *@server*, defined as a string of up to 15 alphanumeric characters in the portable
 4721 character set (see the Base Definitions volume of IEEE Std 1003.1-2001, Section 6.1,
 4722 Portable Character Set) where the first character is alphabetic.

4723 *server* The name of a batch server, defined as a string of alphanumeric characters in the
 4724 portable character set.

4725 If the application omits the batch *server* portion of a destination, then the utility shall use either:

- 4726 • The name of the default batch server

4727 or:

- 4728 • The name of the batch server that is currently managing the batch job

4729 The means by which a utility determines the default batch server is implementation-defined.

4730 If the application omits the *queue* portion of a destination, then the utility shall use the name of
 4731 the default queue at the batch server chosen. The means by which a batch server determines its
 4732 default queue is implementation-defined. If a destination is specified in the *queue@server* form,
 4733 then the utility shall use the specified queue at the specified server.

4734 A strictly conforming application shall use the syntax described for a destination. Whenever a
 4735 destination is specified whose syntax is not recognized by an implementation, then a message
 4736 shall be written to standard error and the utility shall exit with an exit status greater than zero.

4737 **3.3.3 Multiple Keyword-Value Pairs**

4738 For each option that can have multiple keyword-value pair arguments, the following rules shall
 4739 apply. Examples of options that can have list-oriented option-arguments are **-u value@keyword**
 4740 and **-I keyword=value**.

- 4741 1. If a batch utility is presented with a list-oriented option-argument for which a keyword has
 4742 a corresponding value that begins with a single or double quote, then the utility shall stop
 4743 interpreting the input stream for delimiters until a second single or double quote,
 4744 respectively, is encountered. This feature allows some flexibility for a comma (',') or
 4745 equals sign ('=') to be part of the value string for a particular keyword; for example:

4746 keywd1='val1,val2',keywd2="val3,val4"

4747 **Note:** This may require the user to escape the quotes as in the following command:

- 4748 foo -xkeywd1=\`val1,`val2\`,keywd2=\`val3,`val4\`"
- 4749 2. If a batch server is presented with a list-oriented attribute that has a keyword that was
4750 encountered earlier in the list, then the later entry for that keyword shall replace the earlier
4751 entry.
- 4752 3. If a batch server is presented with a list-oriented attribute that has a keyword without any
4753 corresponding value of the form *keyword*= or @*keyword* and the same keyword was
4754 encountered earlier in the list, then the prior entry for that keyword shall be ignored by the
4755 batch server.
- 4756 4. If a batch utility is expecting a list-oriented option-argument entry of the form
4757 *keyword*=*value*, but is presented with an entry of the form *keyword* without any
4758 corresponding *value*, then the entry shall be treated as though a default value of NULL was
4759 assigned (that is, *keyword*=NULL) for entry parsing purposes. The utility shall include only
4760 the keyword, not the NULL value, in the associated job attribute.
- 4761 5. If a batch utility is expecting a list-oriented option-argument entry of the form
4762 *value*@*keyword*, but is presented with an entry of the form *value* without any corresponding
4763 *keyword*, then the entry shall be treated as though a keyword of NULL was assigned (that
4764 is, *value*@NULL) for entry parsing purposes. The utility shall include only the value, not
4765 the NULL keyword, in the associated job attribute.
- 4766 6. A batch server shall accept a list-oriented attribute that has multiple occurrences of the
4767 same keyword, interpreting the keywords, in order, with the last value encountered taking
4768 precedence over prior instances of the same keyword. This rule allows, but does not
4769 require, a batch utility to preprocess the attribute to remove duplicate keywords.
- 4770 7. If a batch utility is presented with multiple list-oriented option-arguments on the
4771 command line or in script directives, or both, for a single option, then the utility shall
4772 concatenate, in order, any command line keyword and value pairs to the end of any
4773 directive keyword and value pairs separated by a single comma to produce a single string
4774 that is an equivalent, valid option-argument. The resulting string shall be assigned to the
4775 associated attribute of the batch job (after optionally removing duplicate entries as
4776 described in item 6).

Utilities

4778

This chapter contains the definitions of the utilities, as follows:

4779

- Mandatory utilities that are present on every conformant system

4780

- Optional utilities that are present only on systems supporting the associated option; see Section 1.8.1 (on page 9) for information on the options in this volume of IEEE Std 1003.1-2001

4781

4782

4783 NAME

4784 admin — create and administer SCCS files (**DEVELOPMENT**)

4785 SYNOPSIS

```
4786 XSI    admin -i[name][-n][-a login][-d flag][-e login][-f flag][-m mrlist]
4787      [-r rel][-t[name][-y[comment]]] newfile
4788
4789      admin -n[-a login][-d flag][-e login][-f flag][-m mrlist][-t[name]]
4790      [-y[comment]] newfile ...
4791
4792      admin [-a login][-d flag][-m mrlist][-r rel][-t[name]] file ...
4793
4794      admin -h file ...
4795
4796      admin -z file ...
```

4794 DESCRIPTION

4795 The *admin* utility shall create new SCCS files or change parameters of existing ones. If a named
4796 file does not exist, it shall be created, and its parameters shall be initialized according to the
4797 specified options. Parameters not initialized by an option shall be assigned a default value. If a
4798 named file does exist, parameters corresponding to specified options shall be changed, and other
4799 parameters shall be left as is.

4800 All SCCS filenames supplied by the application shall be of the form *s.filename*. New SCCS files
4801 shall be given read-only permission mode. Write permission in the parent directory is required
4802 to create a file. All writing done by *admin* shall be to a temporary *x-file*, named *x.filename* (see *get*)
4803 created with read-only mode if *admin* is creating a new SCCS file, or created with the same mode
4804 as that of the SCCS file if the file already exists. After successful execution of *admin*, the SCCS file
4805 shall be removed (if it exists), and the *x-file* shall be renamed with the name of the SCCS file. This
4806 ensures that changes are made to the SCCS file only if no errors occur.

4807 The *admin* utility shall also use a transient lock file (named *z.filename*), which is used to prevent
4808 simultaneous updates to the SCCS file; see *get*.

4809 OPTIONS

4810 The *admin* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section
4811 12.2, Utility Syntax Guidelines, except that the **-i**, **-t**, and **-y** options have optional option-
4812 arguments. These optional option-arguments shall not be presented as separate arguments. The
4813 following options are supported:

- 4814 **-n** Create a new SCCS file. When **-n** is used without **-i**, the SCCS file shall be created
4815 with control information but without any file data.
- 4816 **-i[name]** Specify the *name* of a file from which the text for a new SCCS file shall be taken.
4817 The text constitutes the first delta of the file (see the **-r** option for the delta
4818 numbering scheme). If the **-i** option is used, but the *name* option-argument is
4819 omitted, the text shall be obtained by reading the standard input. If this option is
4820 omitted, the SCCS file shall be created with control information but without any
4821 file data. The **-i** option implies the **-n** option.
- 4822 **-r SID** Specify the SID of the initial delta to be inserted. This SID shall be a trunk SID; that
4823 is, the branch and sequence numbers shall be zero or missing. The level number is
4824 optional, and defaults to 1.
- 4825 **-t[name]** Specify the *name* of a file from which descriptive text for the SCCS file shall be
4826 taken. In the case of existing SCCS files (neither **-i** nor **-n** is specified):

- 4827 • A **-t** option without a *name* option-argument shall cause the removal of
 4828 descriptive text (if any) currently in the SCCS file.
- 4829 • A **-t** option with a *name* option-argument shall cause the text (if any) in the
 4830 named file to replace the descriptive text (if any) currently in the SCCS file.
- 4831 **-f flag** Specify a *flag*, and, possibly, a value for the *flag*, to be placed in the SCCS file.
 4832 Several **-f** options may be supplied on a single *admin* command line.
 4833 Implementations shall recognize the following flags and associated values:
- 4834 **b** Allow use of the **-b** option on a *get* command to create branch deltas.
- 4835 **cceil** Specify the highest release (that is, ceiling), a number less than or equal to
 4836 9 999, which may be retrieved by a *get* command for editing. The default
 4837 value for an unspecified **c** flag shall be 9 999.
- 4838 **ffloor** Specify the lowest release (that is, floor), a number greater than 0 but less
 4839 than 9 999, which may be retrieved by a *get* command for editing. The
 4840 default value for an unspecified **f** flag shall be 1.
- 4841 **dSID** Specify the default delta number (SID) to be used by a *get* command.
- 4842 **istr** Treat the “No ID keywords” message issued by *get* or *delta* as a fatal
 4843 error. In the absence of this flag, the message is only a warning. The
 4844 message is issued if no SCCS identification keywords (see *get*) are found
 4845 in the text retrieved or stored in the SCCS file. If a value is supplied, the
 4846 application shall ensure that the keywords exactly match the given string;
 4847 however, the string shall contain a keyword, and no embedded
 4848 <newline>s.
- 4849 **j** Allow concurrent *get* commands for editing on the same SID of an SCCS
 4850 file. This allows multiple concurrent updates to the same version of the
 4851 SCCS file.
- 4852 **llist** Specify a *list* of releases to which deltas can no longer be made (that is, *get*
 4853 **-e** against one of these locked releases fails). Conforming applications
 4854 shall use the following syntax to specify a *list*. Implementations may
 4855 accept additional forms as an extension:
- ```
4856 <list> ::= a | <range-list>
 4857 <range-list> ::= <range> | <range-list>, <range>
 4858 <range> ::= <SID>
```
- 4859       The character *a* in the *list* shall be equivalent to specifying all releases for  
 4860           the named SCCS file. The non-terminal *<SID>* in *range* shall be the delta  
 4861           number of an existing delta associated with the SCCS file.
- 4862       **n**       Cause *delta* to create a null delta in each of those releases (if any) being  
 4863           skipped when a delta is made in a new release (for example, in making  
 4864           delta 5.1 after delta 2.7, releases 3 and 4 are skipped). These null deltas  
 4865           shall serve as anchor points so that branch deltas may later be created  
 4866           from them. The absence of this flag shall cause skipped releases to be  
 4867           nonexistent in the SCCS file, preventing branch deltas from being created  
 4868           from them in the future. During the initial creation of an SCCS file, the **n**  
 4869           flag may be ignored; that is, if the **-r** option is used to set the release  
 4870           number of the initial SID to a value greater than 1, null deltas need not be  
 4871           created for the “skipped” releases.

|      |                    |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
|------|--------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 4872 | <b>qtext</b>       | Substitute user-definable <i>text</i> for all occurrences of the %Q% keyword in the SCCS file text retrieved by <i>get</i> .                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| 4873 |                    |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| 4874 | <b>mmod</b>        | Specify the module name of the SCCS file substituted for all occurrences of the %M% keyword in the SCCS file text retrieved by <i>get</i> . If the <b>m</b> flag is not specified, the value assigned shall be the name of the SCCS file with the leading '.' removed.                                                                                                                                                                                                                                                                                                                      |
| 4875 |                    |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| 4876 |                    |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| 4877 |                    |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| 4878 | <b>ttype</b>       | Specify the <i>type</i> of module in the SCCS file substituted for all occurrences of the %Y% keyword in the SCCS file text retrieved by <i>get</i> .                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| 4879 |                    |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| 4880 | <b>vpgm</b>        | Cause <i>delta</i> to prompt for modification request (MR) numbers as the reason for creating a delta. The optional value specifies the name of an MR number validation program. (If this flag is set when creating an SCCS file, the application shall ensure that the <b>m</b> option is also used even if its value is null.)                                                                                                                                                                                                                                                            |
| 4881 |                    |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| 4882 |                    |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| 4883 |                    |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| 4884 |                    |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| 4885 | <b>-d flag</b>     | Remove (delete) the specified <i>flag</i> from an SCCS file. Several <b>-d</b> options may be supplied on a single <i>admin</i> command. See the <b>-f</b> option for allowable <i>flag</i> names. (The <b>llist</b> flag gives a <i>list</i> of releases to be unlocked. See the <b>-f</b> option for further description of the <b>l</b> flag and the syntax of a <i>list</i> .)                                                                                                                                                                                                          |
| 4886 |                    |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| 4887 |                    |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| 4888 |                    |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| 4889 | <b>-a login</b>    | Specify a <i>login</i> name, or numerical group ID, to be added to the list of users who may make deltas (changes) to the SCCS file. A group ID shall be equivalent to specifying all <i>login</i> names common to that group ID. Several <b>-a</b> options may be used on a single <i>admin</i> command line. As many <i>logins</i> , or numerical group IDs, as desired may be on the list simultaneously. If the list of users is empty, then anyone may add deltas. If <i>login</i> or group ID is preceded by a '!', the users so specified shall be denied permission to make deltas. |
| 4890 |                    |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| 4891 |                    |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| 4892 |                    |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| 4893 |                    |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| 4894 |                    |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| 4895 |                    |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| 4896 | <b>-e login</b>    | Specify a <i>login</i> name, or numerical group ID, to be erased from the list of users allowed to make deltas (changes) to the SCCS file. Specifying a group ID is equivalent to specifying all <i>login</i> names common to that group ID. Several <b>-e</b> options may be used on a single <i>admin</i> command line.                                                                                                                                                                                                                                                                   |
| 4897 |                    |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| 4898 |                    |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| 4899 |                    |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| 4900 | <b>-y[comment]</b> | Insert the <i>comment</i> text into the SCCS file as a comment for the initial delta in a manner identical to that of <i>delta</i> . In the POSIX locale, omission of the <b>-y</b> option shall result in a default comment line being inserted in the form:                                                                                                                                                                                                                                                                                                                               |
| 4901 |                    | "date and time created %s %s by %s", <date>, <time>, <login>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| 4902 |                    | where <date> is expressed in the format of the <i>date</i> utility's %y/%m/%d conversion specification, <time> in the format of the <i>date</i> utility's %T conversion specification format, and <login> is the login name of the user creating the file.                                                                                                                                                                                                                                                                                                                                  |
| 4903 |                    |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| 4904 |                    |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| 4905 |                    |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| 4906 |                    |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| 4907 | <b>-m mrlist</b>   | Insert the list of modification request (MR) numbers into the SCCS file as the reason for creating the initial delta in a manner identical to <i>delta</i> . The application shall ensure that the <b>v</b> flag is set and the MR numbers are validated if the <b>v</b> flag has a value (the name of an MR number validation program). A diagnostic message shall be written if the <b>v</b> flag is not set or MR validation fails.                                                                                                                                                      |
| 4908 |                    |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| 4909 |                    |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| 4910 |                    |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| 4911 |                    |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| 4912 | <b>-h</b>          | Check the structure of the SCCS file and compare the newly computed checksum with the checksum that is stored in the SCCS file. If the newly computed checksum does not match the checksum in the SCCS file, a diagnostic message shall be written.                                                                                                                                                                                                                                                                                                                                         |
| 4913 |                    |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| 4914 |                    |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| 4915 |                    |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| 4916 | <b>-z</b>          | Recompute the SCCS file checksum and store it in the first line of the SCCS file (see the <b>-h</b> option above). Note that use of this option on a truly corrupted file may                                                                                                                                                                                                                                                                                                                                                                                                               |
| 4917 |                    |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |

4918 prevent future detection of the corruption.

4919 **OPERANDS**

4920 The following operands shall be supported:

4921 *file* A pathname of an existing SCCS file or a directory. If *file* is a directory, the *admin* utility shall behave as though each file in the directory were specified as a named file, except that non-SCCS files (last component of the pathname does not begin with **s.**) and unreadable files shall be silently ignored.

4925 *newfile* A pathname of an SCCS file to be created.

4926 If exactly one *file* or *newfile* operand appears, and it is '**-**', the standard input shall be read; each line of the standard input shall be taken to be the name of an SCCS file to be processed. Non-SCCS files and unreadable files shall be silently ignored.

4929 **STDIN**

4930 The standard input shall be a text file used only if **-i** is specified without an option-argument or if a *file* or *newfile* operand is specified as '**-**'. If the first character of any standard input line is <SOH> in the POSIX locale, the results are unspecified.

4933 **INPUT FILES**

4934 The existing SCCS files shall be text files of an unspecified format.

4935 The application shall ensure that the file named by the **-i** option's *name* option-argument shall be a text file; if the first character of any line in this file is <SOH> in the POSIX locale, the results are unspecified. If this file contains more than 99 999 lines, the number of lines recorded in the header for this file shall be 99 999 for this delta.

4939 **ENVIRONMENT VARIABLES**

4940 The following environment variables shall affect the execution of *admin*:

4941 *LANG* Provide a default value for the internationalization variables that are unset or null.  
(See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2, Internationalization Variables for the precedence of internationalization variables used to determine the values of locale categories.)

4945 *LC\_ALL* If set to a non-empty string value, override the values of all the other internationalization variables.

4947 *LC\_CTYPE* Determine the locale for the interpretation of sequences of bytes of text data as characters (for example, single-byte as opposed to multi-byte characters in arguments and input files).

4950 *LC\_MESSAGES*

4951 Determine the locale that should be used to affect the format and contents of  
4952 diagnostic messages written to standard error and the contents of the default **-y**  
4953 comment.

4954 *NLSPATH* Determine the location of message catalogs for the processing of *LC\_MESSAGES*.

4955 **ASYNCHRONOUS EVENTS**

4956 Default.

4957 **STDOUT**

4958 Not used.

**4959   STDERR**

4960       The standard error shall be used only for diagnostic messages.

**4961   OUTPUT FILES**

4962       Any SCCS files created shall be text files of an unspecified format. During processing of a *file*, a  
4963       locking *z-file*, as described in *get* (on page 476), may be created and deleted.

**4964   EXTENDED DESCRIPTION**

4965       None.

**4966   EXIT STATUS**

4967       The following exit values shall be returned:

4968           0   Successful completion.

4969           >0   An error occurred.

**4970   CONSEQUENCES OF ERRORS**

4971       Default.

**4972   APPLICATION USAGE**

4973       It is recommended that directories containing SCCS files be writable by the owner only, and that  
4974       SCCS files themselves be read-only. The mode of the directories should allow only the owner to  
4975       modify SCCS files contained in the directories. The mode of the SCCS files prevents any  
4976       modification at all except by SCCS commands.

**4977   EXAMPLES**

4978       None.

**4979   RATIONALE**

4980       None.

**4981   FUTURE DIRECTIONS**

4982       None.

**4983   SEE ALSO**

4984       *delta, get, prs, what*

**4985   CHANGE HISTORY**

4986       First released in Issue 2.

**4987   Issue 6**

4988       The normative text is reworded to avoid use of the term “must” for application requirements.

4989       The normative text is reworded to emphasize the term “shall” for implementation requirements.

4990       The grammar is updated.

4991       The Open Group Base Resolution bwg2001-007 is applied, adding new text to the INPUT FILES  
4992       section warning that the maximum lines recorded in the file is 99 999.

4993       The Open Group Base Resolution bwg2001-009 is applied, amending the description of the **-h**  
4994       option.

**4995 NAME**

4996 alias — define or display aliases

**4997 SYNOPSIS**

4998 UP alias [*alias-name[=string]* ... ]

4999

**5000 DESCRIPTION**

5001 The *alias* utility shall create or redefine alias definitions or write the values of existing alias  
5002 definitions to standard output. An alias definition provides a string value that shall replace a  
5003 command name when it is encountered; see Section 2.3.1 (on page 32).

5004 An alias definition shall affect the current shell execution environment and the execution  
5005 environments of the subshells of the current shell. When used as specified by this volume of  
5006 IEEE Std 1003.1-2001, the alias definition shall not affect the parent process of the current shell  
5007 nor any utility environment invoked by the shell; see Section 2.12 (on page 61).

**5008 OPTIONS**

5009 None.

**5010 OPERANDS**

5011 The following operands shall be supported:

5012 *alias-name* Write the alias definition to standard output.

5013 *alias-name*=*string*

5014 Assign the value of *string* to the alias *alias-name*.

5015 If no operands are given, all alias definitions shall be written to standard output.

**5016 STDIN**

5017 Not used.

**5018 INPUT FILES**

5019 None.

**5020 ENVIRONMENT VARIABLES**

5021 The following environment variables shall affect the execution of *alias*:

5022 *LANG* Provide a default value for the internationalization variables that are unset or null.  
5023 (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2,  
5024 Internationalization Variables for the precedence of internationalization variables  
5025 used to determine the values of locale categories.)

5026 *LC\_ALL* If set to a non-empty string value, override the values of all the other  
5027 internationalization variables.

5028 *LC\_CTYPE* Determine the locale for the interpretation of sequences of bytes of text data as  
5029 characters (for example, single-byte as opposed to multi-byte characters in  
5030 arguments).

**5031 *LC\_MESSAGES***

5032 Determine the locale that should be used to affect the format and contents of  
5033 diagnostic messages written to standard error.

5034 XSI *NLSPATH* Determine the location of message catalogs for the processing of *LC\_MESSAGES*.

5035 **ASYNCHRONOUS EVENTS**

5036 Default.

5037 **STDOUT**

5038 The format for displaying aliases (when no operands or only *name* operands are specified) shall  
5039 be:

5040 "%s=%s\n", *name*, *value*

5041 The *value* string shall be written with appropriate quoting so that it is suitable for reinput to the  
5042 shell. See the description of shell quoting in Section 2.2 (on page 30).

5043 **STDERR**

5044 The standard error shall be used only for diagnostic messages.

5045 **OUTPUT FILES**

5046 None.

5047 **EXTENDED DESCRIPTION**

5048 None.

5049 **EXIT STATUS**

5050 The following exit values shall be returned:

5051 0 Successful completion.

5052 >0 One of the *name* operands specified did not have an alias definition, or an error occurred.

5053 **CONSEQUENCES OF ERRORS**

5054 Default.

5055 **APPLICATION USAGE**

5056 None.

5057 **EXAMPLES**

- 5058 1. Change *ls* to give a columnated, more annotated output:

5059 alias ls="ls -CF"

- 5060 2. Create a simple “redo” command to repeat previous entries in the command history file:

5061 alias r='fc -s'

- 5062 3. Use 1K units for *du*:

5063 alias du=du\ -k

- 5064 4. Set up *nohup* so that it can deal with an argument that is itself an alias name:

5065 alias nohup="nohup "

5066 **RATIONALE**

5067 The *alias* description is based on historical KornShell implementations. Known differences exist  
5068 between that and the C shell. The KornShell version was adopted to be consistent with all the  
5069 other KornShell features in this volume of IEEE Std 1003.1-2001, such as command line editing.

5070 Since *alias* affects the current shell execution environment, it is generally provided as a shell  
5071 regular built-in.

5072 Historical versions of the KornShell have allowed aliases to be exported to scripts that are  
5073 invoked by the same shell. This is triggered by the *alias -x* flag; it is allowed by this volume of  
5074 IEEE Std 1003.1-2001 only when an explicit extension such as *-x* is used. The standard  
5075 developers considered that aliases were of use primarily to interactive users and that they

5076 should normally not affect shell scripts called by those users; functions are available to such  
5077 scripts.

5078 Historical versions of the KornShell had not written aliases in a quoted manner suitable for  
5079 reentry to the shell, but this volume of IEEE Std 1003.1-2001 has made this a requirement for all  
5080 similar output. Therefore, consistency with this volume of IEEE Std 1003.1-2001 was chosen over  
5081 this detail of historical practice.

5082 **FUTURE DIRECTIONS**

5083 None.

5084 **SEE ALSO**

5085 Section 2.9.5 (on page 54)

5086 **CHANGE HISTORY**

5087 First released in Issue 4.

5088 **Issue 6**

5089 This utility is marked as part of the User Portability Utilities option.

5090 The APPLICATION USAGE section is added.

## 5091 NAME

5092 ar — create and maintain library archives

## 5093 SYNOPSIS

5094 SD ar -d[-v] archive file ...

5095

5096 XSI ar -m [-v] archive file ... 1

5097 ar -m -a[-v] posname archive file ... 1

5098 ar -m -b[-v] posname archive file ... 1

5099 ar -m -i[-v] posname archive file ... 1

5100

5101 XSI ar -p[-v] [-s]archive [file ...]

5102 XSI ar -q[-cv] archive file ...

5103

5104 ar -r[-cuv] archive file ... 1

5105 XSI ar -r -a[-cuv] posname archive file ... 1

5106 ar -r -b[-cuv] posname archive file ... 1

5107 ar -r -i[-cuv] posname archive file ... 1

5108

5109 XSI ar -t[-v] [-s]archive [file ...]

5110 XSI ar -x[-v] [-sCT]archive [file ...]

## 5111 DESCRIPTION

5112 The ar utility is part of the Software Development Utilities option.

5113 The ar utility can be used to create and maintain groups of files combined into an archive. Once  
5114 an archive has been created, new files can be added, and existing files in an archive can be  
5115 extracted, deleted, or replaced. When an archive consists entirely of valid object files, the  
5116 implementation shall format the archive so that it is usable as a library for link editing (see c99  
5117 and fort77). When some of the archived files are not valid object files, the suitability of the  
5118 XSI archive for library use is undefined. If an archive consists entirely of printable files, the entire  
5119 archive shall be printable.

5120 When ar creates an archive, it creates administrative information indicating whether a symbol  
5121 table is present in the archive. When there is at least one object file that ar recognizes as such in  
5122 the archive, an archive symbol table shall be created in the archive and maintained by ar; it is  
5123 used by the link editor to search the archive. Whenever the ar utility is used to create or update  
5124 the contents of such an archive, the symbol table shall be rebuilt. The -s option shall force the  
5125 symbol table to be rebuilt.

5126 All file operands can be pathnames. However, files within archives shall be named by a filename,  
5127 which is the last component of the pathname used when the file was entered into the archive.  
5128 The comparison of file operands to the names of files in archives shall be performed by  
5129 comparing the last component of the operand to the name of the file in the archive.

5130 It is unspecified whether multiple files in the archive may be identically named. In the case of  
5131 XSI such files, however, each file and posname operand shall match only the first file in the archive  
5132 having a name that is the same as the last component of the operand.

5133 **OPTIONS**

- 5134      The **ar** utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section 12.2,  
5135      Utility Syntax Guidelines.
- 5136      The following options shall be supported:
- 5137    XSI    **-a**      Position new files in the archive after the file named by the *posname* operand.
- 5138    XSI    **-b**      Position new files in the archive before the file named by the *posname* operand.
- 5139    **-c**      Suppress the diagnostic message that is written to standard error by default when  
5140      the archive *archive* is created.
- 5141    XSI    **-C**      Prevent extracted files from replacing like-named files in the file system. This  
5142      option is useful when **-T** is also used, to prevent truncated filenames from  
5143      replacing files with the same prefix.
- 5144      **-d**      Delete one or more *files* from *archive*.
- 5145    XSI    **-i**      Position new files in the archive before the file in the archive named by the *posname*  
5146      operand (equivalent to **-b**).
- 5147    XSI    **-m**      Move the named files in the archive. The **-a**, **-b**, or **-i** options with the *posname*  
5148      operand indicate the position; otherwise, move the names files in the archive to the  
5149      end of the archive.
- 5150      **-p**      Write the contents of the *files* in the archive named by *file* operands from *archive* to  
5151      the standard output. If no *file* operands are specified, the contents of all files in the  
5152      archive shall be written in the order of the archive.
- 5153    XSI    **-q**      Append the named files to the end of the archive. In this case **ar** does not check  
5154      whether the added files are already in the archive. This is useful to bypass the  
5155      searching otherwise done when creating a large archive piece by piece.
- 5156      **-r**      Replace or add *files* to *archive*. If the archive named by *archive* does not exist, a  
5157      new archive shall be created and a diagnostic message shall be written to standard  
5158      error (unless the **-c** option is specified). If no *files* are specified and the *archive*  
5159      exists, the results are undefined. Files that replace existing files in the archive shall  
5160      not change the order of the archive. Files that do not replace existing files in the  
5161      archive shall be appended to the archive unless a **-a**, **-b**, or **-i** option specifies  
5162      another position.
- 5163    XSI    **-s**      Force the regeneration of the archive symbol table even if **ar** is not invoked with an  
5164      option that modifies the archive contents. This option is useful to restore the  
5165      archive symbol table after it has been stripped; see *strip*.
- 5166      **-t**      Write a table of contents of *archive* to the standard output. The files specified by the  
5167      *file* operands shall be included in the written list. If no *file* operands are specified,  
5168      all files in *archive* shall be included in the order of the archive.
- 5169    XSI    **-T**      Allow filename truncation of extracted files whose archive names are longer than  
5170      the file system can support. By default, extracting a file with a name that is too  
5171      long shall be an error; a diagnostic message shall be written and the file shall not  
5172      be extracted.
- 5173      **-u**      Update older files in the archive. When used with the **-r** option, files in the archive  
5174      shall be replaced only if the corresponding *file* has a modification time that is at  
5175      least as new as the modification time of the file in the archive.

- 5176       **-v**       Give verbose output. When used with the option characters **-d**, **-r**, or **-x**, write a  
5177       detailed file-by-file description of the archive creation and maintenance activity, as  
5178       described in the STDOUT section.
- 5179       When used with **-p**, write the name of the file in the archive to the standard output  
5180       before writing the file in the archive itself to the standard output, as described in  
5181       the STDOUT section.
- 5182       When used with **-t**, include a long listing of information about the files in the  
5183       archive, as described in the STDOUT section.
- 5184       **-x**       Extract the files in the archive named by the *file* operands from *archive*. The  
5185       contents of the archive shall not be changed. If no *file* operands are given, all files  
5186       in the archive shall be extracted. The modification time of each file extracted shall  
5187       be set to the time the file is extracted from the archive.

## 5188 OPERANDS

5189       The following operands shall be supported:

- 5190       **archive**     A pathname of the archive.
- 5191       **file**       A pathname. Only the last component shall be used when comparing against the  
5192       names of files in the archive. If two or more *file* operands have the same last  
5193       pathname component (basename), the results are unspecified. The  
5194       implementation's archive format shall not truncate valid filenames of files added  
5195       to or replaced in the archive.
- 5196       **XSI posname**     The name of a file in the archive, used for relative positioning; see options **-m** and  
5197       **-r**.

## 5198 STDIN

5199       Not used.

## 5200 INPUT FILES

5201       The archive named by *archive* shall be a file in the format created by *ar -r*.

## 5202 ENVIRONMENT VARIABLES

5203       The following environment variables shall affect the execution of *ar*:

- 5204       **LANG**       Provide a default value for the internationalization variables that are unset or null.  
5205       (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2,  
5206       Internationalization Variables for the precedence of internationalization variables  
5207       used to determine the values of locale categories.)
- 5208       **LC\_ALL**     If set to a non-empty string value, override the values of all the other  
5209       internationalization variables.
- 5210       **LC\_CTYPE**    Determine the locale for the interpretation of sequences of bytes of text data as  
5211       characters (for example, single-byte as opposed to multi-byte characters in  
5212       arguments and input files).

## 5213 LC\_MESSAGES

5214       Determine the locale that should be used to affect the format and contents of  
5215       diagnostic messages written to standard error.

5216       **LC\_TIME**     Determine the format and content for date and time strings written by *ar -tv*.

5217       **XSI NLSPATH**     Determine the location of message catalogs for the processing of *LC\_MESSAGES*.

5218       **TMPDIR**     Determine the pathname that overrides the default directory for temporary files, if  
5219       any.

5220        **TZ**        Determine the timezone used to calculate date and time strings written by *ar -tv*.  
5221        If *TZ* is unset or null, an unspecified default timezone shall be used.

5222 **ASYNCHRONOUS EVENTS**

5223        Default.

5224 **STDOUT**

5225        If the **-d** option is used with the **-v** option, the standard output format shall be:

5226        "*d* - %s\n", <*file*>

5227        where *file* is the operand specified on the command line.

5228        If the **-p** option is used with the **-v** option, *ar* shall precede the contents of each file with:

5229        "\n<%s>\n\n", <*file*>

5230        where *file* is the operand specified on the command line, if *file* operands were specified, and the  
5231        name of the file in the archive if they were not.

5232        If the **-r** option is used with the **-v** option:

- If *file* is already in the archive, the standard output format shall be:

5234        "r - %s\n", <*file*>

5235        where <*file*> is the operand specified on the command line.

- If *file* is not already in the archive, the standard output format shall be:

5237        "a - %s\n", <*file*>

5238        where <*file*> is the operand specified on the command line.

5239        If the **-t** option is used, *ar* shall write the names of the files in the archive to the standard output  
5240        in the format:

5241        "%s\n", <*file*>

5242        where *file* is the operand specified on the command line, if *file* operands were specified, or the  
5243        name of the file in the archive if they were not.

5244        If the **-t** option is used with the **-v** option, the standard output format shall be:

5245        "%s %u/%u %u %s %d %d:%d %d %s\n", <*member mode*>, <*user ID*>,  
5246        <*group ID*>, <*number of bytes in member*>,  
5247        <*abbreviated month*>, <*day-of-month*>, <*hour*>,  
5248        <*minute*>, <*year*>, <*file*>

5249        where:

5250        <*file*>        Shall be the operand specified on the command line, if *file* operands were specified,  
5251        or the name of the file in the archive if they were not.

5252        <*member mode*>

5253        Shall be formatted the same as the <*file mode*> string defined in the STDOUT  
5254        section of *ls*, except that the first character, the <*entry type*>, is not used; the string  
5255        represents the file mode of the file in the archive at the time it was added to or  
5256        replaced in the archive.

5257        The following represent the last-modification time of a file when it was most recently added to  
5258        or replaced in the archive:

5259       *<abbreviated month>*  
5260           Equivalent to the format of the %b conversion specification format in *date*.  
5261       *<day-of-month>*  
5262           Equivalent to the format of the %e conversion specification format in *date*.  
5263       *<hour>*     Equivalent to the format of the %H conversion specification format in *date*.  
5264       *<minute>*    Equivalent to the format of the %M conversion specification format in *date*.  
5265       *<year>*     Equivalent to the format of the %Y conversion specification format in *date*.  
5266       When *LC\_TIME* does not specify the POSIX locale, a different format and order of presentation  
5267       of these fields relative to each other may be used in a format appropriate in the specified locale.  
5268       If the **-x** option is used with the **-v** option, the standard output format shall be:  
5269       " *x* = %s\n", *<file>*  
5270       where *file* is the operand specified on the command line, if *file* operands were specified, or the  
5271       name of the file in the archive if they were not.

## 5272   **STDERR**

5273       The standard error shall be used only for diagnostic messages. The diagnostic message about  
5274       creating a new archive when **-c** is not specified shall not modify the exit status.

## 5275   **OUTPUT FILES**

5276       Archives are files with unspecified formats.

## 5277   **EXTENDED DESCRIPTION**

5278       None.

## 5279   **EXIT STATUS**

5280       The following exit values shall be returned:

5281       0   Successful completion.  
5282       >0   An error occurred.

## 5283   **CONSEQUENCES OF ERRORS**

5284       Default.

## 5285   **APPLICATION USAGE**

5286       None.

## 5287   **EXAMPLES**

5288       None.

## 5289   **RATIONALE**

5290       The archive format is not described. It is recognized that there are several known *ar* formats,  
5291       which are not compatible. The *ar* utility is included, however, to allow creation of archives that  
5292       are intended for use only on one machine. The archive is specified as a file, and it can be moved  
5293       as a file. This does allow an archive to be moved from one machine to another machine that uses  
5294       the same implementation of *ar*.

5295       Utilities such as *pax* (and its forebears *tar* and *cpio*) also provide portable “archives”. This is a not  
5296       a duplication; the *ar* utility is included to provide an interface primarily for *make* and the  
5297       compilers, based on a historical model.

5298       In historical implementations, the **-q** option (available on XSI-conforming systems) is known to  
5299       execute quickly because *ar* does not check on whether the added members are already in the  
5300       archive. This is useful to bypass the searching otherwise done when creating a large archive

5301 piece-by-piece. These remarks may but need not remain true for a brand new implementation of  
5302 this utility; hence, these remarks have been moved into the RATIONALE.

5303 BSD implementations historically required applications to provide the **-s** option whenever the  
5304 archive was supposed to contain a symbol table. As in this volume of IEEE Std 1003.1-2001,  
5305 System V historically creates or updates an archive symbol table whenever an object file is  
5306 removed from, added to, or updated in the archive.

5307 The OPERANDS section requires what might seem to be true without specifying it: the archive  
5308 cannot truncate the filenames below {NAME\_MAX}. Some historical implementations do so,  
5309 however, causing unexpected results for the application. Therefore, this volume of  
5310 IEEE Std 1003.1-2001 makes the requirement explicit to avoid misunderstandings.

5311 According to the System V documentation, the options **-dmpqrtx** are not required to begin with  
5312 a hyphen ('-'). This volume of IEEE Std 1003.1-2001 requires that a conforming application use  
5313 the leading hyphen.

5314 The archive format used by the 4.4 BSD implementation is documented in this RATIONALE as  
5315 an example:

5316 A file created by *ar* begins with the “magic” string " !<arch>\n". The rest of the archive is  
5317 made up of objects, each of which is composed of a header for a file, a possible filename, and  
5318 the file contents. The header is portable between machine architectures, and, if the file  
5319 contents are printable, the archive is itself printable.

5320 The header is made up of six ASCII fields, followed by a two-character trailer. The fields are  
5321 the object name (16 characters), the file last modification time (12 characters), the user and  
5322 group IDs (each 6 characters), the file mode (8 characters), and the file size (10 characters). All  
5323 numeric fields are in decimal, except for the file mode, which is in octal.

5324 The modification time is the file *st\_mtime* field. The user and group IDs are the file *st\_uid* and  
5325 *st\_gid* fields. The file mode is the file *st\_mode* field. The file size is the file *st\_size* field. The  
5326 two-byte trailer is the string " `<newline>". 1

5327 Only the name field has any provision for overflow. If any filename is more than 16  
5328 characters in length or contains an embedded space, the string "#1/" followed by the ASCII  
5329 length of the name is written in the name field. The file size (stored in the archive header) is  
5330 incremented by the length of the name. The name is then written immediately following the  
5331 archive header.

5332 Any unused characters in any of these fields are written as <space>s. If any fields are their  
5333 particular maximum number of characters in length, there is no separation between the  
5334 fields.

5335 Objects in the archive are always an even number of bytes long; files that are an odd number  
5336 of bytes long are padded with a <newline>, although the size in the header does not reflect  
5337 this.

5338 The *ar* utility description requires that (when all its members are valid object files) *ar* produce an  
5339 object code library, which the linkage editor can use to extract object modules. If the linkage  
5340 editor needs a symbol table to permit random access to the archive, *ar* must provide it; however,  
5341 *ar* does not require a symbol table.

5342 The BSD **-o** option was omitted. It is a rare conforming application that uses *ar* to extract object  
5343 code from a library with concern for its modification time, since this can only be of importance  
5344 to *make*. Hence, since this functionality is not deemed important for applications portability, the  
5345 modification time of the extracted files is set to the current time.

5346 There is at least one known implementation (for a small computer) that can accommodate only  
5347 object files for that system, disallowing mixed object and other files. The ability to handle any  
5348 type of file is not only historical practice for most implementations, but is also a reasonable  
5349 expectation.

5350 Consideration was given to changing the output format of *ar -tv* to the same format as the  
5351 output of *ls -l*. This would have made parsing the output of *ar* the same as that of *ls*. This was  
5352 rejected in part because the current *ar* format is commonly used and changes would break  
5353 historical usage. Second, *ar* gives the user ID and group ID in numeric format separated by a  
5354 slash. Changing this to be the user name and group name would not be correct if the archive  
5355 were moved to a machine that contained a different user database. Since *ar* cannot know  
5356 whether the archive was generated on the same machine, it cannot tell what to report.

5357 The text on the **-ur** option combination is historical practice—since one filename can easily  
5358 represent two different files (for example, **/a/foo** and **/b/foo**), it is reasonable to replace the file in  
5359 the archive even when the modification time in the archive is identical to that in the file system.

## 5360 FUTURE DIRECTIONS

5361 None.

## 5362 SEE ALSO

5363 *c99*, *date*, *fort77*, *pax*, *strip* the Base Definitions volume of IEEE Std 1003.1-2001, Chapter 13,  
5364 Headers, <unistd.h> description of {POSIX\_NO\_TRUNC}

## 5365 CHANGE HISTORY

5366 First released in Issue 2.

### 5367 Issue 5

5368 The FUTURE DIRECTIONS section is added.

### 5369 Issue 6

5370 This utility is marked as part of the Software Development Utilities option.

5371 The STDOUT description is changed for the **-v** option to align with the IEEE P1003.2b draft  
5372 standard.

5373 The normative text is reworded to avoid use of the term “must” for application requirements.

5374 The *TZ* entry is added to the ENVIRONMENT VARIABLES section.

5375 IEEE PASC Interpretation 1003.2 #198 is applied, changing the description to consistently use  
5376 “file” to refer to a file in the file system hierarchy, “archive” to refer to the archive being  
5377 operated upon by the *ar* utility, and “file in the archive” to refer to a copy of a file that is  
5378 contained in the archive.

5379 IEEE Std 1003.1-2001/Cor 1-2002, item XCU/TC1/D6/10 is applied, making corrections to the 1  
5380 SYNOPSIS. The change was needed since the **-a**, **-b**, and **-i** options are mutually-exclusive, and 1  
5381 *posname* is required if any of these options is specified. 1

5382 IEEE Std 1003.1-2001/Cor 1-2002, item XCU/TC1/D6/11 is applied, correcting the description 1  
5383 of the two-byte trailer in RATIONALE which had missed out a backquote. The correct trailer is a 1  
5384 backquote followed by a <newline>. 1

**5385 NAME**

5386        *asa* — interpret carriage-control characters

**5387 SYNOPSIS**

5388 FR        *asa* [ *file* . . . ]

**5390 DESCRIPTION**

5391        The *asa* utility shall write its input files to standard output, mapping carriage-control characters  
5392        from the text files to line-printer control sequences in an implementation-defined manner.

5393        The first character of every line shall be removed from the input, and the following actions are  
5394        performed.

5395        If the character removed is:

5396            <space> The rest of the line is output without change.

5397            0        A <newline> is output, then the rest of the input line.

5398            1        One or more implementation-defined characters that causes an advance to the next  
5399        page shall be output, followed by the rest of the input line.

5400            +        The <newline> of the previous line shall be replaced with one or more  
5401        implementation-defined characters that causes printing to return to column position 1,  
5402        followed by the rest of the input line. If the '+' is the first character in the input, it shall  
5403        be equivalent to <space>.

5404        The action of the *asa* utility is unspecified upon encountering any character other than those  
5405        listed above as the first character in a line.

**5406 OPTIONS**

5407        None.

**5408 OPERANDS**

5409        *file*        A pathname of a text file used for input. If no *file* operands are specified, the  
5410        standard input shall be used.

**5411 STDIN**

5412        The standard input shall be used only if no *file* operands are specified; see the INPUT FILES  
5413        section.

**5414 INPUT FILES**

5415        The input files shall be text files.

**5416 ENVIRONMENT VARIABLES**

5417        The following environment variables shall affect the execution of *asa*:

5418        *LANG*      Provide a default value for the internationalization variables that are unset or null.  
5419        (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2,  
5420        Internationalization Variables for the precedence of internationalization variables  
5421        used to determine the values of locale categories.)

5422        *LC\_ALL*     If set to a non-empty string value, override the values of all the other  
5423        internationalization variables.

5424        *LC\_CTYPE*   Determine the locale for the interpretation of sequences of bytes of text data as  
5425        characters (for example, single-byte as opposed to multi-byte characters in  
5426        arguments and input files).

5427       ***LC\_MESSAGES***  
5428              Determine the locale that should be used to affect the format and contents of  
5429              diagnostic messages written to standard error.

5430 XSI       **NLSPATH**   Determine the location of message catalogs for the processing of *LC\_MESSAGES*.

5431 **ASYNCHRONOUS EVENTS**  
5432              Default.

5433 **STDOUT**  
5434              The standard output shall be the text from the input file modified as described in the  
5435              DESCRIPTION section.

5436 **STDERR**  
5437              None.

5438 **OUTPUT FILES**  
5439              None.

5440 **EXTENDED DESCRIPTION**  
5441              None.

5442 **EXIT STATUS**  
5443              The following exit values shall be returned:  
5444                  0 All input files were output successfully.  
5445                  >0 An error occurred.

5446 **CONSEQUENCES OF ERRORS**  
5447              Default.

5448 **APPLICATION USAGE**  
5449              None.

5450 **EXAMPLES**

5451              1. The following command:  
5452                     `asa file`  
5453                      permits the viewing of *file* (created by a program using FORTRAN-style carriage-control  
5454                      characters) on a terminal.

5455              2. The following command:  
5456                     `a.out | asa | lp`  
5457                      formats the FORTRAN output of **a.out** and directs it to the printer.

5458 **RATIONALE**  
5459              The *asa* utility is needed to map “standard” FORTRAN 77 output into a form acceptable to  
5460              contemporary printers. Usually, *asa* is used to pipe data to the *lp* utility; see *lp*.  
5461              This utility is generally used only by FORTRAN programs. The standard developers decided to  
5462              retain *asa* to avoid breaking the historical large base of FORTRAN applications that put  
5463              carriage-control characters in their output files. There is no requirement that a system have a  
5464              FORTRAN compiler in order to run applications that need *asa*.  
5465              Historical implementations have used an ASCII <form-feed> in response to a 1 and an ASCII  
5466              <carriage-return> in response to a '+'. It is suggested that implementations treat characters  
5467              other than 0, 1, and '+' as <space> in the absence of any compelling reason to do otherwise.  
5468              However, the action is listed here as “unspecified”, permitting an implementation to provide

5469 extensions to access fast multiple-line slewing and channel seeking in a non-portable manner.

5470 **FUTURE DIRECTIONS**

5471 None.

5472 **SEE ALSO**

5473 *fort77, lp*

5474 **CHANGE HISTORY**

5475 First released in Issue 4.

5476 **Issue 6**

5477 This utility is marked as part of the FORTRAN Runtime Utilities option.

5478 The normative text is reworded to avoid use of the term “must” for application requirements.

5479 **NAME**

5480 at — execute commands at a later time

5481 **SYNOPSIS**

5482 UP at [-m][-f file][-q queuename] -t time\_arg

5483 at [-m][-f file][-q queuename] timespec ...

5484 at -r at\_job\_id ...

5485 at -l -q queuename

5486 at -l [at\_job\_id ...]

5487

5488 **DESCRIPTION**

5489 The *at* utility shall read commands from standard input and group them together as an *at-job*, to  
5490 be executed at a later time.

5491 The *at-job* shall be executed in a separate invocation of the shell, running in a separate process  
5492 group with no controlling terminal, except that the environment variables, current working  
5493 directory, file creation mask, and other implementation-defined execution-time attributes in  
5494 effect when the *at* utility is executed shall be retained and used when the *at-job* is executed.

5495 When the *at-job* is submitted, the *at\_job\_id* and scheduled time shall be written to standard error.  
5496 The *at\_job\_id* is an identifier that shall be a string consisting solely of alphanumeric characters  
5497 and the period character. The *at\_job\_id* shall be assigned by the system when the job is scheduled  
5498 such that it uniquely identifies a particular job.

5499 User notification and the processing of the job's standard output and standard error are  
5500 described under the **-m** option.

5501 XSI Users shall be permitted to use *at* if their name appears in the file **/usr/lib/cron/at.allow**. If that  
5502 file does not exist, the file **/usr/lib/cron/at.deny** shall be checked to determine whether the user  
5503 shall be denied access to *at*. If neither file exists, only a process with the appropriate privileges  
5504 shall be allowed to submit a job. If only **at.deny** exists and is empty, global usage shall be  
5505 permitted. The **at.allow** and **at.deny** files shall consist of one user name per line.

5506 **OPTIONS**

5507 The *at* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section 12.2,  
5508 Utility Syntax Guidelines.

5509 The following options shall be supported:

5510 **-f file** Specify the pathname of a file to be used as the source of the *at-job*, instead of  
5511 standard input.

5512 **-l** (The letter ell.) Report all jobs scheduled for the invoking user if no *at\_job\_id*  
5513 operands are specified. If *at\_job\_ids* are specified, report only information for these  
5514 jobs. The output shall be written to standard output.

5515 **-m** Send mail to the invoking user after the *at-job* has run, announcing its completion.  
5516 Standard output and standard error produced by the *at-job* shall be mailed to the  
5517 user as well, unless redirected elsewhere. Mail shall be sent even if the job  
5518 produces no output.

5519 If **-m** is not used, the job's standard output and standard error shall be provided to  
5520 the user by means of mail, unless they are redirected elsewhere; if there is no such  
5521 output to provide, the implementation need not notify the user of the job's  
5522 completion.

- 5523           **-q queueName**  
 5524           Specify in which queue to schedule a job for submission. When used with the **-l**  
 5525           option, limit the search to that particular queue. By default, at-jobs shall be  
 5526           scheduled in queue *a*. In contrast, queue *b* shall be reserved for batch jobs; see  
 5527           *batch*. The meanings of all other *queueNames* are implementation-defined. If **-q** is  
 5528           specified along with either of the **-t time\_arg** or *timespec* arguments, the results are  
 5529           unspecified.
- 5530           **-r**  
 5531           Remove the jobs with the specified *at\_job\_id* operands that were previously  
 5532           scheduled by the *at* utility.
- 5533           **-t time\_arg** Submit the job to be run at the time specified by the *time* option-argument, which  
 5533           the application shall ensure has the format as specified by the *touch -t time* utility.

## 5534 OPERANDS

5535           The following operands shall be supported:

- 5536           **at\_job\_id** The name reported by a previous invocation of the *at* utility at the time the job was  
 5537           scheduled.
- 5538           **timespec** Submit the job to be run at the date and time specified. All of the *timespec* operands  
 5539           are interpreted as if they were separated by <space>s and concatenated, and shall  
 5540           be parsed as described in the grammar at the end of this section. The date and time  
 5541           shall be interpreted as being in the timezone of the user (as determined by the *TZ*  
 5542           variable), unless a timezone name appears as part of *time*, below.

5543           In the POSIX locale, the following describes the three parts of the time  
 5544           specification string. All of the values from the *LC\_TIME* categories in the POSIX  
 5545           locale shall be recognized in a case-insensitive manner.

- 5546           **time** The time can be specified as one, two, or four digits. One-digit and  
 5547           two-digit numbers shall be taken to be hours; four-digit numbers to  
 5548           be hours and minutes. The time can alternatively be specified as two  
 5549           numbers separated by a colon, meaning *hour:min*. An AM/PM  
 5550           indication (one of the values from the **am\_pm** keywords in the  
 5551           *LC\_TIME* locale category) can follow the time; otherwise, a 24-hour  
 5552           clock time shall be understood. A timezone name can also follow to  
 5553           further qualify the time. The acceptable timezone names are  
 5554           implementation-defined, except that they shall be case-insensitive  
 5555           and the string **utc** is supported to indicate the time is in Coordinated  
 5556           Universal Time. In the POSIX locale, the *time* field can also be one of  
 5557           the following tokens:

- 5558           **midnight** Indicates the time 12:00 am (00:00).
- 5559           **noon** Indicates the time 12:00 pm.
- 5560           **now** Indicates the current day and time. Invoking *at <now>*  
 5561           shall submit an at-job for potentially immediate  
 5562           execution (that is, subject only to unspecified  
 5563           scheduling delays).

- 5564           **date** An optional *date* can be specified as either a month name (one of the  
 5565           values from the **mon** or **abmon** keywords in the *LC\_TIME* locale  
 5566           category) followed by a day number (and possibly year number  
 5567           preceded by a comma), or a day of the week (one of the values from  
 5568           the **day** or **abday** keywords in the *LC\_TIME* locale category). In the  
 5569           POSIX locale, two special days shall be recognized:

5570                   **today**       Indicates the current day.  
5571                   **tomorrow**    Indicates the day following the current day.  
5572                   If no *date* is given, **today** shall be assumed if the given time is greater  
5573                   than the current time, and **tomorrow** shall be assumed if it is less. If  
5574                   the given month is less than the current month (and no year is given),  
5575                   next year shall be assumed.  
5576                   **increment**     The optional *increment* shall be a number preceded by a plus sign  
5577                   ('+') and suffixed by one of the following: **minutes**, **hours**, **days**,  
5578                   **weeks**, **months**, or **years**. (The singular forms shall also be  
5579                   accepted.) The keyword **next** shall be equivalent to an increment  
5580                   number of +1. For example, the following are equivalent commands:  
5581                   at 2pm + 1 week  
5582                   at 2pm next week  
5583                   The following grammar describes the precise format of *timespec* in the POSIX locale. The general  
5584                   conventions for this style of grammar are described in Section 1.10 (on page 19). This formal  
5585                   syntax shall take precedence over the preceding text syntax description. The longest possible  
5586                   token or delimiter shall be recognized at a given point. When used in a *timespec*, white space  
5587                   shall also delimit tokens.  
5588                   %token hr24clock\_hr\_min  
5589                   %token hr24clock\_hour  
5590                   /\*  
5591                    An hr24clock\_hr\_min is a one, two, or four-digit number. A one-digit  
5592                    or two-digit number constitutes an hr24clock\_hour. An hr24clock\_hour  
5593                    may be any of the single digits [0,9], or may be double digits, ranging  
5594                    from [00,23]. If an hr24clock\_hr\_min is a four-digit number, the  
5595                    first two digits shall be a valid hr24clock\_hour, while the last two  
5596                    represent the number of minutes, from [00,59].  
5597                   \*/  
5598                   %token wallclock\_hr\_min  
5599                   %token wallclock\_hour  
5600                   /\*  
5601                    A wallclock\_hr\_min is a one, two-digit, or four-digit number.  
5602                    A one-digit or two-digit number constitutes a wallclock\_hour.  
5603                    A wallclock\_hour may be any of the single digits [1,9], or may  
5604                    be double digits, ranging from [01,12]. If a wallclock\_hr\_min  
5605                    is a four-digit number, the first two digits shall be a valid  
5606                    wallclock\_hour, while the last two represent the number of  
5607                    minutes, from [00,59].  
5608                   \*/  
5609                   %token minute  
5610                   /\*  
5611                    A minute is a one or two-digit number whose value can be [0,9]  
5612                    or [00,59].  
5613                   \*/  
5614                   %token day\_number  
5615                   /\*  
5616                    A day\_number is a number in the range appropriate for the particular  
5617                    month and year specified by month\_name and year\_number, respectively.

```
5618 If no year_number is given, the current year is assumed if the given
5619 date and time are later this year. If no year_number is given and
5620 the date and time have already occurred this year and the month is
5621 not the current month, next year is the assumed year.
5622 */
5623 %token year_number
5624 /*
5625 A year_number is a four-digit number representing the year A.D., in
5626 which the at_job is to be run.
5627 */
5628 %token inc_number
5629 /*
5630 The inc_number is the number of times the succeeding increment
5631 period is to be added to the specified date and time.
5632 */
5633 %token timezone_name
5634 /*
5635 The name of an optional timezone suffix to the time field, in an
5636 implementation-defined format.
5637 */
5638 %token month_name
5639 /*
5640 One of the values from the mon or abmon keywords in the LC_TIME
5641 locale category.
5642 */
5643 %token day_of_week
5644 /*
5645 One of the values from the day or abday keywords in the LC_TIME
5646 locale category.
5647 */
5648 %token am_pm
5649 /*
5650 One of the values from the am_pm keyword in the LC_TIME locale
5651 category.
5652 */
5653 %start timespec
5654 %%
5655 timespec : time
5656 | time date
5657 | time increment
5658 | time date increment
5659 | nowspec
5660 ;
5661 nowspec : "now"
5662 | "now" increment
5663 ;
5664 time : hr24clock_hr_min
5665 | hr24clock_hr_min timezone_name
```

```
5666 | hr24clock_hour ":" minute
5667 | hr24clock_hour ":" minute timezone_name
5668 | wallclock_hr_min am_pm
5669 | wallclock_hr_min am_pm timezone_name
5670 | wallclock_hour ":" minute am_pm
5671 | wallclock_hour ":" minute am_pm timezone_name
5672 | "noon"
5673 | "midnight"
5674 ;
5675 date : month_name day_number
5676 | month_name day_number ", " year_number
5677 | day_of_week
5678 | "today"
5679 | "tomorrow"
5680 ;
5681 increment : "+" inc_number inc_period
5682 | "next" inc_period
5683 ;
5684 inc_period : "minute" | "minutes"
5685 | "hour" | "hours"
5686 | "day" | "days"
5687 | "week" | "weeks"
5688 | "month" | "months"
5689 | "year" | "years"
5690 ;
```

## 5691 STDIN

5692 The standard input shall be a text file consisting of commands acceptable to the shell command  
5693 language described in Chapter 2 (on page 29). The standard input shall only be used if no **-f file**  
5694 option is specified.

## 5695 INPUT FILES

5696 See the STDIN section.

5697 XSI The text files **/usr/lib/cron/at.allow** and **/usr/lib/cron/at.deny** shall contain zero or more user  
5698 names, one per line, of users who are, respectively, authorized or denied access to the **at** and  
5699 **batch** utilities.

## 5700 ENVIRONMENT VARIABLES

5701 The following environment variables shall affect the execution of **at**:

5702 **LANG** Provide a default value for the internationalization variables that are unset or null.  
5703 (See the **Base Definitions** volume of IEEE Std 1003.1-2001, Section 8.2,  
5704 Internationalization Variables for the precedence of internationalization variables  
5705 used to determine the values of locale categories.)

5706 **LC\_ALL** If set to a non-empty string value, override the values of all the other  
5707 internationalization variables.

5708 **LC\_CTYPE** Determine the locale for the interpretation of sequences of bytes of text data as  
5709 characters (for example, single-byte as opposed to multi-byte characters in  
5710 arguments and input files).

## 5711 **LC\_MESSAGES**

5712 Determine the locale that should be used to affect the format and contents of

|      |     |                                                                                                                                                                                                                                                                                                                                                                                                                                        |
|------|-----|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 5713 |     | diagnostic messages written to standard error and informative messages written to standard output.                                                                                                                                                                                                                                                                                                                                     |
| 5715 | XSI | <b>NLSPATH</b> Determine the location of message catalogs for the processing of <i>LC_MESSAGES</i> .                                                                                                                                                                                                                                                                                                                                   |
| 5716 |     | <b>LC_TIME</b> Determine the format and contents for date and time strings written and accepted by <i>at</i> .                                                                                                                                                                                                                                                                                                                         |
| 5717 |     |                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| 5718 |     | <b>SHELL</b> Determine a name of a command interpreter to be used to invoke the <i>at</i> -job. If the variable is unset or null, <i>sh</i> shall be used. If it is set to a value other than a name for <i>sh</i> , the implementation shall do one of the following: use that shell; use <i>sh</i> ; use the login shell from the user database; or any of the preceding accompanied by a warning diagnostic about which was chosen. |
| 5719 |     |                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| 5720 |     |                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| 5721 |     |                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| 5722 |     |                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| 5723 |     | <b>TZ</b> Determine the timezone. The job shall be submitted for execution at the time specified by <i>timespec</i> or <b>-t</b> <i>time</i> relative to the timezone specified by the <b>TZ</b> variable. If <i>timespec</i> specifies a timezone, it shall override <b>TZ</b> . If <i>timespec</i> does not specify a timezone and <b>TZ</b> is unset or null, an unspecified default timezone shall be used.                        |
| 5724 |     |                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| 5725 |     |                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| 5726 |     |                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| 5727 |     |                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| 5728 |     | <b>ASYNCHRONOUS EVENTS</b>                                                                                                                                                                                                                                                                                                                                                                                                             |
| 5729 |     | Default.                                                                                                                                                                                                                                                                                                                                                                                                                               |
| 5730 |     | <b>STDOUT</b>                                                                                                                                                                                                                                                                                                                                                                                                                          |
| 5731 |     | When standard input is a terminal, prompts of unspecified format for each line of the user input described in the <b>STDIN</b> section may be written to standard output.                                                                                                                                                                                                                                                              |
| 5732 |     |                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| 5733 |     | In the POSIX locale, the following shall be written to the standard output for each job when jobs are listed in response to the <b>-l</b> option:                                                                                                                                                                                                                                                                                      |
| 5734 |     |                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| 5735 |     | "%s\t%s\n", <i>at_job_id</i> , < <i>date</i> >                                                                                                                                                                                                                                                                                                                                                                                         |
| 5736 |     | where <i>date</i> shall be equivalent in format to the output of:                                                                                                                                                                                                                                                                                                                                                                      |
| 5737 |     | date +"%a %b %e %T %Y"                                                                                                                                                                                                                                                                                                                                                                                                                 |
| 5738 |     | The date and time written shall be adjusted so that they appear in the timezone of the user (as determined by the <b>TZ</b> variable).                                                                                                                                                                                                                                                                                                 |
| 5739 |     |                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| 5740 |     | <b>STDERR</b>                                                                                                                                                                                                                                                                                                                                                                                                                          |
| 5741 |     | In the POSIX locale, the following shall be written to standard error when a job has been successfully submitted:                                                                                                                                                                                                                                                                                                                      |
| 5742 |     |                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| 5743 |     | " job %s at %s\n", <i>at_job_id</i> , < <i>date</i> >                                                                                                                                                                                                                                                                                                                                                                                  |
| 5744 |     | where <i>date</i> has the same format as that described in the <b>STDOUT</b> section. Neither this, nor warning messages concerning the selection of the command interpreter, shall be considered a diagnostic that changes the exit status.                                                                                                                                                                                           |
| 5745 |     |                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| 5746 |     |                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| 5747 |     | Diagnostic messages, if any, shall be written to standard error.                                                                                                                                                                                                                                                                                                                                                                       |
| 5748 |     | <b>OUTPUT FILES</b>                                                                                                                                                                                                                                                                                                                                                                                                                    |
| 5749 |     | None.                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| 5750 |     | <b>EXTENDED DESCRIPTION</b>                                                                                                                                                                                                                                                                                                                                                                                                            |
| 5751 |     | None.                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| 5752 |     | <b>EXIT STATUS</b>                                                                                                                                                                                                                                                                                                                                                                                                                     |
| 5753 |     | The following exit values shall be returned:                                                                                                                                                                                                                                                                                                                                                                                           |
| 5754 |     | 0 The <i>at</i> utility successfully submitted, removed, or listed a job or jobs.                                                                                                                                                                                                                                                                                                                                                      |

5755 >0 An error occurred.

## 5756 CONSEQUENCES OF ERRORS

5757 The job shall not be scheduled, removed, or listed.

## 5758 APPLICATION USAGE

5759 The format of the at command line shown here is guaranteed only for the POSIX locale. Other  
5760 cultures may be supported with substantially different interfaces, although implementations are  
5761 encouraged to provide comparable levels of functionality.

5762 Since the commands run in a separate shell invocation, running in a separate process group with  
5763 no controlling terminal, open file descriptors, traps, and priority inherited from the invoking  
5764 environment are lost.

5765 Some implementations do not allow substitution of different shells using *SHELL*. System V  
5766 systems, for example, have used the login shell value for the user in /etc/passwd. To select  
5767 reliably another command interpreter, the user must include it as part of the script, such as:

```
5768 $ at 1800
5769 myshell myscript
5770 EOT
5771 job ... at ...
5772 $
```

## 5773 EXAMPLES

1. This sequence can be used at a terminal:

```
5775 at -m 0730 tomorrow
5776 sort < file >outfile
5777 EOT
```

2. This sequence, which demonstrates redirecting standard error to a pipe, is useful in a command procedure (the sequence of output redirection specifications is significant):

```
5780 at now + 1 hour <<!
5781 diff file1 file2 2>&1 >outfile | mailx mygroup
5782 !
```

3. To have a job reschedule itself, at can be invoked from within the at-job. For example, this daily processing script named **my.daily** runs every day (although crontab is a more appropriate vehicle for such work):

```
5786 # my.daily runs every day
5787 daily processing
5788 at now tomorrow < my.daily
```

4. The spacing of the three portions of the POSIX locale *timespec* is quite flexible as long as there are no ambiguities. Examples of various times and operand presentation include:

```
5791 at 0815am Jan 24
5792 at 8 :15amjan24
5793 at now "+ 1day"
5794 at 5 pm FRIday
5795 at '17
5796 utc+
5797 30minutes'
```

**RATIONALE**

5798 The *at* utility reads from standard input the commands to be executed at a later time. It may be  
5799 useful to redirect standard output and standard error within the specified commands.

5801 The **-t time** option was added as a new capability to support an internationalized way of  
5802 specifying a time for execution of the submitted job.

5803 Early proposals added a “jobname” concept as a way of giving submitted jobs names that are  
5804 meaningful to the user submitting them. The historical, system-specified *at\_job\_id* gives no  
5805 indication of what the job is. Upon further reflection, it was decided that the benefit of this was  
5806 not worth the change in historical interface. The *at* functionality is useful in simple  
5807 environments, but in large or complex situations, the functionality provided by the Batch  
5808 Services option is more suitable.

5809 The **-q** option historically has been an undocumented option, used mainly by the *batch* utility.

5810 The System V **-m** option was added to provide a method for informing users that an at-job had  
5811 completed. Otherwise, users are only informed when output to standard error or standard  
5812 output are not redirected.

5813 The behavior of *at <now>* was changed in an early proposal from being unspecified to  
5814 submitting a job for potentially immediate execution. Historical BSD *at* implementations  
5815 support this. Historical System V implementations give an error in that case, but a change to the  
5816 System V versions should have no backwards-compatibility ramifications.

5817 On BSD-based systems, a **-u user** option has allowed those with appropriate privileges to access  
5818 the work of other users. Since this is primarily a system administration feature and is not  
5819 universally implemented, it has been omitted. Similarly, a specification for the output format for  
5820 a user with appropriate privileges viewing the queues of other users has been omitted.

5821 The **-f file** option from System V is used instead of the BSD method of using the last operand as  
5822 the pathname. The BSD method is ambiguous—does:

5823     *at 1200 friday*

5824 mean the same thing if there is a file named **friday** in the current directory?

5825 The *at\_job\_id* is composed of a limited character set in historical practice, and it is mandated here  
5826 to invalidate systems that might try using characters that require shell quoting or that could not  
5827 be easily parsed by shell scripts.

5828 The *at* utility varies between System V and BSD systems in the way timezones are used. On  
5829 System V systems, the *TZ* variable affects the at-job submission times and the times displayed  
5830 for the user. On BSD systems, *TZ* is not taken into account. The BSD behavior is easily achieved  
5831 with the current specification. If the user wishes to have the timezone default to that of the  
5832 system, they merely need to issue the *at* command immediately following an unsetting or null  
5833 assignment to *TZ*. For example:

5834     *TZ= at noon ...*

5835 gives the desired BSD result.

5836 While the yacc-like grammar specified in the OPERANDS section is lexically unambiguous with  
5837 respect to the digit strings, a lexical analyzer would probably be written to look for and return  
5838 digit strings in those cases. The parser could then check whether the digit string returned is a  
5839 valid *day\_number*, *year\_number*, and so on, based on the context.

5840 **FUTURE DIRECTIONS**

5841       None.

5842 **SEE ALSO**5843       *batch, crontab*5844 **CHANGE HISTORY**

5845       First released in Issue 2.

5846 **Issue 6**

5847       This utility is marked as part of the User Portability Utilities option.

5848       The following new requirements on POSIX implementations derive from alignment with the  
5849       Single UNIX Specification:

- 5850       • If **-m** is not used, the job's standard output and standard error are provided to the user by  
5851       mail.

5852       The effects of using the **-q** and **-t** options as defined in the IEEE P1003.2b draft standard are  
5853       specified.

5854       The normative text is reworded to avoid use of the term "must" for application requirements.

5855 **NAME**

5856 awk — pattern scanning and processing language

5857 **SYNOPSIS**

5858 awk [-F *ERE*] [-v *assignment*] ... *program* [*argument* ...]

5859 awk [-F *ERE*] -f *progfile* ... [-v *assignment*] ... [*argument* ...]

5860 **DESCRIPTION**

5861 The *awk* utility shall execute programs written in the *awk* programming language, which is  
5862 specialized for textual data manipulation. An *awk* program is a sequence of patterns and  
5863 corresponding actions. When input is read that matches a pattern, the action associated with  
5864 that pattern is carried out.

5865 Input shall be interpreted as a sequence of records. By default, a record is a line, less its  
5866 terminating <newline>, but this can be changed by using the **RS** built-in variable. Each record of  
5867 input shall be matched in turn against each pattern in the program. For each pattern matched,  
5868 the associated action shall be executed.

5869 The *awk* utility shall interpret each input record as a sequence of fields where, by default, a field  
5870 is a string of non-<blank>s. This default white-space field delimiter can be changed by using the  
5871 **FS** built-in variable or **-F *ERE***. The *awk* utility shall denote the first field in a record \$1, the  
5872 second \$2, and so on. The symbol \$0 shall refer to the entire record; setting any other field causes  
5873 the re-evaluation of \$0. Assigning to \$0 shall reset the values of all other fields and the **NF** built-  
5874 in variable.

5875 **OPTIONS**

5876 The *awk* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section  
5877 12.2, Utility Syntax Guidelines.

5878 The following options shall be supported:

5879 **-F *ERE*** Define the input field separator to be the extended regular expression *ERE*, before  
5880 any input is read; see **Regular Expressions** (on page 161).

5881 **-f *progfile*** Specify the pathname of the file *progfile* containing an *awk* program. If multiple  
5882 instances of this option are specified, the concatenation of the files specified as  
5883 *progfile* in the order specified shall be the *awk* program. The *awk* program can  
5884 alternatively be specified in the command line as a single argument.

5885 **-v *assignment***

5886 The application shall ensure that the *assignment* argument is in the same form as an  
5887 *assignment* operand. The specified variable *assignment* shall occur prior to  
5888 executing the *awk* program, including the actions associated with **BEGIN** patterns  
5889 (if any). Multiple occurrences of this option can be specified.

5890 **OPERANDS**

5891 The following operands shall be supported:

5892 **program** If no **-f** option is specified, the first operand to *awk* shall be the text of the *awk*  
5893 program. The application shall supply the *program* operand as a single argument to  
5894 *awk*. If the text does not end in a <newline>, *awk* shall interpret the text as if it did.

5895 **argument** Either of the following two types of *argument* can be intermixed:

5896 **file** A pathname of a file that contains the input to be read, which is  
5897 matched against the set of patterns in the program. If no *file* operands  
5898 are specified, or if a *file* operand is ‘-’, the standard input shall be  
5899 used.

5900                   **assignment**   An operand that begins with an underscore or alphabetic character  
5901                   from the portable character set (see the table in the Base Definitions  
5902                   volume of IEEE Std 1003.1-2001, Section 6.1, Portable Character Set),  
5903                   followed by a sequence of underscores, digits, and alphabets from  
5904                   the portable character set, followed by the '=' character, shall  
5905                   specify a variable assignment rather than a pathname. The  
5906                   characters before the '=' represent the name of an *awk* variable; if  
5907                   that name is an *awk* reserved word (see **Grammar** (on page 170)) the  
5908                   behavior is undefined. The characters following the equal sign shall  
5909                   be interpreted as if they appeared in the *awk* program preceded and  
5910                   followed by a double-quote ('") character, as a **STRING** token (see  
5911                   **Grammar** (on page 170)), except that if the last character is an  
5912                   unescape backslash, it shall be interpreted as a literal backslash  
5913                   rather than as the first character of the sequence "\ ". The variable  
5914                   shall be assigned the value of that **STRING** token and, if  
5915                   appropriate, shall be considered a *numeric string* (see **Expressions in**  
5916                   *awk* (on page 156)), the variable shall also be assigned its numeric  
5917                   value. Each such variable assignment shall occur just prior to the  
5918                   processing of the following *file*, if any. Thus, an assignment before  
5919                   the first *file* argument shall be executed after the **BEGIN** actions (if  
5920                   any), while an assignment after the last *file* argument shall occur  
5921                   before the **END** actions (if any). If there are no *file* arguments,  
5922                   assignments shall be executed before processing the standard input.

## 5923   **STDIN**

5924                   The standard input shall be used only if no *file* operands are specified, or if a *file* operand is '-';  
5925                   see the INPUT FILES section. If the *awk* program contains no actions and no patterns, but is  
5926                   otherwise a valid *awk* program, standard input and any *file* operands shall not be read and *awk*  
5927                   shall exit with a return status of zero.

## 5928   **INPUT FILES**

5929                   Input files to the *awk* program from any of the following sources shall be text files:

- 5930                   • Any *file* operands or their equivalents, achieved by modifying the *awk* variables **ARGV** and  
5931                   **ARGC**
- 5932                   • Standard input in the absence of any *file* operands
- 5933                   • Arguments to the **getline** function

5934                   Whether the variable **RS** is set to a value other than a <newline> or not, for these files,  
5935                   implementations shall support records terminated with the specified separator up to  
5936                   {LINE\_MAX} bytes and may support longer records.

5937                   If **-f progfile** is specified, the application shall ensure that the files named by each of the *progfile*  
5938                   option-arguments are text files and their concatenation, in the same order as they appear in the  
5939                   arguments, is an *awk* program.

## 5940   **ENVIRONMENT VARIABLES**

5941                   The following environment variables shall affect the execution of *awk*:

5942                   **LANG**       Provide a default value for the internationalization variables that are unset or null.  
5943                   (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2,  
5944                   Internationalization Variables for the precedence of internationalization variables  
5945                   used to determine the values of locale categories.)

|          |                             |                                                                                                                                                                                                                                                                                                                                                                                                                 |
|----------|-----------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 5946     | <i>LC_ALL</i>               | If set to a non-empty string value, override the values of all the other internationalization variables.                                                                                                                                                                                                                                                                                                        |
| 5947     |                             |                                                                                                                                                                                                                                                                                                                                                                                                                 |
| 5948     | <i>LC_COLLATE</i>           | Determine the locale for the behavior of ranges, equivalence classes, and multi-character collating elements within regular expressions and in comparisons of string values.                                                                                                                                                                                                                                    |
| 5949     |                             |                                                                                                                                                                                                                                                                                                                                                                                                                 |
| 5950     |                             |                                                                                                                                                                                                                                                                                                                                                                                                                 |
| 5951     |                             |                                                                                                                                                                                                                                                                                                                                                                                                                 |
| 5952     | <i>LC_CTYPE</i>             | Determine the locale for the interpretation of sequences of bytes of text data as characters (for example, single-byte as opposed to multi-byte characters in arguments and input files), the behavior of character classes within regular expressions, the identification of characters as letters, and the mapping of uppercase and lowercase characters for the <b>toupper</b> and <b>tolower</b> functions. |
| 5953     |                             |                                                                                                                                                                                                                                                                                                                                                                                                                 |
| 5954     |                             |                                                                                                                                                                                                                                                                                                                                                                                                                 |
| 5955     |                             |                                                                                                                                                                                                                                                                                                                                                                                                                 |
| 5956     |                             |                                                                                                                                                                                                                                                                                                                                                                                                                 |
| 5957     | <i>LC_MESSAGES</i>          | Determine the locale that should be used to affect the format and contents of diagnostic messages written to standard error.                                                                                                                                                                                                                                                                                    |
| 5958     |                             |                                                                                                                                                                                                                                                                                                                                                                                                                 |
| 5959     |                             |                                                                                                                                                                                                                                                                                                                                                                                                                 |
| 5960     | <i>LC_NUMERIC</i>           | Determine the radix character used when interpreting numeric input, performing conversions between numeric and string values, and formatting numeric output. Regardless of locale, the period character (the decimal-point character of the POSIX locale) is the decimal-point character recognized in processing awk programs (including assignments in command line arguments).                               |
| 5961     |                             |                                                                                                                                                                                                                                                                                                                                                                                                                 |
| 5962     |                             |                                                                                                                                                                                                                                                                                                                                                                                                                 |
| 5963     |                             |                                                                                                                                                                                                                                                                                                                                                                                                                 |
| 5964     |                             |                                                                                                                                                                                                                                                                                                                                                                                                                 |
| 5965     |                             |                                                                                                                                                                                                                                                                                                                                                                                                                 |
| 5966 XSI | <i>NLSPATH</i>              | Determine the location of message catalogs for the processing of <i>LC_MESSAGES</i> .                                                                                                                                                                                                                                                                                                                           |
| 5967     | <i>PATH</i>                 | Determine the search path when looking for commands executed by <i>system(expr)</i> , or input and output pipes; see the Base Definitions volume of IEEE Std 1003.1-2001, Chapter 8, Environment Variables.                                                                                                                                                                                                     |
| 5968     |                             |                                                                                                                                                                                                                                                                                                                                                                                                                 |
| 5969     |                             |                                                                                                                                                                                                                                                                                                                                                                                                                 |
| 5970     |                             | In addition, all environment variables shall be visible via the <i>awk</i> variable <b>ENVIRON</b> .                                                                                                                                                                                                                                                                                                            |
| 5971     | <b>ASYNCHRONOUS EVENTS</b>  |                                                                                                                                                                                                                                                                                                                                                                                                                 |
| 5972     |                             | Default.                                                                                                                                                                                                                                                                                                                                                                                                        |
| 5973     | <b>STDOUT</b>               |                                                                                                                                                                                                                                                                                                                                                                                                                 |
| 5974     |                             | The nature of the output files depends on the <i>awk</i> program.                                                                                                                                                                                                                                                                                                                                               |
| 5975     | <b>STDERR</b>               |                                                                                                                                                                                                                                                                                                                                                                                                                 |
| 5976     |                             | The standard error shall be used only for diagnostic messages.                                                                                                                                                                                                                                                                                                                                                  |
| 5977     | <b>OUTPUT FILES</b>         |                                                                                                                                                                                                                                                                                                                                                                                                                 |
| 5978     |                             | The nature of the output files depends on the <i>awk</i> program.                                                                                                                                                                                                                                                                                                                                               |
| 5979     | <b>EXTENDED DESCRIPTION</b> |                                                                                                                                                                                                                                                                                                                                                                                                                 |
| 5980     |                             | <b>Overall Program Structure</b>                                                                                                                                                                                                                                                                                                                                                                                |
| 5981     |                             | An <i>awk</i> program is composed of pairs of the form:                                                                                                                                                                                                                                                                                                                                                         |
| 5982     |                             | <i>pattern { action }</i>                                                                                                                                                                                                                                                                                                                                                                                       |
| 5983     |                             | Either the pattern or the action (including the enclosing brace characters) can be omitted.                                                                                                                                                                                                                                                                                                                     |
| 5984     |                             | A missing pattern shall match any record of input, and a missing action shall be equivalent to:                                                                                                                                                                                                                                                                                                                 |
| 5985     |                             | <i>{ print }</i>                                                                                                                                                                                                                                                                                                                                                                                                |
| 5986     |                             | Execution of the <i>awk</i> program shall start by first executing the actions associated with all <b>BEGIN</b> patterns in the order they occur in the program. Then each <i>file</i> operand (or standard input if no                                                                                                                                                                                         |
| 5987     |                             |                                                                                                                                                                                                                                                                                                                                                                                                                 |

5988 files were specified) shall be processed in turn by reading data from the file until a record  
 5989 separator is seen (<newline> by default). Before the first reference to a field in the record is  
 5990 evaluated, the record shall be split into fields, according to the rules in **Regular Expressions** (on  
 5991 page 161), using the value of **FS** that was current at the time the record was read. Each pattern in  
 5992 the program then shall be evaluated in the order of occurrence, and the action associated with  
 5993 each pattern that matches the current record executed. The action for a matching pattern shall be  
 5994 executed before evaluating subsequent patterns. Finally, the actions associated with all **END**  
 5995 patterns shall be executed in the order they occur in the program.

## 5996 Expressions in awk

5997 Expressions describe computations used in *patterns* and *actions*. In the following table, valid  
 5998 expression operations are given in groups from highest precedence first to lowest precedence  
 5999 last, with equal-precedence operators grouped between horizontal lines. In expression  
 6000 evaluation, where the grammar is formally ambiguous, higher precedence operators shall be  
 6001 evaluated before lower precedence operators. In this table *expr*, *expr1*, *expr2*, and *expr3* represent  
 6002 any expression, while *lvalue* represents any entity that can be assigned to (that is, on the left side  
 6003 of an assignment operator). The precise syntax of expressions is given in **Grammar** (on page  
 6004 170).

6005 **Table 4-1** Expressions in Decreasing Precedence in *awk*

| Syntax                     | Name                     | Type of Result      | Associativity |
|----------------------------|--------------------------|---------------------|---------------|
| ( <i>expr</i> )            | Grouping                 | Type of <i>expr</i> | N/A           |
| \$ <i>expr</i>             | Field reference          | String              | N/A           |
| <i>lvalue</i> ++           | Pre-increment            | Numeric             | N/A           |
| <i>lvalue</i> --           | Pre-decrement            | Numeric             | N/A           |
| <i>lvalue</i> ++           | Post-increment           | Numeric             | N/A           |
| <i>lvalue</i> --           | Post-decrement           | Numeric             | N/A           |
| <i>expr</i> ^ <i>expr</i>  | Exponentiation           | Numeric             | Right         |
| ! <i>expr</i>              | Logical not              | Numeric             | N/A           |
| + <i>expr</i>              | Unary plus               | Numeric             | N/A           |
| - <i>expr</i>              | Unary minus              | Numeric             | N/A           |
| <i>expr</i> * <i>expr</i>  | Multiplication           | Numeric             | Left          |
| <i>expr</i> / <i>expr</i>  | Division                 | Numeric             | Left          |
| <i>expr</i> % <i>expr</i>  | Modulus                  | Numeric             | Left          |
| <i>expr</i> + <i>expr</i>  | Addition                 | Numeric             | Left          |
| <i>expr</i> - <i>expr</i>  | Subtraction              | Numeric             | Left          |
| <i>expr</i> <i>expr</i>    | String concatenation     | String              | Left          |
| <i>expr</i> < <i>expr</i>  | Less than                | Numeric             | None          |
| <i>expr</i> <= <i>expr</i> | Less than or equal to    | Numeric             | None          |
| <i>expr</i> != <i>expr</i> | Not equal to             | Numeric             | None          |
| <i>expr</i> == <i>expr</i> | Equal to                 | Numeric             | None          |
| <i>expr</i> > <i>expr</i>  | Greater than             | Numeric             | None          |
| <i>expr</i> >= <i>expr</i> | Greater than or equal to | Numeric             | None          |

|      | Syntax                             | Name                             | Type of Result                                | Associativity |
|------|------------------------------------|----------------------------------|-----------------------------------------------|---------------|
| 6032 | <code>expr ~ expr</code>           | ERE match                        | Numeric                                       | None          |
| 6033 | <code>expr !~ expr</code>          | ERE non-match                    | Numeric                                       | None          |
| 6034 | <code>expr in array</code>         | Array membership                 | Numeric                                       | Left          |
| 6035 | <code>( index ) in array</code>    | Multi-dimension array membership | Numeric                                       | Left          |
| 6037 | <code>expr &amp;&amp; expr</code>  | Logical AND                      | Numeric                                       | Left          |
| 6038 | <code>expr    expr</code>          | Logical OR                       | Numeric                                       | Left          |
| 6039 | <code>expr1 ? expr2 : expr3</code> | Conditional expression           | Type of selected <i>expr2</i> or <i>expr3</i> | Right         |
| 6041 | <code>lvalue ^= expr</code>        | Exponentiation assignment        | Numeric                                       | Right         |
| 6042 | <code>lvalue %= expr</code>        | Modulus assignment               | Numeric                                       | Right         |
| 6043 | <code>lvalue *= expr</code>        | Multiplication assignment        | Numeric                                       | Right         |
| 6044 | <code>lvalue /= expr</code>        | Division assignment              | Numeric                                       | Right         |
| 6045 | <code>lvalue += expr</code>        | Addition assignment              | Numeric                                       | Right         |
| 6046 | <code>lvalue -= expr</code>        | Subtraction assignment           | Numeric                                       | Right         |
| 6047 | <code>lvalue = expr</code>         | Assignment                       | Type of <i>expr</i>                           | Right         |

6048 Each expression shall have either a string value, a numeric value, or both. Except as stated for  
 6049 specific contexts, the value of an expression shall be implicitly converted to the type needed for  
 6050 the context in which it is used. A string value shall be converted to a numeric value by the  
 6051 equivalent of the following calls to functions defined by the ISO C standard:

```
6052 setlocale(LC_NUMERIC, "") ;
6053 numeric_value = atof(string_value) ;
```

6054 A numeric value that is exactly equal to the value of an integer (see Section 1.7.2 (on page 7))  
 6055 shall be converted to a string by the equivalent of a call to the **sprintf** function (see **String**  
 6056 **Functions** (on page 167)) with the string "%d" as the *fmt* argument and the numeric value being  
 6057 converted as the first and only *expr* argument. Any other numeric value shall be converted to a  
 6058 string by the equivalent of a call to the **sprintf** function with the value of the variable  
 6059 **CONVFMT** as the *fmt* argument and the numeric value being converted as the first and only  
 6060 *expr* argument. The result of the conversion is unspecified if the value of **CONVFMT** is not a  
 6061 floating-point format specification. This volume of IEEE Std 1003.1-2001 specifies no explicit  
 6062 conversions between numbers and strings. An application can force an expression to be treated  
 6063 as a number by adding zero to it, or can force it to be treated as a string by concatenating the null  
 6064 string (" ") to it.

6065 A string value shall be considered a *numeric string* if it comes from one of the following:

- 6066 1. Field variables
- 6067 2. Input from the **getline()** function
- 6068 3. **FILENAME**
- 6069 4. **ARGV** array elements
- 6070 5. **ENVIRON** array elements
- 6071 6. Array elements created by the **split()** function
- 6072 7. A command line variable assignment

6073            8. Variable assignment from another numeric string variable  
6074        and after all the following conversions have been applied, the resulting string would lexically be  
6075        recognized as a **NUMBER** token as described by the lexical conventions in **Grammar** (on page  
6076        170):  
6077        • All leading and trailing <blank>s are discarded.  
6078        • If the first non-<blank> is '+' or '-', it is discarded.  
6079        • Changing each occurrence of the decimal point character from the current locale to a period.  
6080        If a '-' character is ignored in the preceding description, the numeric value of the *numeric string*  
6081        shall be the negation of the numeric value of the recognized **NUMBER** token. Otherwise, the  
6082        numeric value of the *numeric string* shall be the numeric value of the recognized **NUMBER**  
6083        token. Whether or not a string is a *numeric string* shall be relevant only in contexts where that  
6084        term is used in this section.  
6085        When an expression is used in a Boolean context, if it has a numeric value, a value of zero shall  
6086        be treated as false and any other value shall be treated as true. Otherwise, a string value of the  
6087        null string shall be treated as false and any other value shall be treated as true. A Boolean  
6088        context shall be one of the following:  
6089        • The first subexpression of a conditional expression  
6090        • An expression operated on by logical NOT, logical AND, or logical OR  
6091        • The second expression of a **for** statement  
6092        • The expression of an **if** statement  
6093        • The expression of the **while** clause in either a **while** or **do...while** statement  
6094        • An expression used as a pattern (as in Overall Program Structure)  
6095        All arithmetic shall follow the semantics of floating-point arithmetic as specified by the ISO C  
6096        standard (see Section 1.7.2 (on page 7)).  
6097        The value of the expression:  
6098        *expr1* ^ *expr2*  
6099        shall be equivalent to the value returned by the ISO C standard function call:  
6100        *pow(expr1, expr2)*  
6101        The expression:  
6102        *lvalue* ^= *expr*  
6103        shall be equivalent to the ISO C standard expression:  
6104        *lvalue* = *pow(lvalue, expr)*  
6105        except that *lvalue* shall be evaluated only once. The value of the expression:  
6106        *expr1* % *expr2*  
6107        shall be equivalent to the value returned by the ISO C standard function call:  
6108        *fmod(expr1, expr2)*  
6109        The expression:  
6110        *lvalue* %= *expr*

6111 shall be equivalent to the ISO C standard expression:

6112     lvalue = fmod(lvalue, expr)

6113 except that lvalue shall be evaluated only once.

6114 Variables and fields shall be set by the assignment statement:

6115     lvalue = expression

6116 and the type of *expression* shall determine the resulting variable type. The assignment includes  
6117 the arithmetic assignments ("+=", "-=", "\*=", "/=", "%=", "^=", "++", "--") all of which  
6118 shall produce a numeric result. The left-hand side of an assignment and the target of increment  
6119 and decrement operators can be one of a variable, an array with index, or a field selector.

6120 The *awk* language supplies arrays that are used for storing numbers or strings. Arrays need not  
6121 be declared. They shall initially be empty, and their sizes shall change dynamically. The  
6122 subscripts, or element identifiers, are strings, providing a type of associative array capability. An  
6123 array name followed by a subscript within square brackets can be used as an lvalue and thus as  
6124 an expression, as described in the grammar; see **Grammar** (on page 170). Unsubscripted array  
6125 names can be used in only the following contexts:

- A parameter in a function definition or function call
- The **NAME** token following any use of the keyword **in** as specified in the grammar (see  
**Grammar** (on page 170)); if the name used in this context is not an array name, the behavior  
is undefined

6130 A valid array *index* shall consist of one or more comma-separated expressions, similar to the way  
6131 in which multi-dimensional arrays are indexed in some programming languages. Because *awk*  
6132 arrays are really one-dimensional, such a comma-separated list shall be converted to a single  
6133 string by concatenating the string values of the separate expressions, each separated from the  
6134 other by the value of the **SUBSEP** variable. Thus, the following two index operations shall be  
6135 equivalent:

6136     var[expr1, expr2, ... exprn]

6137     var[expr1 SUBSEP expr2 SUBSEP ... SUBSEP exprn]

6138 The application shall ensure that a multi-dimensioned *index* used with the **in** operator is  
6139 parenthesized. The **in** operator, which tests for the existence of a particular array element, shall  
6140 not cause that element to exist. Any other reference to a nonexistent array element shall  
6141 automatically create it.

6142 Comparisons (with the '<', '<=', '!=', '==', '>', and '>=' operators) shall be made  
6143 numerically if both operands are numeric, if one is numeric and the other has a string value that  
6144 is a numeric string, or if one is numeric and the other has the uninitialized value. Otherwise,  
6145 operands shall be converted to strings as required and a string comparison shall be made using  
6146 the locale-specific collation sequence. The value of the comparison expression shall be 1 if the  
6147 relation is true, or 0 if the relation is false.

6148

## Variables and Special Variables

6149

Variables can be used in an *awk* program by referencing them. With the exception of function parameters (see **User-Defined Functions** (on page 169)), they are not explicitly declared. Function parameter names shall be local to the function; all other variable names shall be global. The same name shall not be used as both a function parameter name and as the name of a function or a special *awk* variable. The same name shall not be used both as a variable name with global scope and as the name of a function. The same name shall not be used within the same scope both as a scalar variable and as an array. Uninitialized variables, including scalar variables, array elements, and field variables, shall have an uninitialized value. An uninitialized value shall have both a numeric value of zero and a string value of the empty string. Evaluation of variables with an uninitialized value, to either string or numeric, shall be determined by the context in which they are used.

6150

6151

6152

6153

6154

6155

6156

6157

6158

6159

6160

Field variables shall be designated by a '\$' followed by a number or numerical expression. The effect of the field number *expression* evaluating to anything other than a non-negative integer is unspecified; uninitialized variables or string values need not be converted to numeric values in this context. New field variables can be created by assigning a value to them. References to nonexistent fields (that is, fields after **SNF**), shall evaluate to the uninitialized value. Such references shall not create new fields. However, assigning to a nonexistent field (for example,  $\$({NF+2})=5$ ) shall increase the value of **NF**; create any intervening fields with the uninitialized value; and cause the value of **\$0** to be recomputed, with the fields being separated by the value of **OFS**. Each field variable shall have a string value or an uninitialized value when created. Field variables shall have the uninitialized value when created from **\$0** using **FS** and the variable does not contain any characters. If appropriate, the field variable shall be considered a numeric string (see **Expressions in awk** (on page 156)).

6161

6162

6163

6164

6165

6166

6167

6168

6169

6170

6171

6172

Implementations shall support the following other special variables that are set by *awk*:

6173

**ARGC** The number of elements in the **ARGV** array.

6174

**ARGV** An array of command line arguments, excluding options and the *program* argument, numbered from zero to **ARGC**-1.

6175

The arguments in **ARGV** can be modified or added to; **ARGC** can be altered. As each input file ends, *awk* shall treat the next non-null element of **ARGV**, up to the current value of **ARGC**-1, inclusive, as the name of the next input file. Thus, setting an element of **ARGV** to null means that it shall not be treated as an input file. The name '-' indicates the standard input. If an argument matches the format of an *assignment* operand, this argument shall be treated as an *assignment* rather than a *file* argument.

6176

6177

6178

6179

6180

6181

6182

**CONVFMT** The **printf** format for converting numbers to strings (except for output statements, where **OFMT** is used); "% .6g" by default.

6183

6184

**ENVIRON** An array representing the value of the environment, as described in the **exec** functions defined in the System Interfaces volume of IEEE Std 1003.1-2001. The indices of the array shall be strings consisting of the names of the environment variables, and the value of each array element shall be a string consisting of the value of that variable. If appropriate, the environment variable shall be considered a *numeric string* (see **Expressions in awk** (on page 156)); the array element shall also have its numeric value.

6185

6186

6187

6188

6189

6190

6191

In all cases where the behavior of *awk* is affected by environment variables (including the environment of any commands that *awk* executes via the **system** function or via pipeline redirections with the **print** statement, the **printf** statement, or the **getline** function), the environment used shall be the environment at the time

6192

6193

6194

6195

|      |                 |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
|------|-----------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 6196 |                 | awk began executing; it is implementation-defined whether any modification of ENVIRON affects this environment.                                                                                                                                                                                                                                                                                                                                                                                                          |
| 6197 |                 |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| 6198 | <b>FILENAME</b> | A pathname of the current input file. Inside a <b>BEGIN</b> action the value is undefined. Inside an <b>END</b> action the value shall be the name of the last input file processed.                                                                                                                                                                                                                                                                                                                                     |
| 6199 |                 |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| 6200 |                 |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| 6201 | <b>FNR</b>      | The ordinal number of the current record in the current file. Inside a <b>BEGIN</b> action the value shall be zero. Inside an <b>END</b> action the value shall be the number of the last record processed in the last file processed.                                                                                                                                                                                                                                                                                   |
| 6202 |                 |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| 6203 |                 |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| 6204 | <b>FS</b>       | Input field separator regular expression; a <space> by default.                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| 6205 |                 |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| 6206 | <b>NF</b>       | The number of fields in the current record. Inside a <b>BEGIN</b> action, the use of <b>NF</b> is undefined unless a <b>getline</b> function without a <i>var</i> argument is executed previously. Inside an <b>END</b> action, <b>NF</b> shall retain the value it had for the last record read, unless a subsequent, redirected, <b>getline</b> function without a <i>var</i> argument is performed prior to entering the <b>END</b> action.                                                                           |
| 6207 |                 |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| 6208 |                 |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| 6209 |                 |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| 6210 | <b>NR</b>       | The ordinal number of the current record from the start of input. Inside a <b>BEGIN</b> action the value shall be zero. Inside an <b>END</b> action the value shall be the number of the last record processed.                                                                                                                                                                                                                                                                                                          |
| 6211 |                 |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| 6212 |                 |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| 6213 | <b>OFMT</b>     | The <b>printf</b> format for converting numbers to strings in output statements (see <b>Output Statements</b> (on page 165)); "%.6g" by default. The result of the conversion is unspecified if the value of <b>OFMT</b> is not a floating-point format specification.                                                                                                                                                                                                                                                   |
| 6214 |                 |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| 6215 |                 |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| 6216 |                 |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| 6217 | <b>OFS</b>      | The <b>print</b> statement output field separation; <space> by default.                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| 6218 |                 |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| 6219 | <b>ORS</b>      | The <b>print</b> statement output record separator; a <newline> by default.                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| 6220 |                 |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| 6221 | <b>RLENGTH</b>  | The length of the string matched by the <b>match</b> function.                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| 6222 |                 |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| 6223 | <b>RS</b>       | The first character of the string value of <b>RS</b> shall be the input record separator; a <newline> by default. If <b>RS</b> contains more than one character, the results are unspecified. If <b>RS</b> is null, then records are separated by sequences consisting of a <newline> plus one or more blank lines, leading or trailing blank lines shall not result in empty records at the beginning or end of the input, and a <newline> shall always be a field separator, no matter what the value of <b>FS</b> is. |
| 6224 |                 |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| 6225 |                 |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| 6226 | <b>RSTART</b>   | The starting position of the string matched by the <b>match</b> function, numbering from 1. This shall always be equivalent to the return value of the <b>match</b> function.                                                                                                                                                                                                                                                                                                                                            |
| 6227 |                 |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| 6228 | <b>SUBSEP</b>   | The subscript separator string for multi-dimensional arrays; the default value is implementation-defined.                                                                                                                                                                                                                                                                                                                                                                                                                |
| 6229 |                 |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |

## 6230 Regular Expressions

6231 The **awk** utility shall make use of the extended regular expression notation (see the Base  
6232 Definitions volume of IEEE Std 1003.1-2001, Section 9.4, Extended Regular Expressions) except  
6233 that it shall allow the use of C-language conventions for escaping special characters within the  
6234 EREs, as specified in the table in the Base Definitions volume of IEEE Std 1003.1-2001, Chapter 5,  
6235 File Format Notation ('\\', '\a', '\b', '\f', '\n', '\r', '\t', '\v') and the following  
6236 table; these escape sequences shall be recognized both inside and outside bracket expressions.  
6237 Note that records need not be separated by <newline>s and string constants can contain  
6238 <newline>s, so even the "\n" sequence is valid in awk EREs. Using a slash character within an  
6239 ERE requires the escaping shown in the following table.

6240

**Table 4-2** Escape Sequences in awk

| Escape Sequence | Description                                                                                                                                                                                                                           | Meaning                                                                                                                                                                                                                 |
|-----------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| \"              | Backslash quotation-mark                                                                                                                                                                                                              | Quotation-mark character                                                                                                                                                                                                |
| \/              | Backslash slash                                                                                                                                                                                                                       | Slash character                                                                                                                                                                                                         |
| \ddd            | A backslash character followed by the longest sequence of one, two, or three octal-digit characters (01234567). If all of the digits are 0 (that is, representation of the NUL character), the behavior is undefined.                 | The character whose encoding is represented by the one, two, or three-digit octal integer. Multi-byte characters require multiple, concatenated escape sequences of this type, including the leading '\' for each byte. |
| \c              | A backslash character followed by any character not described in this table or in the table in the Base Definitions volume of IEEE Std 1003.1-2001, Chapter 5, File Format Notation ('\\', '\a', '\b', '\f', '\n', '\r', '\t', '\v'). | Undefined                                                                                                                                                                                                               |

6259 A regular expression can be matched against a specific field or string by using one of the two  
 6260 regular expression matching operators, '^' and "!.~". These operators shall interpret their  
 6261 right-hand operand as a regular expression and their left-hand operand as a string. If the regular  
 6262 expression matches the string, the '^' expression shall evaluate to a value of 1, and the "!.~"  
 6263 expression shall evaluate to a value of 0. (The regular expression matching operation is as  
 6264 defined by the term matched in the Base Definitions volume of IEEE Std 1003.1-2001, Section 9.1,  
 6265 Regular Expression Definitions, where a match occurs on any part of the string unless the  
 6266 regular expression is limited with the circumflex or dollar sign special characters.) If the regular  
 6267 expression does not match the string, the '^' expression shall evaluate to a value of 0, and the  
 6268 "!.~" expression shall evaluate to a value of 1. If the right-hand operand is any expression other  
 6269 than the lexical token ERE, the string value of the expression shall be interpreted as an extended  
 6270 regular expression, including the escape conventions described above. Note that these same  
 6271 escape conventions shall also be applied in determining the value of a string literal (the lexical  
 6272 token STRING), and thus shall be applied a second time when a string literal is used in this  
 6273 context.

6274 When an ERE token appears as an expression in any context other than as the right-hand of the  
 6275 '^' or "!.~" operator or as one of the built-in function arguments described below, the value of  
 6276 the resulting expression shall be the equivalent of:

6277 \$0 ~ /ere/

6278 The ere argument to the **gsub**, **match**, **sub** functions, and the fs argument to the **split** function  
 6279 (see **String Functions** (on page 167)) shall be interpreted as extended regular expressions. These  
 6280 can be either ERE tokens or arbitrary expressions, and shall be interpreted in the same manner as  
 6281 the right-hand side of the '^' or "!.~" operator.

6282 An extended regular expression can be used to separate fields by using the -F ERE option or by  
 6283 assigning a string containing the expression to the built-in variable FS. The default value of the  
 6284 FS variable shall be a single <space>. The following describes FS behavior:

- 6285 1. If FS is a null string, the behavior is unspecified.

- 6286        2. If **FS** is a single character:
- 6287            a. If **FS** is <space>, skip leading and trailing <blank>s; fields shall be delimited by sets  
6288            of one or more <blank>s.
- 6290            b. Otherwise, if **FS** is any other character *c*, fields shall be delimited by each single  
6290            occurrence of *c*.
- 6291        3. Otherwise, the string value of **FS** shall be considered to be an extended regular expression.  
6292            Each occurrence of a sequence matching the extended regular expression shall delimit  
6293            fields.

6294        Except for the ' ~ ' and " ! ~ " operators, and in the **gsub**, **match**, **split**, and **sub** built-in functions,  
6295        ERE matching shall be based on input records; that is, record separator characters (the first  
6296        character of the value of the variable **RS**, <newline> by default) cannot be embedded in the  
6297        expression, and no expression shall match the record separator character. If the record separator  
6298        is not <newline>, <newline>s embedded in the expression can be matched. For the ' ~ ' and  
6299        " ! ~ " operators, and in those four built-in functions, ERE matching shall be based on text strings;  
6300        that is, any character (including <newline> and the record separator) can be embedded in the  
6301        pattern, and an appropriate pattern shall match any character. However, in all *awk* ERE  
6302        matching, the use of one or more NUL characters in the pattern, input record, or text string  
6303        produces undefined results.

#### 6304        Patterns

6305        A *pattern* is any valid *expression*, a range specified by two expressions separated by a comma, or  
6306        one of the two special patterns **BEGIN** or **END**.

#### 6307        Special Patterns

6308        The *awk* utility shall recognize two special patterns, **BEGIN** and **END**. Each **BEGIN** pattern  
6309        shall be matched once and its associated action executed before the first record of input is read  
6310        (except possibly by use of the **getline** function—see **Input/Output and General Functions** (on  
6311        page 168)—in a prior **BEGIN** action) and before command line assignment is done. Each **END**  
6312        pattern shall be matched once and its associated action executed after the last record of input has  
6313        been read. These two patterns shall have associated actions.

6314        **BEGIN** and **END** shall not combine with other patterns. Multiple **BEGIN** and **END** patterns  
6315        shall be allowed. The actions associated with the **BEGIN** patterns shall be executed in the order  
6316        specified in the program, as are the **END** actions. An **END** pattern can precede a **BEGIN** pattern  
6317        in a program.

6318        If an *awk* program consists of only actions with the pattern **BEGIN**, and the **BEGIN** action  
6319        contains no **getline** function, *awk* shall exit without reading its input when the last statement in  
6320        the last **BEGIN** action is executed. If an *awk* program consists of only actions with the pattern  
6321        **END** or only actions with the patterns **BEGIN** and **END**, the input shall be read before the  
6322        statements in the **END** actions are executed.

6323       **Expression Patterns**

6324       An expression pattern shall be evaluated as if it were an expression in a Boolean context. If the  
6325       result is true, the pattern shall be considered to match, and the associated action (if any) shall be  
6326       executed. If the result is false, the action shall not be executed.

6327       **Pattern Ranges**

6328       A pattern range consists of two expressions separated by a comma; in this case, the action shall  
6329       be performed for all records between a match of the first expression and the following match of  
6330       the second expression, inclusive. At this point, the pattern range can be repeated starting at  
6331       input records subsequent to the end of the matched range.

6332       **Actions**

6333       An action is a sequence of statements as shown in the grammar in **Grammar** (on page 170). Any  
6334       single statement can be replaced by a statement list enclosed in braces. The application shall  
6335       ensure that statements in a statement list are separated by <newline>s or semicolons. Statements  
6336       in a statement list shall be executed sequentially in the order that they appear.

6337       The **expression** acting as the conditional in an **if** statement shall be evaluated and if it is non-zero  
6338       or non-null, the following statement shall be executed; otherwise, if **else** is present, the statement  
6339       following the **else** shall be executed.

6340       The **if**, **while**, **do...while**, **for**, **break**, and **continue** statements are based on the ISO C standard  
6341       (see Section 1.7.2 (on page 7)), except that the Boolean expressions shall be treated as described  
6342       in **Expressions in awk** (on page 156), and except in the case of:

6343       `for (variable in array)`

6344       which shall iterate, assigning each **index** of **array** to **variable** in an unspecified order. The results of  
6345       adding new elements to **array** within such a **for** loop are undefined. If a **break** or **continue**  
6346       statement occurs outside of a loop, the behavior is undefined.

6347       The **delete** statement shall remove an individual array element. Thus, the following code deletes  
6348       an entire array:

6349       `for (index in array)  
6350            delete array[index]`

6351       The **next** statement shall cause all further processing of the current input record to be  
6352       abandoned. The behavior is undefined if a **next** statement appears or is invoked in a **BEGIN** or  
6353       **END** action.

6354       The **exit** statement shall invoke all **END** actions in the order in which they occur in the program  
6355       source and then terminate the program without reading further input. An **exit** statement inside  
6356       an **END** action shall terminate the program without further execution of **END** actions. If an  
6357       expression is specified in an **exit** statement, its numeric value shall be the exit status of **awk**,  
6358       unless subsequent errors are encountered or a subsequent **exit** statement with an expression is  
6359       executed.

6360      **Output Statements**

6361      Both **print** and **printf** statements shall write to standard output by default. The output shall be  
6362      written to the location specified by *output redirection* if one is supplied, as follows:

6363      > *expression*  
6364      >> *expression*  
6365      | *expression*

6366      In all cases, the *expression* shall be evaluated to produce a string that is used as a pathname into  
6367      which to write (for '>' or ">>") or as a command to be executed (for '|'). Using the first two  
6368      forms, if the file of that name is not currently open, it shall be opened, creating it if necessary and  
6369      using the first form, truncating the file. The output then shall be appended to the file. As long as  
6370      the file remains open, subsequent calls in which *expression* evaluates to the same string value  
6371      shall simply append output to the file. The file remains open until the **close** function (see  
6372      **Input/Output and General Functions** (on page 168)) is called with an expression that evaluates  
6373      to the same string value.

6374      The third form shall write output onto a stream piped to the input of a command. The stream  
6375      shall be created if no stream is currently open with the value of *expression* as its command name.  
6376      The stream created shall be equivalent to one created by a call to the *popen()* function defined in  
6377      the System Interfaces volume of IEEE Std 1003.1-2001 with the value of *expression* as the  
6378      *command* argument and a value of *w* as the *mode* argument. As long as the stream remains open,  
6379      subsequent calls in which *expression* evaluates to the same string value shall write output to the  
6380      existing stream. The stream shall remain open until the **close** function (see **Input/Output and**  
6381      **General Functions** (on page 168)) is called with an expression that evaluates to the same string  
6382      value. At that time, the stream shall be closed as if by a call to the *pclose()* function defined in  
6383      the System Interfaces volume of IEEE Std 1003.1-2001.

6384      As described in detail by the grammar in **Grammar** (on page 170), these output statements shall  
6385      take a comma-separated list of *expressions* referred to in the grammar by the non-terminal  
6386      symbols **expr\_list**, **print\_expr\_list**, or **print\_expr\_list\_opt**. This list is referred to here as the  
6387      *expression list*, and each member is referred to as an *expression argument*.

6388      The **print** statement shall write the value of each *expression argument* onto the indicated output  
6389      stream separated by the current output field separator (see variable **OFS** above), and terminated  
6390      by the output record separator (see variable **ORS** above). All *expression arguments* shall be  
6391      taken as strings, being converted if necessary; this conversion shall be as described in  
6392      **Expressions in awk** (on page 156), with the exception that the **printf** format in **OFMT** shall be  
6393      used instead of the value in **CONVFMT**. An empty *expression list* shall stand for the whole  
6394      input record (\$0).

6395      The **printf** statement shall produce output based on a notation similar to the File Format  
6396      Notation used to describe file formats in this volume of IEEE Std 1003.1-2001 (see the Base  
6397      Definitions volume of IEEE Std 1003.1-2001, Chapter 5, File Format Notation). Output shall be  
6398      produced as specified with the first *expression argument* as the string *format* and subsequent  
6399      *expression arguments* as the strings *arg1* to *argn*, inclusive, with the following exceptions:

- 6400      1. The *format* shall be an actual character string rather than a graphical representation.  
6401      Therefore, it cannot contain empty character positions. The <space> in the *format* string, in  
6402      any context other than a *flag* of a conversion specification, shall be treated as an ordinary  
6403      character that is copied to the output.
- 6404      2. If the character set contains a 'Δ' character and that character appears in the *format* string,  
6405      it shall be treated as an ordinary character that is copied to the output.

- 6406        3. The *escape sequences* beginning with a backslash character shall be treated as sequences of  
6407        ordinary characters that are copied to the output. Note that these same sequences shall be  
6408        interpreted lexically by *awk* when they appear in literal strings, but they shall not be  
6409        treated specially by the **printf** statement.
- 6410        4. A *field width* or *precision* can be specified as the '\*' character instead of a digit string. In  
6411        this case the next argument from the expression list shall be fetched and its numeric value  
6412        taken as the field width or precision.
- 6413        5. The implementation shall not precede or follow output from the d or u conversion  
6414        specifier characters with <blank>s not specified by the *format* string.
- 6415        6. The implementation shall not precede output from the o conversion specifier character  
6416        with leading zeros not specified by the *format* string.
- 6417        7. For the c conversion specifier character: if the argument has a numeric value, the character  
6418        whose encoding is that value shall be output. If the value is zero or is not the encoding of  
6419        any character in the character set, the behavior is undefined. If the argument does not have  
6420        a numeric value, the first character of the string value shall be output; if the string does not  
6421        contain any characters, the behavior is undefined.
- 6422        8. For each conversion specification that consumes an argument, the next expression  
6423        argument shall be evaluated. With the exception of the c conversion specifier character,  
6424        the value shall be converted (according to the rules specified in **Expressions in awk** (on  
6425        page 156)) to the appropriate type for the conversion specification.
- 6426        9. If there are insufficient expression arguments to satisfy all the conversion specifications in  
6427        the *format* string, the behavior is undefined.
- 6428        10. If any character sequence in the *format* string begins with a '%' character, but does not  
6429        form a valid conversion specification, the behavior is unspecified.

6430 Both **print** and **printf** can output at least {LINE\_MAX} bytes.

## 6431 Functions

6432 The *awk* language has a variety of built-in functions: arithmetic, string, input/output, and  
6433 general.

### 6434 Arithmetic Functions

6435 The arithmetic functions, except for **int**, shall be based on the ISO C standard (see Section 1.7.2  
6436 (on page 7)). The behavior is undefined in cases where the ISO C standard specifies that an error  
6437 be returned or that the behavior is undefined. Although the grammar (see **Grammar** (on page  
6438 170)) permits built-in functions to appear with no arguments or parentheses, unless the  
6439 argument or parentheses are indicated as optional in the following list (by displaying them  
6440 within the "[]" brackets), such use is undefined.

- 6441 **atan2(y,x)**      Return arctangent of y/x in radians in the range [-π,π].  
6442 **cos(x)**            Return cosine of x, where x is in radians.  
6443 **sin(x)**            Return sine of x, where x is in radians.  
6444 **exp(x)**            Return the exponential function of x.  
6445 **log(x)**            Return the natural logarithm of x.  
6446 **sqrt(x)**            Return the square root of x.

6447       **int(x)**      Return the argument truncated to an integer. Truncation shall be toward 0 when  
 6448            x>0.  
 6449       **rand()**     Return a random number *n*, such that  $0 \leq n < 1$ .  
 6450        **srand([expr])** Set the seed value for *rand* to *expr* or use the time of day if *expr* is omitted. The  
 6451            previous seed value shall be returned.

## 6452       String Functions

6453       The string functions in the following list shall be supported. Although the grammar (see  
 6454            **Grammar** (on page 170)) permits built-in functions to appear with no arguments or parentheses,  
 6455            unless the argument or parentheses are indicated as optional in the following list (by displaying  
 6456            them within the " [ ] " brackets), such use is undefined.

6457       **gsub(ere, repl[], in)**  
 6458            Behave like **sub** (see below), except that it shall replace all occurrences of the  
 6459            regular expression (like the *ed* utility global substitute) in \$0 or in the *in* argument,  
 6460            when specified.  
 6461       **index(s, t)**    Return the position, in characters, numbering from 1, in string *s* where string *t* first  
 6462            occurs, or zero if it does not occur at all.  
 6463       **length([s])**    Return the length, in characters, of its argument taken as a string, or of the whole  
 6464            record, \$0, if there is no argument.  
 6465       **match(s, ere)**   Return the position, in characters, numbering from 1, in string *s* where the  
 6466            extended regular expression *ere* occurs, or zero if it does not occur at all. RSTART  
 6467            shall be set to the starting position (which is the same as the returned value), zero  
 6468            if no match is found; RLENGTH shall be set to the length of the matched string, -1  
 6469            if no match is found.  
 6470       **split(s, a[, fs ])**  
 6471            Split the string *s* into array elements *a*[1], *a*[2], ..., *a*[*n*], and return *n*. All elements  
 6472            of the array shall be deleted before the split is performed. The separation shall be  
 6473            done with the ERE *fs* or with the field separator FS if *fs* is not given. Each array  
 6474            element shall have a string value when created and, if appropriate, the array  
 6475            element shall be considered a numeric string (see **Expressions in awk** (on page  
 6476            156)). The effect of a null string as the value of *fs* is unspecified.  
 6477       **sprintf(fmt, expr, expr, ...)**  
 6478            Format the expressions according to the **printf** format given by *fmt* and return the  
 6479            resulting string.  
 6480       **sub(ere, repl[], in ])**  
 6481            Substitute the string *repl* in place of the first instance of the extended regular  
 6482            expression *ERE* in string *in* and return the number of substitutions. An ampersand  
 6483            (' & ') appearing in the string *repl* shall be replaced by the string from *in* that  
 6484            matches the *ERE*. An ampersand preceded with a backslash ('\') shall be  
 6485            interpreted as the literal ampersand character. An occurrence of two consecutive  
 6486            backslashes shall be interpreted as just a single literal backslash character. Any  
 6487            other occurrence of a backslash (for example, preceding any other character) shall  
 6488            be treated as a literal backslash character. Note that if *repl* is a string literal (the  
 6489            lexical token **STRING**; see **Grammar** (on page 170)), the handling of the  
 6490            ampersand character occurs after any lexical processing, including any lexical  
 6491            backslash escape sequence processing. If *in* is specified and it is not an lvalue (see  
 6492            **Expressions in awk** (on page 156)), the behavior is undefined. If *in* is omitted, *awk*

|      |                                                                                                                                                                                                                                                                                                                                                                                                                                                         |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
|------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 6493 |                                                                                                                                                                                                                                                                                                                                                                                                                                                         | shall use the current record (\$0) in its place.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
| 6494 | <b>substr(s, m[, n])</b>                                                                                                                                                                                                                                                                                                                                                                                                                                | Return the at most <i>n</i> -character substring of <i>s</i> that begins at position <i>m</i> , numbering from 1. If <i>n</i> is omitted, or if <i>n</i> specifies more characters than are left in the string, the length of the substring shall be limited by the length of the string <i>s</i> .                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
| 6495 |                                                                                                                                                                                                                                                                                                                                                                                                                                                         |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| 6496 |                                                                                                                                                                                                                                                                                                                                                                                                                                                         |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| 6497 |                                                                                                                                                                                                                                                                                                                                                                                                                                                         |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| 6498 | <b>tolower(s)</b>                                                                                                                                                                                                                                                                                                                                                                                                                                       | Return a string based on the string <i>s</i> . Each character in <i>s</i> that is an uppercase letter specified to have a <b>tolower</b> mapping by the <i>LC_CTYPE</i> category of the current locale shall be replaced in the returned string by the lowercase letter specified by the mapping. Other characters in <i>s</i> shall be unchanged in the returned string.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
| 6499 |                                                                                                                                                                                                                                                                                                                                                                                                                                                         |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| 6500 |                                                                                                                                                                                                                                                                                                                                                                                                                                                         |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| 6501 |                                                                                                                                                                                                                                                                                                                                                                                                                                                         |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| 6502 | <b>toupper(s)</b>                                                                                                                                                                                                                                                                                                                                                                                                                                       | Return a string based on the string <i>s</i> . Each character in <i>s</i> that is a lowercase letter specified to have a <b>toupper</b> mapping by the <i>LC_CTYPE</i> category of the current locale is replaced in the returned string by the uppercase letter specified by the mapping. Other characters in <i>s</i> are unchanged in the returned string.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
| 6503 |                                                                                                                                                                                                                                                                                                                                                                                                                                                         |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| 6504 |                                                                                                                                                                                                                                                                                                                                                                                                                                                         |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| 6505 |                                                                                                                                                                                                                                                                                                                                                                                                                                                         |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| 6506 | All of the preceding functions that take <i>ERE</i> as a parameter expect a pattern or a string valued expression that is a regular expression as defined in <b>Regular Expressions</b> (on page 161).                                                                                                                                                                                                                                                  |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| 6507 |                                                                                                                                                                                                                                                                                                                                                                                                                                                         |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| 6508 | <b>Input/Output and General Functions</b>                                                                                                                                                                                                                                                                                                                                                                                                               |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| 6509 | The input/output and general functions are:                                                                                                                                                                                                                                                                                                                                                                                                             |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| 6510 | <b>close(expression)</b>                                                                                                                                                                                                                                                                                                                                                                                                                                | Close the file or pipe opened by a <b>print</b> or <b>printf</b> statement or a call to <b>getline</b> with the same string-valued <i>expression</i> . The limit on the number of open <i>expression</i> arguments is implementation-defined. If the close was successful, the function shall return zero; otherwise, it shall return non-zero.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| 6511 |                                                                                                                                                                                                                                                                                                                                                                                                                                                         |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| 6512 |                                                                                                                                                                                                                                                                                                                                                                                                                                                         |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| 6513 |                                                                                                                                                                                                                                                                                                                                                                                                                                                         |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| 6514 |                                                                                                                                                                                                                                                                                                                                                                                                                                                         |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| 6515 | <b>expression   getline [var]</b>                                                                                                                                                                                                                                                                                                                                                                                                                       | Read a record of input from a stream piped from the output of a command. The stream shall be created if no stream is currently open with the value of <i>expression</i> as its command name. The stream created shall be equivalent to one created by a call to the <i>popen()</i> function with the value of <i>expression</i> as the <i>command</i> argument and a value of <i>r</i> as the <i>mode</i> argument. As long as the stream remains open, subsequent calls in which <i>expression</i> evaluates to the same string value shall read subsequent records from the stream. The stream shall remain open until the <b>close</b> function is called with an expression that evaluates to the same string value. At that time, the stream shall be closed as if by a call to the <i>pclose()</i> function. If <i>var</i> is omitted, \$0 and <b>NF</b> shall be set; otherwise, <i>var</i> shall be set and, if appropriate, it shall be considered a numeric string (see <b>Expressions in awk</b> (on page 156)). |
| 6516 |                                                                                                                                                                                                                                                                                                                                                                                                                                                         |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| 6517 |                                                                                                                                                                                                                                                                                                                                                                                                                                                         |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| 6518 |                                                                                                                                                                                                                                                                                                                                                                                                                                                         |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| 6519 |                                                                                                                                                                                                                                                                                                                                                                                                                                                         |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| 6520 |                                                                                                                                                                                                                                                                                                                                                                                                                                                         |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| 6521 |                                                                                                                                                                                                                                                                                                                                                                                                                                                         |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| 6522 |                                                                                                                                                                                                                                                                                                                                                                                                                                                         |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| 6523 |                                                                                                                                                                                                                                                                                                                                                                                                                                                         |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| 6524 |                                                                                                                                                                                                                                                                                                                                                                                                                                                         |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| 6525 |                                                                                                                                                                                                                                                                                                                                                                                                                                                         |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| 6526 |                                                                                                                                                                                                                                                                                                                                                                                                                                                         |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| 6527 | The <b>getline</b> operator can form ambiguous constructs when there are unparenthesized operators (including concatenate) to the left of the '  ' (to the beginning of the expression containing <b>getline</b> ). In the context of the '\$' operator, '  ' shall behave as if it had a lower precedence than '\$'. The result of evaluating other operators is unspecified, and conforming applications shall parenthesize properly all such usages. |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| 6528 |                                                                                                                                                                                                                                                                                                                                                                                                                                                         |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| 6529 |                                                                                                                                                                                                                                                                                                                                                                                                                                                         |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| 6530 |                                                                                                                                                                                                                                                                                                                                                                                                                                                         |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| 6531 |                                                                                                                                                                                                                                                                                                                                                                                                                                                         |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| 6532 |                                                                                                                                                                                                                                                                                                                                                                                                                                                         |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| 6533 | <b>getline</b>                                                                                                                                                                                                                                                                                                                                                                                                                                          | Set \$0 to the next input record from the current input file. This form of <b>getline</b> shall set the <b>NF</b> , <b>NR</b> , and <b>FNR</b> variables.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
| 6534 |                                                                                                                                                                                                                                                                                                                                                                                                                                                         |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| 6535 | <b>getline var</b>                                                                                                                                                                                                                                                                                                                                                                                                                                      | Set variable <i>var</i> to the next input record from the current input file and, if appropriate, <i>var</i> shall be considered a numeric string (see <b>Expressions in awk</b> (on page 156)). This form of <b>getline</b> shall set the <b>FNR</b> and <b>NR</b> variables.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| 6536 |                                                                                                                                                                                                                                                                                                                                                                                                                                                         |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| 6537 |                                                                                                                                                                                                                                                                                                                                                                                                                                                         |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |

**getline [var] < expression**

6538 Read the next record of input from a named file. The *expression* shall be evaluated  
 6539 to produce a string that is used as a pathname. If the file of that name is not  
 6540 currently open, it shall be opened. As long as the stream remains open, subsequent  
 6541 calls in which *expression* evaluates to the same string value shall read subsequent  
 6542 records from the file. The file shall remain open until the **close** function is called  
 6543 with an expression that evaluates to the same string value. If var is omitted, \$0 and  
 6544 NF shall be set; otherwise, var shall be set and, if appropriate, it shall be considered  
 6545 a numeric string (see **Expressions in awk** (on page 156)).  
 6546

6547 The **getline** operator can form ambiguous constructs when there are  
 6548 unparenthesized binary operators (including concatenate) to the right of the '<'  
 6549 (up to the end of the expression containing the **getline**). The result of evaluating  
 6550 such a construct is unspecified, and conforming applications shall parenthesize  
 6551 properly all such usages.

**system(expression)**

6552 Execute the command given by *expression* in a manner equivalent to the **system()**  
 6553 function defined in the System Interfaces volume of IEEE Std 1003.1-2001 and  
 6554 return the exit status of the command.  
 6555

6556 All forms of **getline** shall return 1 for successful input, zero for end-of-file, and -1 for an error.

6557 Where strings are used as the name of a file or pipeline, the application shall ensure that the  
 6558 strings are textually identical. The terminology "same string value" implies that "equivalent  
 6559 strings", even those that differ only by <space>s, represent different files.

**User-Defined Functions**

6561 The *awk* language also provides user-defined functions. Such functions can be defined as:

```
6562 function name([parameter, . . .]) { statements }
```

6563 A function can be referred to anywhere in an *awk* program; in particular, its use can precede its  
 6564 definition. The scope of a function is global.

6565 Function parameters, if present, can be either scalars or arrays; the behavior is undefined if an  
 6566 array name is passed as a parameter that the function uses as a scalar, or if a scalar expression is  
 6567 passed as a parameter that the function uses as an array. Function parameters shall be passed by  
 6568 value if scalar and by reference if array name.

6569 The number of parameters in the function definition need not match the number of parameters  
 6570 in the function call. Excess formal parameters can be used as local variables. If fewer arguments  
 6571 are supplied in a function call than are in the function definition, the extra parameters that are  
 6572 used in the function body as scalars shall evaluate to the uninitialized value until they are  
 6573 otherwise initialized, and the extra parameters that are used in the function body as arrays shall  
 6574 be treated as uninitialized arrays where each element evaluates to the uninitialized value until  
 6575 otherwise initialized.

6576 When invoking a function, no white space can be placed between the function name and the  
 6577 opening parenthesis. Function calls can be nested and recursive calls can be made upon  
 6578 functions. Upon return from any nested or recursive function call, the values of all of the calling  
 6579 function's parameters shall be unchanged, except for array parameters passed by reference. The  
 6580 **return** statement can be used to return a value. If a **return** statement appears outside of a  
 6581 function definition, the behavior is undefined.

6582 In the function definition, <newline>s shall be optional before the opening brace and after the  
 6583 closing brace. Function definitions can appear anywhere in the program where a *pattern-action*

6584 pair is allowed.

### 6585 Grammar

6586 The grammar in this section and the lexical conventions in the following section shall together  
 6587 describe the syntax for *awk* programs. The general conventions for this style of grammar are  
 6588 described in Section 1.10 (on page 19). A valid program can be represented as the non-terminal  
 6589 symbol *program* in the grammar. This formal syntax shall take precedence over the preceding  
 6590 text syntax description.

```

6591 %token NAME NUMBER STRING ERE
6592 %token FUNC_NAME /* Name followed by '()' without white space. */
6593 /* Keywords */
6594 %token Begin End
6595 /* 'BEGIN' 'END' */
6596 %token Break Continue Delete Do Else
6597 /* 'break' 'continue' 'delete' 'do' 'else' */
6598 %token Exit For Function If In
6599 /* 'exit' 'for' 'function' 'if' 'in' */
6600 %token Next Print Printf Return While
6601 /* 'next' 'print' 'printf' 'return' 'while' */
6602 /* Reserved function names */
6603 %token BUILTIN_FUNC_NAME
6604 /* One token for the following:
 * atan2 cos sin exp log sqrt int rand srand
 * gsub index length match split sprintf sub
 * substr tolower toupper close system
 */
6605 %token GETLINE
6606 /* Syntactically different from other built-ins. */
6607
6608 /* Two-character tokens. */
6609 %token ADD_ASSIGN SUB_ASSIGN MUL_ASSIGN DIV_ASSIGN MOD_ASSIGN POW_ASSIGN
6610 /* '+=' '-=' '*=' '/=' '%=' '^=' */
6611 %token OR AND NO_MATCH EQ LE GE NE INCR DECR APPEND
6612 /* '| '| '&&' '!~' '===' '<=' '>=' '!=' '++' '--' '>>' */
6613
6614 /* One-character tokens. */
6615 %token '{' '}' '(' ')' '[' ']' ',' ';' NEWLINE
6616 %token '+' '-' '*' '%' '^' '!' '>' '<' '|' '?' ':' '^' '$' '='
6617
6618 %start program
6619 %%
```

```

6620
6621 program : item_list
6622 | actionless_item_list
6623 ;
6624
6625 item_list : newline_opt
6626 | actionless_item_list item_terminator
6627 | item_list item_terminator
6628 | item_list action_terminator
6629 ;
```

```

6629 actionless_item_list : item_list pattern terminator
6630 | actionless_item_list pattern terminator
6631 ;
6632 item : pattern action
6633 | Function NAME '(' param_list_opt ')'
6634 newline_opt action
6635 | Function FUNC_NAME '(' param_list_opt ')'
6636 newline_opt action
6637 ;
6638 param_list_opt : /* empty */
6639 | param_list
6640 ;
6641 param_list : NAME
6642 | param_list ',' NAME
6643 ;
6644 pattern : Begin
6645 | End
6646 | expr
6647 | expr ',' newline_opt expr
6648 ;
6649 action : '{' newline_opt '}'
6650 | '{' newline_opt terminated_statement_list '}'
6651 | '{' newline_opt unterminated_statement_list '}'
6652 ;
6653 terminator : terminator ';' '
6654 | terminator NEWLINE
6655 |
6656 | ';' '
6657 ;
6658 terminated_statement_list : terminated_statement
6659 | terminated_statement_list terminated_statement
6660 ;
6661 unterminated_statement_list : unterminated_statement
6662 | terminated_statement_list unterminated_statement
6663 ;
6664 terminated_statement : action newline_opt
6665 | If '(' expr ')' newline_opt terminated_statement
6666 | If '(' expr ')' newline_opt terminated_statement
6667 Else newline_opt terminated_statement
6668 | While '(' expr ')' newline_opt terminated_statement
6669 | For '(' simple_statement_opt ';' '
6670 expr_opt ';' simple_statement_opt ')' newline_opt
6671 terminated_statement
6672 | For '(' NAME In NAME ')' newline_opt
6673 terminated_statement
6674 | ';' newline_opt
6675 | terminatable_statement NEWLINE newline_opt
6676 | terminatable_statement ';' newline_opt

```

```
6677 ;
6678 unterminated_statement : terminatable_statement
6679 | If '(' expr ')' newline_opt unterminated_statement
6680 | If '(' expr ')' newline_opt terminated_statement
6681 Else newline_opt unterminated_statement
6682 | While '(' expr ')' newline_opt unterminated_statement
6683 | For '(' simple_statement_opt ';' '
6684 expr_opt ';' simple_statement_opt ')' newline_opt
6685 unterminated_statement
6686 | For '(' NAME In NAME ')' newline_opt
6687 unterminated_statement
6688 ;
6689 terminatable_statement : simple_statement
6690 | Break
6691 | Continue
6692 | Next
6693 | Exit expr_opt
6694 | Return expr_opt
6695 | Do newline_opt terminated_statement While '(' expr ')'
6696 ;
6697 simple_statement_opt : /* empty */
6698 | simple_statement
6699 ;
6700 simple_statement : Delete NAME '[' expr_list ']'
6701 | expr
6702 | print_statement
6703 ;
6704 print_statement : simple_print_statement
6705 | simple_print_statement output_redirection
6706 ;
6707 simple_print_statement : Print print_expr_list_opt
6708 | Print '(' multiple_expr_list ')'
6709 | Printf print_expr_list
6710 | Printf '(' multiple_expr_list ')'
6711 ;
6712 output_redirection : '>' expr
6713 | APPEND expr
6714 | '| ' expr
6715 ;
6716 expr_list_opt : /* empty */
6717 | expr_list
6718 ;
6719 expr_list : expr
6720 | multiple_expr_list
6721 ;
6722 multiple_expr_list : expr ',' newline_opt expr
6723 | multiple_expr_list ',' newline_opt expr
```

```

6724 ;
6725 expr_opt : /* empty */
6726 | expr
6727 ;
6728 expr : unary_expr
6729 | non_unary_expr
6730 ;
6731 unary_expr : '+' expr
6732 | '-' expr
6733 | unary_expr '^' expr
6734 | unary_expr '*' expr
6735 | unary_expr '/' expr
6736 | unary_expr '%' expr
6737 | unary_expr '+' expr
6738 | unary_expr '-' expr
6739 | unary_expr non_unary_expr
6740 | unary_expr '<' expr
6741 | unary_expr LE expr
6742 | unary_expr NE expr
6743 | unary_expr EQ expr
6744 | unary_expr '>' expr
6745 | unary_expr GE expr
6746 | unary_expr '^' expr
6747 | unary_expr NO_MATCH expr
6748 | unary_expr In NAME
6749 | unary_expr AND newline_opt expr
6750 | unary_expr OR newline_opt expr
6751 | unary_expr '?' expr ':' expr
6752 | unary_input_function
6753 ;
6754 non_unary_expr : '(' expr ')'
6755 | '!' expr
6756 | non_unary_expr '^' expr
6757 | non_unary_expr '*' expr
6758 | non_unary_expr '/' expr
6759 | non_unary_expr '%' expr
6760 | non_unary_expr '+' expr
6761 | non_unary_expr '-' expr
6762 | non_unary_expr non_unary_expr
6763 | non_unary_expr '<' expr
6764 | non_unary_expr LE expr
6765 | non_unary_expr NE expr
6766 | non_unary_expr EQ expr
6767 | non_unary_expr '>' expr
6768 | non_unary_expr GE expr
6769 | non_unary_expr '^' expr
6770 | non_unary_expr NO_MATCH expr
6771 | non_unary_expr In NAME
6772 | '(' multiple_expr_list ')' In NAME
6773 | non_unary_expr AND newline_opt expr

```

```
6774 | non_unary_expr OR newline_opt expr
6775 | non_unary_expr '?' expr ':' expr
6776 | NUMBER
6777 | STRING
6778 | lvalue
6779 | ERE
6780 | lvalue INCR
6781 | lvalue DECR
6782 | INCR lvalue
6783 | DECR lvalue
6784 | lvalue POW_ASSIGN expr
6785 | lvalue MOD_ASSIGN expr
6786 | lvalue MUL_ASSIGN expr
6787 | lvalue DIV_ASSIGN expr
6788 | lvalue ADD_ASSIGN expr
6789 | lvalue SUB_ASSIGN expr
6790 | lvalue '=' expr
6791 | FUNC_NAME '(' expr_list_opt ')'
6792 /* no white space allowed before '(' */
6793 | BUILTIN_FUNC_NAME '(' expr_list_opt ')'
6794 | BUILTIN_FUNC_NAME
6795 | non_unary_input_function
6796 ;
6797 print_expr_list_opt : /* empty */
6798 | print_expr_list
6799 ;
6800 print_expr_list : print_expr
6801 | print_expr_list ',' newline_opt print_expr
6802 ;
6803 print_expr : unary_print_expr
6804 | non_unary_print_expr
6805 ;
6806 unary_print_expr : '+' print_expr
6807 | '-' print_expr
6808 | unary_print_expr '^' print_expr
6809 | unary_print_expr '*' print_expr
6810 | unary_print_expr '/' print_expr
6811 | unary_print_expr '%' print_expr
6812 | unary_print_expr '+' print_expr
6813 | unary_print_expr '-' print_expr
6814 | unary_print_expr non_unary_print_expr
6815 | unary_print_expr '~~' print_expr
6816 | unary_print_expr NO_MATCH print_expr
6817 | unary_print_expr In NAME
6818 | unary_print_expr AND newline_opt print_expr
6819 | unary_print_expr OR newline_opt print_expr
6820 | unary_print_expr '?' print_expr ':' print_expr
6821 ;
6822 non_unary_print_expr : '(' expr ')'
6823 | '!' print_expr
```

```

6824 | non_unary_print_expr '^' print_expr
6825 | non_unary_print_expr '*' print_expr
6826 | non_unary_print_expr '/' print_expr
6827 | non_unary_print_expr '%' print_expr
6828 | non_unary_print_expr '+' print_expr
6829 | non_unary_print_expr '-' print_expr
6830 | non_unary_print_expr non_unary_print_expr
6831 | non_unary_print_expr '~~' print_expr
6832 | non_unary_print_expr NO_MATCH print_expr
6833 | non_unary_print_expr In NAME
6834 | (' multiple_expr_list ') In NAME
6835 | non_unary_print_expr AND newline_opt print_expr
6836 | non_unary_print_expr OR newline_opt print_expr
6837 | non_unary_print_expr '?' print_expr ':' print_expr
6838 | NUMBER
6839 | STRING
6840 | lvalue
6841 | ERE
6842 | lvalue INCR
6843 | lvalue DECR
6844 | INCR lvalue
6845 | DECR lvalue
6846 | lvalue POW_ASSIGN print_expr
6847 | lvalue MOD_ASSIGN print_expr
6848 | lvalue MUL_ASSIGN print_expr
6849 | lvalue DIV_ASSIGN print_expr
6850 | lvalue ADD_ASSIGN print_expr
6851 | lvalue SUB_ASSIGN print_expr
6852 | lvalue '=' print_expr
6853 | FUNC_NAME '(' expr_list_opt ')'
6854 /* no white space allowed before '(' */
6855 | BUILTIN_FUNC_NAME '(' expr_list_opt ')'
6856 | BUILTIN_FUNC_NAME
6857 ;
6858 lvalue : NAME
6859 | NAME '[' expr_list ']'
6860 | '$' expr
6861 ;
6862 non_unary_input_function : simple_get
6863 | simple_get '<' expr
6864 | non_unary_expr '| simple_get
6865 ;
6866 unary_input_function : unary_expr '| simple_get
6867 ;
6868 simple_get : GETLINE
6869 | GETLINE lvalue
6870 ;
6871 newline_opt : /* empty */
6872 | newline_opt NEWLINE
6873 ;

```

- 6874        This grammar has several ambiguities that shall be resolved as follows:
- 6875        • Operator precedence and associativity shall be as described in Table 4-1 (on page 156).
- 6876        • In case of ambiguity, an **else** shall be associated with the most immediately preceding **if** that  
6877        would satisfy the grammar.
- 6878        • In some contexts, a slash (' / ') that is used to surround an ERE could also be the division  
6879        operator. This shall be resolved in such a way that wherever the division operator could  
6880        appear, a slash is assumed to be the division operator. (There is no unary division operator.)
- 6881        One convention that might not be obvious from the formal grammar is where <newline>s are  
6882        acceptable. There are several obvious placements such as terminating a statement, and a  
6883        backslash can be used to escape <newline>s between any lexical tokens. In addition, <newline>s  
6884        without backslashes can follow a comma, an open brace, logical AND operator ("&&"), logical  
6885        OR operator ("||"), the **do** keyword, the **else** keyword, and the closing parenthesis of an **if**, **for**,  
6886        or **while** statement. For example:
- 6887        { print \$1,  
6888                \$2 }
- 6889        **Lexical Conventions**
- 6890        The lexical conventions for *awk* programs, with respect to the preceding grammar, shall be as  
6891        follows:
- 6892        1. Except as noted, *awk* shall recognize the longest possible token or delimiter beginning at a  
6893        given point.
  - 6894        2. A comment shall consist of any characters beginning with the number sign character and  
6895        terminated by, but excluding the next occurrence of, a <newline>. Comments shall have  
6896        no effect, except to delimit lexical tokens.
  - 6897        3. The <newline> shall be recognized as the token **NEWLINE**.
  - 6898        4. A backslash character immediately followed by a <newline> shall have no effect.
  - 6899        5. The token **STRING** shall represent a string constant. A string constant shall begin with the  
6900        character '\"'. Within a string constant, a backslash character shall be considered to begin  
6901        an escape sequence as specified in the table in the Base Definitions volume of  
6902        IEEE Std 1003.1-2001, Chapter 5, File Format Notation ('\\", '\a', '\b', '\f', '\n',  
6903        '\r', '\t', '\v'). In addition, the escape sequences in Table 4-2 (on page 162) shall be  
6904        recognized. A <newline> shall not occur within a string constant. A string constant shall be  
6905        terminated by the first unescaped occurrence of the character '\"' after the one that begins  
6906        the string constant. The value of the string shall be the sequence of all unescaped  
6907        characters and values of escape sequences between, but not including, the two delimiting  
6908        '\"' characters.
  - 6909        6. The token **ERE** represents an extended regular expression constant. An ERE constant shall  
6910        begin with the slash character. Within an ERE constant, a backslash character shall be  
6911        considered to begin an escape sequence as specified in the table in the Base Definitions  
6912        volume of IEEE Std 1003.1-2001, Chapter 5, File Format Notation. In addition, the escape  
6913        sequences in Table 4-2 (on page 162) shall be recognized. The application shall ensure that  
6914        a <newline> does not occur within an ERE constant. An ERE constant shall be terminated  
6915        by the first unescaped occurrence of the slash character after the one that begins the ERE  
6916        constant. The extended regular expression represented by the ERE constant shall be the  
6917        sequence of all unescaped characters and values of escape sequences between, but not  
6918        including, the two delimiting slash characters.

- 6919        7. A <blank> shall have no effect, except to delimit lexical tokens or within **STRING** or **ERE**  
 6920        tokens.
- 6921        8. The token **NUMBER** shall represent a numeric constant. Its form and numeric value shall  
 6922        be equivalent to either of the tokens **floating-constant** or **integer-constant** as specified by  
 6923        the ISO C standard, with the following exceptions:
- 6924        a. An integer constant cannot begin with 0x or include the hexadecimal digits 'a', 'b',  
 6925            'c', 'd', 'e', 'f', 'A', 'B', 'C', 'D', 'E', or 'F'.
- 6926        b. The value of an integer constant beginning with 0 shall be taken in decimal rather  
 6927            than octal.
- 6928        c. An integer constant cannot include a suffix ('u', 'U', 'l', or 'L').
- 6929        d. A floating constant cannot include a suffix ('f', 'F', 'l', or 'L').

6930        If the value is too large or too small to be representable (see Section 1.7.2 (on page 7)), the  
 6931        behavior is undefined.

- 6932        9. A sequence of underscores, digits, and alphabetics from the portable character set (see the  
 6933        Base Definitions volume of IEEE Std 1003.1-2001, Section 6.1, Portable Character Set),  
 6934        beginning with an underscore or alphabetic, shall be considered a word.
- 6935        10. The following words are keywords that shall be recognized as individual tokens; the name  
 6936        of the token is the same as the keyword:

|          |        |      |          |       |        |
|----------|--------|------|----------|-------|--------|
| BEGIN    | delete | END  | function | in    | printf |
| break    | do     | exit | getline  | next  | return |
| continue | else   | for  | if       | print | while  |

- 6940        11. The following words are names of built-in functions and shall be recognized as the token  
 6941        **BUILTIN\_FUNC\_NAME**:

|       |        |       |         |         |         |
|-------|--------|-------|---------|---------|---------|
| atan2 | gsub   | log   | split   | sub     | toupper |
| close | index  | match | sprintf | substr  |         |
| cos   | int    | rand  | sqrt    | system  |         |
| exp   | length | sin   | srand   | tolower |         |

6946        The above-listed keywords and names of built-in functions are considered reserved words.

- 6947        12. The token **NAME** shall consist of a word that is not a keyword or a name of a built-in  
 6948        function and is not followed immediately (without any delimiters) by the '(' character.
- 6949        13. The token **FUNC\_NAME** shall consist of a word that is not a keyword or a name of a  
 6950        built-in function, followed immediately (without any delimiters) by the '(' character. The  
 6951        ')' character shall not be included as part of the token.
- 6952        14. The following two-character sequences shall be recognized as the named tokens:

| Token Name | Sequence | Token Name | Sequence |
|------------|----------|------------|----------|
| ADD_ASSIGN | +=       | NO_MATCH   | !~       |
| SUB_ASSIGN | -=       | EQ         | ==       |
| MUL_ASSIGN | *=       | LE         | <=       |
| DIV_ASSIGN | /=       | GE         | >=       |
| MOD_ASSIGN | %=       | NE         | !=       |
| POW_ASSIGN | ^=       | INCR       | ++       |
| OR         |          | DECR       | --       |
| AND        | &&       | APPEND     | >>       |

6962        15. The following single characters shall be recognized as tokens whose names are the  
6963            character:

6964        <newline> { } ( ) [ ] , ; + - \* % ^ ! > < | ? : ~ \$ =

6965        There is a lexical ambiguity between the token **ERE** and the tokens **'/'** and **DIV\_ASSIGN**.  
6966        When an input sequence begins with a slash character in any syntactic context where the token  
6967            **'/'** or **DIV\_ASSIGN** could appear as the next token in a valid program, the longer of those two  
6968            tokens that can be recognized shall be recognized. In any other syntactic context where the token  
6969            **ERE** could appear as the next token in a valid program, the token **ERE** shall be recognized.

## 6970 EXIT STATUS

6971        The following exit values shall be returned:

6972        0 All input files were processed successfully.

6973        >0 An error occurred.

6974        The exit status can be altered within the program by using an **exit** expression.

## 6975 CONSEQUENCES OF ERRORS

6976        If any *file* operand is specified and the named file cannot be accessed, *awk* shall write a  
6977            diagnostic message to standard error and terminate without any further action.

6978        If the program specified by either the *program* operand or a *progfile* operand is not a valid *awk*  
6979            program (as specified in the EXTENDED DESCRIPTION section), the behavior is undefined.

## 6980 APPLICATION USAGE

6981        The **index**, **length**, **match**, and **substr** functions should not be confused with similar functions in  
6982            the ISO C standard; the *awk* versions deal with characters, while the ISO C standard deals with  
6983            bytes.

6984        Because the concatenation operation is represented by adjacent expressions rather than an  
6985            explicit operator, it is often necessary to use parentheses to enforce the proper evaluation  
6986            precedence.

## 6987 EXAMPLES

6988        The *awk* program specified in the command line is most easily specified within single-quotes (for  
6989            example, '*program*') for applications using *sh*, because *awk* programs commonly contain  
6990            characters that are special to the shell, including double-quotes. In the cases where an *awk*  
6991            program contains single-quote characters, it is usually easiest to specify most of the program as  
6992            strings within single-quotes concatenated by the shell with quoted single-quote characters. For  
6993            example:

6994        awk '''\'' { print "quote:", \$0 }'

6995        prints all lines from the standard input containing a single-quote character, prefixed with *quote*:

6996        The following are examples of simple *awk* programs:

6997        1. Write to the standard output all input lines for which field 3 is greater than 5:

6998            \$3 > 5

6999        2. Write every tenth line:

7000            (NR % 10) == 0

7001        3. Write any line with a substring matching the regular expression:

7002            /(G|D)(2[0-9][[:alpha:]])\*/

- 7003        4. Print any line with a substring containing a 'G' or 'D', followed by a sequence of digits  
 7004        and characters. This example uses character classes **digit** and **alpha** to match language-  
 7005        independent digit and alphabetic characters respectively:
- ```
7006 / (G|D) ([[[:digit:]][:alpha:]]*) /
```
- 7007 5. Write any line in which the second field matches the regular expression and the fourth
 7008 field does not:
- ```
7009 $2 ~ /xyz/ && $4 !~ /xyz/
```
- 7010        6. Write any line in which the second field contains a backslash:
- ```
7011 $2 ~ /\\\/
```
- 7012 7. Write any line in which the second field contains a backslash. Note that backslash escapes
 7013 are interpreted twice; once in lexical processing of the string and once in processing the
 7014 regular expression:
- ```
7015 $2 ~ "\\\\\"
```
- 7016        8. Write the second to the last and the last field in each line. Separate the fields by a colon:
- ```
7017 {OFS=":";print $(NF-1), $NF}
```
- 7018 9. Write the line number and number of fields in each line. The three strings representing the
 7019 line number, the colon, and the number of fields are concatenated and that string is written
 7020 to standard output:
- ```
7021 {print NR ":" NF}
```
- 7022        10. Write lines longer than 72 characters:
- ```
7023 length($0) > 72
```
- 7024 11. Write the first two fields in opposite order separated by **OFS**:
- ```
7025 { print $2, $1 }
```
- 7026        12. Same, with input fields separated by a comma or <space>s and <tab>s, or both:
- ```
7027 BEGIN { FS = "[ \t]*|[ \t]+"; }  
7028 { print $2, $1 }
```
- 7029 13. Add up the first column, print sum, and average:
- ```
7030 { s += $1 }
7031 END {print "sum is ", s, " average is", s/NR}
```
- 7032        14. Write fields in reverse order, one per line (many lines out for each line in):
- ```
7033 { for (i = NF; i > 0; --i) print $i }
```
- 7034 15. Write all lines between occurrences of the strings **start** and **stop**:
- ```
7035 /start/, /stop/
```
- 7036        16. Write all lines whose first field is different from the previous one:
- ```
7037 $1 != prev { print; prev = $1 }
```
- 7038 17. Simulate **echo**:
- ```
7039 BEGIN {
7040 for (i = 1; i < ARGC; ++i)
7041 printf("%s%s", ARGV[i], i==ARGC-1?"\n":")
```

7042                  }

7043    18. Write the path prefixes contained in the *PATH* environment variable, one per line:

```
7044 BEGIN {
7045 n = split (ENVIRON["PATH"], path, ":")
7046 for (i = 1; i <= n; ++i)
7047 print path[i]
7048 }
```

7049    19. If there is a file named **input** containing page headers of the form:

7050                Page #

7051                and a file named **program** that contains:

```
7052 /Page/ { $2 = n++; }
7053 { print }
```

7054                then the command line:

7055                awk -f program n=5 input

7056                prints the file **input**, filling in page numbers starting at 5.

#### 7057 RATIONALE

7058        This description is based on the new *awk*, “nawk”, (see the referenced *The AWK Programming*  
7059        *Language*), which introduced a number of new features to the historical *awk*:

- 7060        1. New keywords: **delete**, **do**, **function**, **return**
- 7061        2. New built-in functions: **atan2**, **close**, **cos**, **gsub**, **match**, **rand**, **sin**, **srand**, **sub**, **system**
- 7062        3. New predefined variables: **FNR**, **ARGC**, **ARGV**, **RSTART**, **RLENGTH**, **SUBSEP**
- 7063        4. New expression operators: ?, :, , ^
- 7064        5. The **FS** variable and the third argument to **split**, now treated as extended regular  
7065        expressions.
- 7066        6. The operator precedence, changed to more closely match the C language. Two examples  
7067        of code that operate differently are:

```
7068 while (n /= 10 > 1) ...
7069 if (!"wk" ~ /bwk/) ...
```

7070        Several features have been added based on newer implementations of *awk*:

- 7071        • Multiple instances of **-f** *progfile* are permitted.
- 7072        • The new option **-v** *assignment*.
- 7073        • The new predefined variable **ENVIRON**.
- 7074        • New built-in functions **toupper** and **tolower**.
- 7075        • More formatting capabilities are added to **printf** to match the ISO C standard.

7076        The overall *awk* syntax has always been based on the C language, with a few features from the  
7077        shell command language and other sources. Because of this, it is not completely compatible with  
7078        any other language, which has caused confusion for some users. It is not the intent of the  
7079        standard developers to address such issues. A few relatively minor changes toward making the  
7080        language more compatible with the ISO C standard were made; most of these changes are based  
7081        on similar changes in recent implementations, as described above. There remain several C-

language conventions that are not in *awk*. One of the notable ones is the comma operator, which is commonly used to specify multiple expressions in the C language **for** statement. Also, there are various places where *awk* is more restrictive than the C language regarding the type of expression that can be used in a given context. These limitations are due to the different features that the *awk* language does provide.

Regular expressions in *awk* have been extended somewhat from historical implementations to make them a pure superset of extended regular expressions, as defined by IEEE Std 1003.1-2001 (see the Base Definitions volume of IEEE Std 1003.1-2001, Section 9.4, Extended Regular Expressions). The main extensions are internationalization features and interval expressions. Historical implementations of *awk* have long supported backslash escape sequences as an extension to extended regular expressions, and this extension has been retained despite inconsistency with other utilities. The number of escape sequences recognized in both extended regular expressions and strings has varied (generally increasing with time) among implementations. The set specified by IEEE Std 1003.1-2001 includes most sequences known to be supported by popular implementations and by the ISO C standard. One sequence that is not supported is hexadecimal value escapes beginning with '\x'. This would allow values expressed in more than 9 bits to be used within *awk* as in the ISO C standard. However, because this syntax has a non-deterministic length, it does not permit the subsequent character to be a hexadecimal digit. This limitation can be dealt with in the C language by the use of lexical string concatenation. In the *awk* language, concatenation could also be a solution for strings, but not for extended regular expressions (either lexical ERE tokens or strings used dynamically as regular expressions). Because of this limitation, the feature has not been added to IEEE Std 1003.1-2001.

When a string variable is used in a context where an extended regular expression normally appears (where the lexical token ERE is used in the grammar) the string does not contain the literal slashes.

Some versions of *awk* allow the form:

```
func name(args, ...) { statements }
```

This has been deprecated by the authors of the language, who asked that it not be specified.

Historical implementations of *awk* produce an error if a **next** statement is executed in a **BEGIN** action, and cause *awk* to terminate if a **next** statement is executed in an **END** action. This behavior has not been documented, and it was not believed that it was necessary to standardize it.

The specification of conversions between string and numeric values is much more detailed than in the documentation of historical implementations or in the referenced *The AWK Programming Language*. Although most of the behavior is designed to be intuitive, the details are necessary to ensure compatible behavior from different implementations. This is especially important in relational expressions since the types of the operands determine whether a string or numeric comparison is performed. From the perspective of an application writer, it is usually sufficient to expect intuitive behavior and to force conversions (by adding zero or concatenating a null string) when the type of an expression does not obviously match what is needed. The intent has been to specify historical practice in almost all cases. The one exception is that, in historical implementations, variables and constants maintain both string and numeric values after their original value is converted by any use. This means that referencing a variable or constant can have unexpected side effects. For example, with historical implementations the following program:

```
{
 a = "+2"
 b = 2
```

```

7130 if (NR % 2)
7131 c = a + b
7132 if (a == b)
7133 print "numeric comparison"
7134 else
7135 print "string comparison"
7136 }
```

7137 would perform a numeric comparison (and output numeric comparison) for each odd-numbered line, but perform a string comparison (and output string comparison) for each even-numbered line. IEEE Std 1003.1-2001 ensures that comparisons will be numeric if necessary. With historical implementations, the following program:

```

7141 BEGIN {
7142 OFMT = "%e"
7143 print 3.14
7144 OFMT = "%f"
7145 print 3.14
7146 }
```

7147 would output "3.140000e+00" twice, because in the second **print** statement the constant  
7148 "3.14" would have a string value from the previous conversion. IEEE Std 1003.1-2001 requires  
7149 that the output of the second **print** statement be "3.14000". The behavior of historical  
7150 implementations was seen as too unintuitive and unpredictable.

7151 It was pointed out that with the rules contained in early drafts, the following script would print  
7152 nothing:

```

7153 BEGIN {
7154 y[1.5] = 1
7155 OFMT = "%e"
7156 print y[1.5]
7157 }
```

7158 Therefore, a new variable, **CONVFMT**, was introduced. The **OFMT** variable is now restricted to  
7159 affecting output conversions of numbers to strings and **CONVFMT** is used for internal  
7160 conversions, such as comparisons or array indexing. The default value is the same as that for  
7161 **OFMT**, so unless a program changes **CONVFMT** (which no historical program would do), it  
7162 will receive the historical behavior associated with internal string conversions.

7163 The POSIX *awk* lexical and syntactic conventions are specified more formally than in other  
7164 sources. Again the intent has been to specify historical practice. One convention that may not be  
7165 obvious from the formal grammar as in other verbal descriptions is where <newline>s are  
7166 acceptable. There are several obvious placements such as terminating a statement, and a  
7167 backslash can be used to escape <newline>s between any lexical tokens. In addition, <newline>s  
7168 without backslashes can follow a comma, an open brace, a logical AND operator ("&&"), a  
7169 logical OR operator ("| |"), the **do** keyword, the **else** keyword, and the closing parenthesis of an  
7170 **if**, **for**, or **while** statement. For example:

```

7171 { print $1,
7172 $2 }
```

7173 The requirement that *awk* add a trailing <newline> to the program argument text is to simplify  
7174 the grammar, making it match a text file in form. There is no way for an application or test suite  
7175 to determine whether a literal <newline> is added or whether *awk* simply acts as if it did.

7176 IEEE Std 1003.1-2001 requires several changes from historical implementations in order to  
7177 support internationalization. Probably the most subtle of these is the use of the decimal-point  
7178 character, defined by the *LC\_NUMERIC* category of the locale, in representations of floating-  
7179 point numbers. This locale-specific character is used in recognizing numeric input, in converting  
7180 between strings and numeric values, and in formatting output. However, regardless of locale,  
7181 the period character (the decimal-point character of the POSIX locale) is the decimal-point  
7182 character recognized in processing awk programs (including assignments in command line  
7183 arguments). This is essentially the same convention as the one used in the ISO C standard. The  
7184 difference is that the C language includes the *setlocale()* function, which permits an application  
7185 to modify its locale. Because of this capability, a C application begins executing with its locale  
7186 set to the C locale, and only executes in the environment-specified locale after an explicit call to  
7187 *setlocale()*. However, adding such an elaborate new feature to the awk language was seen as  
7188 inappropriate for IEEE Std 1003.1-2001. It is possible to execute an awk program explicitly in any  
7189 desired locale by setting the environment in the shell.

7190 The undefined behavior resulting from NULs in extended regular expressions allows future  
7191 extensions for the GNU gawk program to process binary data.

7192 The behavior in the case of invalid awk programs (including lexical, syntactic, and semantic  
7193 errors) is undefined because it was considered overly limiting on implementations to specify. In  
7194 most cases such errors can be expected to produce a diagnostic and a non-zero exit status.  
7195 However, some implementations may choose to extend the language in ways that make use of  
7196 certain invalid constructs. Other invalid constructs might be deemed worthy of a warning, but  
7197 otherwise cause some reasonable behavior. Still other constructs may be very difficult to detect  
7198 in some implementations. Also, different implementations might detect a given error during an  
7199 initial parsing of the program (before reading any input files) while others might detect it when  
7200 executing the program after reading some input. Implementors should be aware that diagnosing  
7201 errors as early as possible and producing useful diagnostics can ease debugging of applications,  
7202 and thus make an implementation more usable.

7203 The unspecified behavior from using multi-character RS values is to allow possible future  
7204 extensions based on extended regular expressions used for record separators. Historical  
7205 implementations take the first character of the string and ignore the others.

7206 Unspecified behavior when *split(string,array,<null>)* is used is to allow a proposed future  
7207 extension that would split up a string into an array of individual characters.

7208 In the context of the **getline** function, equally good arguments for different precedences of the |  
7209 and < operators can be made. Historical practice has been that:

7210 `getline < "a" "b"`

7211 is parsed as:

7212 `( getline < "a" ) "b"`

7213 although many would argue that the intent was that the file **ab** should be read. However:

7214 `getline < "x" + 1`

7215 parses as:

7216 `getline < ( "x" + 1 )`

7217 Similar problems occur with the | version of **getline**, particularly in combination with \$. For  
7218 example:

7219 `$"echo hi" | getline`

7220 (This situation is particularly problematic when used in a **print** statement, where the **|getline**  
7221 part might be a redirection of the **print**.)

7222 Since in most cases such constructs are not (or at least should not) be used (because they have a  
7223 natural ambiguity for which there is no conventional parsing), the meaning of these constructs  
7224 has been made explicitly unspecified. (The effect is that a conforming application that runs into  
7225 the problem must parenthesize to resolve the ambiguity.) There appeared to be few if any actual  
7226 uses of such constructs.

7227 Grammars can be written that would cause an error under these circumstances. Where  
7228 backwards-compatibility is not a large consideration, implementors may wish to use such  
7229 grammars.

7230 Some historical implementations have allowed some built-in functions to be called without an  
7231 argument list, the result being a default argument list chosen in some “reasonable” way. Use of  
7232 **length** as a synonym for **length(\$0)** is the only one of these forms that is thought to be widely  
7233 known or widely used; this particular form is documented in various places (for example, most  
7234 historical *awk* reference pages, although not in the referenced *The AWK Programming Language*)  
7235 as legitimate practice. With this exception, default argument lists have always been  
7236 undocumented and vaguely defined, and it is not at all clear how (or if) they should be  
7237 generalized to user-defined functions. They add no useful functionality and preclude possible  
7238 future extensions that might need to name functions without calling them. Not standardizing  
7239 them seems the simplest course. The standard developers considered that **length** merited special  
7240 treatment, however, since it has been documented in the past and sees possibly substantial use  
7241 in historical programs. Accordingly, this usage has been made legitimate, but Issue 5 removed  
7242 the obsolescent marking for XSI-conforming implementations and many otherwise conforming  
7243 applications depend on this feature.

7244 In **sub** and **gsub**, if *repl* is a string literal (the lexical token **STRING**), then two consecutive  
7245 backslash characters should be used in the string to ensure a single backslash will precede the  
7246 ampersand when the resultant string is passed to the function. (For example, to specify one  
7247 literal ampersand in the replacement string, use **gsub(ERE, "\\\&")**.)

7248 Historically the only special character in the *repl* argument of **sub** and **gsub** string functions was  
7249 the ampersand ('&') character and preceding it with the backslash character was used to turn  
7250 off its special meaning.

7251 The description in the ISO POSIX-2:1993 standard introduced behavior such that the backslash  
7252 character was another special character and it was unspecified whether there were any other  
7253 special characters. This description introduced several portability problems, some of which are  
7254 described below, and so it has been replaced with the more historical description. Some of the  
7255 problems include:

- 7256 • Historically, to create the replacement string, a script could use **gsub(ERE, "\\\&")**, but with  
7257 the ISO POSIX-2:1993 standard wording, it was necessary to use **gsub(ERE, "\\\\\\\&")**.  
7258 Backslash characters are doubled here because all string literals are subject to lexical analysis,  
7259 which would reduce each pair of backslash characters to a single backslash before being  
7260 passed to **gsub**.
- 7261 • Since it was unspecified what the special characters were, for portable scripts to guarantee  
7262 that characters are printed literally, each character had to be preceded with a backslash. (For  
7263 example, a portable script had to use **gsub(ERE, "\\\h\\\i")** to produce a replacement string  
7264 of "hi".)

7265 The description for comparisons in the ISO POSIX-2:1993 standard did not properly describe  
7266 historical practice because of the way numeric strings are compared as numbers. The current  
7267 rules cause the following code:

```
7268 if (0 == "000")
7269 print "strange, but true"
7270 else
7271 print "not true"
```

7272 to do a numeric comparison, causing the `if` to succeed. It should be intuitively obvious that this  
7273 is incorrect behavior, and indeed, no historical implementation of `awk` actually behaves this way.

7274 To fix this problem, the definition of *numeric string* was enhanced to include only those values  
7275 obtained from specific circumstances (mostly external sources) where it is not possible to  
7276 determine unambiguously whether the value is intended to be a string or a numeric.

7277 Variables that are assigned to a numeric string shall also be treated as a numeric string. (For  
7278 example, the notion of a numeric string can be propagated across assignments.) In comparisons,  
7279 all variables having the uninitialized value are to be treated as a numeric operand evaluating to  
7280 the numeric value zero.

7281 Uninitialized variables include all types of variables including scalars, array elements, and fields.  
7282 The definition of an uninitialized value in **Variables and Special Variables** (on page 160) is  
7283 necessary to describe the value placed on uninitialized variables and on fields that are valid (for  
7284 example, < \$NF) but have no characters in them and to describe how these variables are to be  
7285 used in comparisons. A valid field, such as `$1`, that has no characters in it can be obtained from  
7286 an input line of "\t\t" when `FS='\'\t'`. Historically, the comparison (`$1<10`) was done  
7287 numerically after evaluating `$1` to the value zero.

7288 The phrase "... also shall have the numeric value of the numeric string" was removed from  
7289 several sections of the ISO POSIX-2:1993 standard because is specifies an unnecessary  
7290 implementation detail. It is not necessary for IEEE Std 1003.1-2001 to specify that these objects be  
7291 assigned two different values. It is only necessary to specify that these objects may evaluate to  
7292 two different values depending on context.

7293 The description of numeric string processing is based on the behavior of the `atof()` function in  
7294 the ISO C standard. While it is not a requirement for an implementation to use this function,  
7295 many historical implementations of `awk` do. In the ISO C standard, floating-point constants use a  
7296 period as a decimal point character for the language itself, independent of the current locale, but  
7297 the `atof()` function and the associated `strtod()` function use the decimal point character of the  
7298 current locale when converting strings to numeric values. Similarly in `awk`, floating-point  
7299 constants in an `awk` script use a period independent of the locale, but input strings use the  
7300 decimal point character of the locale.

### 7301 FUTURE DIRECTIONS

7302 None.

### 7303 SEE ALSO

7304 Section 1.10 (on page 19), `grep`, `lex`, `sed`, the System Interfaces volume of IEEE Std 1003.1-2001,  
7305 `atof()`, `exec`, `popen()`, `setlocale()`, `strtod()`

### 7306 CHANGE HISTORY

7307 First released in Issue 2.

### 7308 Issue 5

7309 The FUTURE DIRECTIONS section is added.

### 7310 Issue 6

7311 The `awk` utility is aligned with the IEEE P1003.2b draft standard.

7312 The normative text is reworded to avoid use of the term "must" for application requirements.

7313  
7314  
7315

IEEE PASC Interpretation 1003.2 #211 is applied, adding the sentence “An occurrence of two consecutive backslashes shall be interpreted as just a single literal backslash character.” into the description of the **sub** string function.

**7316 NAME**

7317        basename — return non-directory portion of a pathname

**7318 SYNOPSIS**

7319        basename *string* [*suffix*]

**7320 DESCRIPTION**

7321        The *string* operand shall be treated as a pathname, as defined in the Base Definitions volume of  
7322        IEEE Std 1003.1-2001, Section 3.266, Pathname. The string *string* shall be converted to the  
7323        filename corresponding to the last pathname component in *string* and then the suffix string  
7324        *suffix*, if present, shall be removed. This shall be done by performing actions equivalent to the  
7325        following steps in order:

- 7326        1. If *string* is a null string, it is unspecified whether the resulting string is ‘.’ or a null string.  
7327        In either case, skip steps 2 through 6.
- 7328        2. If *string* is “//”, it is implementation-defined whether steps 3 to 6 are skipped or  
7329        processed.
- 7330        3. If *string* consists entirely of slash characters, *string* shall be set to a single slash character. In  
7331        this case, skip steps 4 to 6.
- 7332        4. If there are any trailing slash characters in *string*, they shall be removed.
- 7333        5. If there are any slash characters remaining in *string*, the prefix of *string* up to and including  
7334        the last slash character in *string* shall be removed.
- 7335        6. If the *suffix* operand is present, is not identical to the characters remaining in *string*, and is  
7336        identical to a suffix of the characters remaining in *string*, the suffix *suffix* shall be removed  
7337        from *string*. Otherwise, *string* is not modified by this step. It shall not be considered an  
7338        error if *suffix* is not found in *string*.

7339        The resulting string shall be written to standard output.

**7340 OPTIONS**

7341        None.

**7342 OPERANDS**

7343        The following operands shall be supported:

7344        *string*        A string.

7345        *suffix*      A string.

**7346 STDIN**

7347        Not used.

**7348 INPUT FILES**

7349        None.

**7350 ENVIRONMENT VARIABLES**

7351        The following environment variables shall affect the execution of *basename*:

7352        *LANG*        Provide a default value for the internationalization variables that are unset or null.  
7353        (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2,  
7354        Internationalization Variables for the precedence of internationalization variables  
7355        used to determine the values of locale categories.)

7356        *LC\_ALL*     If set to a non-empty string value, override the values of all the other  
7357        internationalization variables.

7358           ***LC\_CTYPE***   Determine the locale for the interpretation of sequences of bytes of text data as  
7359            characters (for example, single-byte as opposed to multi-byte characters in  
7360            arguments).

7361           ***LC\_MESSAGES***  
7362            Determine the locale that should be used to affect the format and contents of  
7363            diagnostic messages written to standard error.

7364    XSI      ***NLSPATH***   Determine the location of message catalogs for the processing of ***LC\_MESSAGES***.

7365   **ASYNCHRONOUS EVENTS**

7366           Default.

7367   **STDOUT**  
7368           The **basename** utility shall write a line to the standard output in the following format:  
7369           "**%s\n**", <resulting string>

7370   **STDERR**  
7371           The standard error shall be used only for diagnostic messages.

7372   **OUTPUT FILES**  
7373           None.

7374   **EXTENDED DESCRIPTION**  
7375           None.

7376   **EXIT STATUS**  
7377           The following exit values shall be returned:  
7378            0   Successful completion.  
7379            >0   An error occurred.

7380   **CONSEQUENCES OF ERRORS**  
7381           Default.

7382   **APPLICATION USAGE**  
7383           The definition of *pathname* specifies implementation-defined behavior for pathnames starting  
7384           with two slash characters. Therefore, applications shall not arbitrarily add slashes to the  
7385           beginning of a pathname unless they can ensure that there are more or less than two or are  
7386           prepared to deal with the implementation-defined consequences.

7387   **EXAMPLES**  
7388           If the string *string* is a valid pathname:  
7389            \$(basename "string")  
7390           produces a filename that could be used to open the file named by *string* in the directory returned  
7391           by:  
7392            \$(dirname "string")  
7393           If the string *string* is not a valid pathname, the same algorithm is used, but the result need not be  
7394           a valid filename. The **basename** utility is not expected to make any judgements about the validity  
7395           of *string* as a pathname; it just follows the specified algorithm to produce a result string.  
7396           The following shell script compiles **/usr/src/cmd/cat.c** and moves the output to a file named **cat**  
7397           in the current directory when invoked with the argument **/usr/src/cmd/cat** or with the argument  
7398           **/usr/src/cmd/cat.c**:

7399        c99 \$(dirname "\$1")/\$(basename "\$1" .c).c  
7400        mv a.out \$(basename "\$1" .c)

7401 **RATIONALE**

7402        The behaviors of *basename* and *dirname* have been coordinated so that when *string* is a valid  
7403        pathname:

7404        \$(basename "string")

7405        would be a valid filename for the file in the directory:

7406        \$(dirname "string")

7407        This would not work for the early proposal versions of these utilities due to the way it specified  
7408        handling of trailing slashes.

7409        Since the definition of *pathname* specifies implementation-defined behavior for pathnames  
7410        starting with two slash characters, this volume of IEEE Std 1003.1-2001 specifies similar  
7411        implementation-defined behavior for the *basename* and *dirname* utilities.

7412 **FUTURE DIRECTIONS**

7413        None.

7414 **SEE ALSO**

7415        Section 2.5 (on page 33), *dirname*

7416 **CHANGE HISTORY**

7417        First released in Issue 2.

7418 **Issue 6**

7419        IEEE PASC Interpretation 1003.2 #164 is applied.

7420        The normative text is reworded to avoid use of the term “must” for application requirements.

**7421 NAME**

7422       batch — schedule commands to be executed in a batch queue

**7423 SYNOPSIS**

7424 UP       *batch*

7425

**7426 DESCRIPTION**

7427       The *batch* utility shall read commands from standard input and schedule them for execution in a  
7428       batch queue. It shall be the equivalent of the command:

7429       at -q *b* -m now

7430       where queue *b* is a special *at* queue, specifically for batch jobs. Batch jobs shall be submitted to  
7431       the batch queue with no time constraints and shall be run by the system using algorithms, based  
7432       on unspecified factors, that may vary with each invocation of *batch*.

7433 XSI       Users shall be permitted to use *batch* if their name appears in the file **/usr/lib/cron/at.allow**. If  
7434       that file does not exist, the file **/usr/lib/cron/at.deny** shall be checked to determine whether the  
7435       user shall be denied access to *batch*. If neither file exists, only a process with the appropriate  
7436       privileges shall be allowed to submit a job. If only **at.deny** exists and is empty, global usage shall  
7437       be permitted. The **at.allow** and **at.deny** files shall consist of one user name per line.

**7438 OPTIONS**

7439       None.

**7440 OPERANDS**

7441       None.

**7442 STDIN**

7443       The standard input shall be a text file consisting of commands acceptable to the shell command  
7444       language described in Chapter 2 (on page 29).

**7445 INPUT FILES**

7446 XSI       The text files **/usr/lib/cron/at.allow** and **/usr/lib/cron/at.deny** shall contain zero or more user  
7447       names, one per line, of users who are, respectively, authorized or denied access to the *at* and  
7448       *batch* utilities.

**7449 ENVIRONMENT VARIABLES**

7450       The following environment variables shall affect the execution of *batch*:

7451       **LANG**       Provide a default value for the internationalization variables that are unset or null.  
7452               (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2,  
7453               Internationalization Variables for the precedence of internationalization variables  
7454               used to determine the values of locale categories.)

7455       **LC\_ALL**      If set to a non-empty string value, override the values of all the other  
7456               internationalization variables.

7457       **LC\_CTYPE**     Determine the locale for the interpretation of sequences of bytes of text data as  
7458               characters (for example, single-byte as opposed to multi-byte characters in  
7459               arguments and input files).

**7460 *LC\_MESSAGES***

7461               Determine the locale that should be used to affect the format and contents of  
7462               diagnostic messages written to standard error and informative messages written to  
7463               standard output.

7464       **LC\_TIME**     Determine the format and contents for date and time strings written by *batch*.

|      |                                                                                                                                                                    |                |                                                                                                                                                                                                                                                                                                                                                                                                                  |
|------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------|----------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 7465 | XSI                                                                                                                                                                | <b>NLSPATH</b> | Determine the location of message catalogs for the processing of <i>LC_MESSAGES</i> .                                                                                                                                                                                                                                                                                                                            |
| 7466 |                                                                                                                                                                    | <b>SHELL</b>   | Determine the name of a command interpreter to be used to invoke the at-job. If the variable is unset or null, <i>sh</i> shall be used. If it is set to a value other than a name for <i>sh</i> , the implementation shall do one of the following: use that shell; use <i>sh</i> ; use the login shell from the user database; any of the preceding accompanied by a warning diagnostic about which was chosen. |
| 7467 |                                                                                                                                                                    |                |                                                                                                                                                                                                                                                                                                                                                                                                                  |
| 7468 |                                                                                                                                                                    |                |                                                                                                                                                                                                                                                                                                                                                                                                                  |
| 7469 |                                                                                                                                                                    |                |                                                                                                                                                                                                                                                                                                                                                                                                                  |
| 7470 |                                                                                                                                                                    |                |                                                                                                                                                                                                                                                                                                                                                                                                                  |
| 7471 |                                                                                                                                                                    | <b>TZ</b>      | Determine the timezone. The job shall be submitted for execution at the time specified by <i>timespec</i> or <b>-t time</b> relative to the timezone specified by the <i>TZ</i> variable. If <i>timespec</i> specifies a timezone, it overrides <i>TZ</i> . If <i>timespec</i> does not specify a timezone and <i>TZ</i> is unset or null, an unspecified default timezone shall be used.                        |
| 7472 |                                                                                                                                                                    |                |                                                                                                                                                                                                                                                                                                                                                                                                                  |
| 7473 |                                                                                                                                                                    |                |                                                                                                                                                                                                                                                                                                                                                                                                                  |
| 7474 |                                                                                                                                                                    |                |                                                                                                                                                                                                                                                                                                                                                                                                                  |
| 7475 |                                                                                                                                                                    |                |                                                                                                                                                                                                                                                                                                                                                                                                                  |
| 7476 | <b>ASYNCHRONOUS EVENTS</b>                                                                                                                                         |                |                                                                                                                                                                                                                                                                                                                                                                                                                  |
| 7477 | Default.                                                                                                                                                           |                |                                                                                                                                                                                                                                                                                                                                                                                                                  |
| 7478 | <b>STDOUT</b>                                                                                                                                                      |                |                                                                                                                                                                                                                                                                                                                                                                                                                  |
| 7479 | When standard input is a terminal, prompts of unspecified format for each line of the user input described in the STDIN section may be written to standard output. |                |                                                                                                                                                                                                                                                                                                                                                                                                                  |
| 7480 |                                                                                                                                                                    |                |                                                                                                                                                                                                                                                                                                                                                                                                                  |
| 7481 | <b>STDERR</b>                                                                                                                                                      |                |                                                                                                                                                                                                                                                                                                                                                                                                                  |
| 7482 | The following shall be written to standard error when a job has been successfully submitted:                                                                       |                |                                                                                                                                                                                                                                                                                                                                                                                                                  |
| 7483 | "job %s at %s\n", <i>at_job_id</i> , < <i>date</i> >                                                                                                               |                |                                                                                                                                                                                                                                                                                                                                                                                                                  |
| 7484 | where <i>date</i> shall be equivalent in format to the output of:                                                                                                  |                |                                                                                                                                                                                                                                                                                                                                                                                                                  |
| 7485 | date +"%a %b %e %T %Y"                                                                                                                                             |                |                                                                                                                                                                                                                                                                                                                                                                                                                  |
| 7486 | The date and time written shall be adjusted so that they appear in the timezone of the user (as determined by the <i>TZ</i> variable).                             |                |                                                                                                                                                                                                                                                                                                                                                                                                                  |
| 7487 |                                                                                                                                                                    |                |                                                                                                                                                                                                                                                                                                                                                                                                                  |
| 7488 | Neither this, nor warning messages concerning the selection of the command interpreter, are considered a diagnostic that changes the exit status.                  |                |                                                                                                                                                                                                                                                                                                                                                                                                                  |
| 7489 |                                                                                                                                                                    |                |                                                                                                                                                                                                                                                                                                                                                                                                                  |
| 7490 | Diagnostic messages, if any, shall be written to standard error.                                                                                                   |                |                                                                                                                                                                                                                                                                                                                                                                                                                  |
| 7491 | <b>OUTPUT FILES</b>                                                                                                                                                |                |                                                                                                                                                                                                                                                                                                                                                                                                                  |
| 7492 | None.                                                                                                                                                              |                |                                                                                                                                                                                                                                                                                                                                                                                                                  |
| 7493 | <b>EXTENDED DESCRIPTION</b>                                                                                                                                        |                |                                                                                                                                                                                                                                                                                                                                                                                                                  |
| 7494 | None.                                                                                                                                                              |                |                                                                                                                                                                                                                                                                                                                                                                                                                  |
| 7495 | <b>EXIT STATUS</b>                                                                                                                                                 |                |                                                                                                                                                                                                                                                                                                                                                                                                                  |
| 7496 | The following exit values shall be returned:                                                                                                                       |                |                                                                                                                                                                                                                                                                                                                                                                                                                  |
| 7497 | 0 Successful completion.                                                                                                                                           |                |                                                                                                                                                                                                                                                                                                                                                                                                                  |
| 7498 | >0 An error occurred.                                                                                                                                              |                |                                                                                                                                                                                                                                                                                                                                                                                                                  |
| 7499 | <b>CONSEQUENCES OF ERRORS</b>                                                                                                                                      |                |                                                                                                                                                                                                                                                                                                                                                                                                                  |
| 7500 | The job shall not be scheduled.                                                                                                                                    |                |                                                                                                                                                                                                                                                                                                                                                                                                                  |

7501 **APPLICATION USAGE**

7502 It may be useful to redirect standard output within the specified commands.

7503 **EXAMPLES**

- 7504 1. This sequence can be used at a terminal:

```
7505 batch
7506 sort < file >outfile
7507 EOT
```

- 7508 2. This sequence, which demonstrates redirecting standard error to a pipe, is useful in a
7509 command procedure (the sequence of output redirection specifications is significant):

```
7510 batch <<
7511 ! diff file1 file2 2>&1 >outfile | mailx mygroup
7512 !
```

7513 **RATIONALE**

7514 Early proposals described *batch* in a manner totally separated from *at*, even though the historical
7515 model treated it almost as a synonym for *at -qb*. A number of features were added to list and
7516 control batch work separately from those in *at*. Upon further reflection, it was decided that the
7517 benefit of this did not merit the change to the historical interface.

7518 The **-m** option was included on the equivalent *at* command because it is historical practice to
7519 mail results to the submitter, even if all job-produced output is redirected. As explained in the
7520 RATIONALE for *at*, the **now** keyword submits the job for immediate execution (after scheduling
7521 delays), despite some historical systems where *at now* would have been considered an error.

7522 **FUTURE DIRECTIONS**

7523 None.

7524 **SEE ALSO**

7525 *at*

7526 **CHANGE HISTORY**

7527 First released in Issue 2.

7528 **Issue 6**

7529 This utility is marked as part of the User Portability Utilities option.

7530 The NAME is changed to align with the IEEE P1003.2b draft standard.

7531 The normative text is reworded to avoid use of the term “must” for application requirements.

**7532 NAME**

7533 bc — arbitrary-precision arithmetic language

**7534 SYNOPSIS**

7535 bc [-l] [*file* ...]

**7536 DESCRIPTION**

7537 The *bc* utility shall implement an arbitrary precision calculator. It shall take input from any files given, then read from the standard input. If the standard input and standard output to *bc* are attached to a terminal, the invocation of *bc* shall be considered to be *interactive*, causing behavioral constraints described in the following sections.

**7541 OPTIONS**

7542 The *bc* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section 12.2, Utility Syntax Guidelines.

7544 The following option shall be supported:

7545 -l (The letter ell.) Define the math functions and initialize *scale* to 20, instead of the  
7546 default zero; see the EXTENDED DESCRIPTION section.

**7547 OPERANDS**

7548 The following operand shall be supported:

7549 *file* A pathname of a text file containing *bc* program statements. After all *files* have  
7550 been read, *bc* shall read the standard input.

**7551 STDIN**

7552 See the INPUT FILES section.

**7553 INPUT FILES**

7554 Input files shall be text files containing a sequence of comments, statements, and function  
7555 definitions that shall be executed as they are read.

**7556 ENVIRONMENT VARIABLES**

7557 The following environment variables shall affect the execution of *bc*:

7558 *LANG* Provide a default value for the internationalization variables that are unset or null.  
7559 (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2,  
7560 Internationalization Variables for the precedence of internationalization variables  
7561 used to determine the values of locale categories.)

7562 *LC\_ALL* If set to a non-empty string value, override the values of all the other  
7563 internationalization variables.

7564 *LC\_CTYPE* Determine the locale for the interpretation of sequences of bytes of text data as  
7565 characters (for example, single-byte as opposed to multi-byte characters in  
7566 arguments and input files).

**7567 *LC\_MESSAGES***

7568 Determine the locale that should be used to affect the format and contents of  
7569 diagnostic messages written to standard error.

7570 XSI *NLSPATH* Determine the location of message catalogs for the processing of *LC\_MESSAGES*.

**7571 ASYNCHRONOUS EVENTS**

7572 Default.

7573 **STDOUT**

7574 The output of the *bc* utility shall be controlled by the program read, and consist of zero or more  
 7575 lines containing the value of all executed expressions without assignments. The radix and  
 7576 precision of the output shall be controlled by the values of the **obase** and **scale** variables; see the  
 7577 EXTENDED DESCRIPTION section.

7578 **STDERR**

7579 The standard error shall be used only for diagnostic messages.

7580 **OUTPUT FILES**

7581 None.

7582 **EXTENDED DESCRIPTION**7583 **Grammar**

7584 The grammar in this section and the lexical conventions in the following section shall together  
 7585 describe the syntax for *bc* programs. The general conventions for this style of grammar are  
 7586 described in Section 1.10 (on page 19). A valid program can be represented as the non-terminal  
 7587 symbol **program** in the grammar. This formal syntax shall take precedence over the text syntax  
 7588 description.

```

7589 %token EOF NEWLINE STRING LETTER NUMBER
7590 %token MUL_OP
7591 /* '*', '/', '%' */
7592 %token ASSIGN_OP
7593 /* '=' , '+=' , '-=' , '*=' , '/=' , '%=' , '^=' */
7594 %token REL_OP
7595 /* '==' , '<=' , '>=' , '!=' , '<' , '>' */
7596 %token INCR_DECR
7597 /* '++' , '--' */
7598 %token Define Break Quit Length
7599 /* 'define' , 'break' , 'quit' , 'length' */
7600 %token Return For If While Sqrt
7601 /* 'return' , 'for' , 'if' , 'while' , 'sqrt' */
7602 %token Scale Ibase Obase Auto
7603 /* 'scale' , 'ibase' , 'obase' , 'auto' */
7604 %start program
7605 %%
7606 program : EOF
7607 | input_item program
7608 ;
7609 input_item : semicolon_list NEWLINE
7610 | function
7611 ;
7612 semicolon_list : /* empty */
7613 | statement
7614 | semicolon_list ';' statement
7615 | semicolon_list ';'
```

```
7616 ;
7617 statement_list : /* empty */
7618 | statement
7619 | statement_list NEWLINE
7620 | statement_list NEWLINE statement
7621 | statement_list ';' ;
7622 | statement_list ';' statement
7623 ;
7624 statement : expression
7625 | STRING
7626 | Break
7627 | Quit
7628 | Return
7629 | Return '(' return_expression ')' ;
7630 | For '(' expression ';' ;
7631 relational_expression ';' ;
7632 expression ')' statement
7633 | If '(' relational_expression ')' statement
7634 | While '(' relational_expression ')' statement
7635 | '{' statement_list '}' ;
7636 ;
7637 function : Define LETTER '(' opt_parameter_list ')' ;
7638 '{' NEWLINE opt_auto_define_list
7639 statement_list '}' ;
7640 ;
7641 opt_parameter_list : /* empty */
7642 | parameter_list
7643 ;
7644 parameter_list : LETTER
7645 | define_list ',' LETTER
7646 ;
7647 opt_auto_define_list : /* empty */
7648 | Auto define_list NEWLINE
7649 | Auto define_list ';' ;
7650 ;
7651 define_list : LETTER
7652 | LETTER '[' ']' ;
7653 | define_list ',' LETTER
7654 | define_list ',' LETTER '[' ']' ;
7655 ;
7656 opt_argument_list : /* empty */
7657 | argument_list
7658 ;
7659 argument_list : expression
7660 | LETTER '[' ']' ',' argument_list ;
7661 ;
```

```
7662 relational_expression : expression
7663 | expression REL_OP expression
7664 ;
7665 return_expression : /* empty */
7666 | expression
7667 ;
7668 expression : named_expression
7669 | NUMBER
7670 | '(' expression ')'
7671 | LETTER '(' opt_argument_list ')'
7672 | '-' expression
7673 | expression '+' expression
7674 | expression '-' expression
7675 | expression MUL_OP expression
7676 | expression '^' expression
7677 | INCR_DECR named_expression
7678 | named_expression INCR_DECR
7679 | named_expression ASSIGN_OP expression
7680 | Length '(' expression ')'
7681 | Sqrt '(' expression ')'
7682 | Scale '(' expression ')'
7683 ;
7684 named_expression : LETTER
7685 | LETTER '[' expression ']'
7686 | Scale
7687 | Ibase
7688 | Obase
7689 ;
```

## 7690 Lexical Conventions in bc

7691 The lexical conventions for *bc* programs, with respect to the preceding grammar, shall be as  
7692 follows:

- 7693 1. Except as noted, *bc* shall recognize the longest possible token or delimiter beginning at a  
7694 given point.
- 7695 2. A comment shall consist of any characters beginning with the two adjacent characters  
7696 "/\*" and terminated by the next occurrence of the two adjacent characters "\*/".  
7697 Comments shall have no effect except to delimit lexical tokens.
- 7698 3. The <newline> shall be recognized as the token NEWLINE.
- 7699 4. The token **STRING** shall represent a string constant; it shall consist of any characters  
7700 beginning with the double-quote character ('') and terminated by another occurrence of  
7701 the double-quote character. The value of the string is the sequence of all characters  
7702 between, but not including, the two double-quote characters. All characters shall be taken  
7703 literally from the input, and there is no way to specify a string containing a double-quote  
7704 character. The length of the value of each string shall be limited to {BC\_STRING\_MAX}  
7705 bytes.
- 7706 5. A <blank> shall have no effect except as an ordinary character if it appears within a  
7707 **STRING** token, or to delimit a lexical token other than **STRING**.

- 7708        6. The combination of a backslash character immediately followed by a <newline> shall have  
 7709        no effect other than to delimit lexical tokens with the following exceptions:

- 7710            • It shall be interpreted as the character sequence "\<newline>" in **STRING** tokens.  
 7711            • It shall be ignored as part of a multi-line **NUMBER** token.

- 7712        7. The token **NUMBER** shall represent a numeric constant. It shall be recognized by the  
 7713        following grammar:

```
7714 NUMBER : integer
 7715 | '.' integer
 7716 | integer '.'
 7717 | integer '.' integer
 7718 ;
 7719 integer : digit
 7720 | integer digit
 7721 ;
 7722 digit : 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7
 7723 | 8 | 9 | A | B | C | D | E | F
 7724 ;
```

- 7725        8. The value of a **NUMBER** token shall be interpreted as a numeral in the base specified by  
 7726        the value of the internal register **ibase** (described below). Each of the **digit** characters shall  
 7727        have the value from 0 to 15 in the order listed here, and the period character shall represent  
 7728        the radix point. The behavior is undefined if digits greater than or equal to the value of  
 7729        **ibase** appear in the token. However, note the exception for single-digit values being  
 7730        assigned to **ibase** and **obase** themselves, in **Operations in bc** (on page 198).

- 7731        9. The following keywords shall be recognized as tokens:

|               |              |               |               |              |
|---------------|--------------|---------------|---------------|--------------|
| <b>auto</b>   | <b>ibase</b> | <b>length</b> | <b>return</b> | <b>while</b> |
| <b>break</b>  | <b>if</b>    | <b>obase</b>  | <b>scale</b>  |              |
| <b>define</b> | <b>for</b>   | <b>quit</b>   | <b>sqrt</b>   |              |

- 7735        10. Any of the following characters occurring anywhere except within a keyword shall be  
 7736        recognized as the token **LETTER**:

7737        a b c d e f g h i j k l m n o p q r s t u v w x y z

- 7738        11. The following single-character and two-character sequences shall be recognized as the  
 7739        token **ASSIGN\_OP**:

7740        = += -= \*= /= %= ^=

- 7741        12. If an '=' character, as the beginning of a token, is followed by a '-' character with no  
 7742        intervening delimiter, the behavior is undefined.

- 7743        13. The following single-characters shall be recognized as the token **MUL\_OP**:

7744        \* / %

- 7745        14. The following single-character and two-character sequences shall be recognized as the  
 7746        token **REL\_OP**:

7747        == <= >= != < >

- 7748        15. The following two-character sequences shall be recognized as the token **INCR\_DECR**:

- 7749                $\text{++}$      $\text{--}$
- 7750     16. The following single characters shall be recognized as tokens whose names are the  
7751       character:
- 7752                $\langle\text{newline}\rangle$      $($      $)$      $,$      $+$      $-$      $;$      $[$      $]$      $^$      $\{$      $\}$
- 7753     17. The token **EOF** is returned when the end of input is reached.

7754     **Operations in bc**

7755     There are three kinds of identifiers: ordinary identifiers, array identifiers, and function  
7756       identifiers. All three types consist of single lowercase letters. Array identifiers shall be followed  
7757       by square brackets ("[ ]"). An array subscript is required except in an argument or auto list.  
7758       Arrays are singly dimensioned and can contain up to {BC\_DIM\_MAX} elements. Indexing shall  
7759       begin at zero so an array is indexed from 0 to {BC\_DIM\_MAX}-1. Subscripts shall be truncated  
7760       to integers. The application shall ensure that function identifiers are followed by parentheses,  
7761       possibly enclosing arguments. The three types of identifiers do not conflict.

7762     The following table summarizes the rules for precedence and associativity of all operators.  
7763     Operators on the same line shall have the same precedence; rows are in order of decreasing  
7764       precedence.

7765     **Table 4-3 Operators in bc**

| 7766 <b>Operator</b>                                | 7767 <b>Associativity</b> |
|-----------------------------------------------------|---------------------------|
| 7767 $\text{++}$ , $\text{--}$                      | N/A                       |
| 7768               unary $-$                        | N/A                       |
| 7769 $^$                                            | Right to left             |
| 7770 $*$ , $/$ , $\%$                               | Left to right             |
| 7771 $+$ , binary $-$                               | Left to right             |
| 7772 $=$ , $+=$ , $-=$ , $*=$ , $/=$ , $\%=$ , $^=$ | Right to left             |
| 7773 $==$ , $<=$ , $>=$ , $!=$ , $<$ , $>$          | None                      |

7774     Each expression or named expression has a *scale*, which is the number of decimal digits that  
7775       shall be maintained as the fractional portion of the expression.

7776     **Named expressions** are places where values are stored. Named expressions shall be valid on the  
7777       left side of an assignment. The value of a named expression shall be the value stored in the place  
7778       named. Simple identifiers and array elements are named expressions; they have an initial value  
7779       of zero and an initial scale of zero.

7780     The internal registers **scale**, **ibase**, and **obase** are all named expressions. The scale of an  
7781       expression consisting of the name of one of these registers shall be zero; values assigned to any  
7782       of these registers are truncated to integers. The **scale** register shall contain a global value used in  
7783       computing the scale of expressions (as described below). The value of the register **scale** is  
7784       limited to  $0 \leq \text{scale} \leq \{\text{BC\_SCALE\_MAX}\}$  and shall have a default value of zero. The **ibase** and  
7785       **obase** registers are the input and output number radix, respectively. The value of **ibase** shall be  
7786       limited to:

7787      $2 \leq \text{ibase} \leq 16$

7788     The value of **obase** shall be limited to:

7789      $2 \leq \text{obase} \leq \{\text{BC\_BASE\_MAX}\}$

7790     When either **ibase** or **obase** is assigned a single **digit** value from the list in **Lexical Conventions**  
7791       in **bc** (on page 196), the value shall be assumed in hexadecimal. (For example, **ibase=A** sets to

base ten, regardless of the current **ibase** value.) Otherwise, the behavior is undefined when digits greater than or equal to the value of **ibase** appear in the input. Both **ibase** and **obase** shall have initial values of 10.

Internal computations shall be conducted as if in decimal, regardless of the input and output bases, to the specified number of decimal digits. When an exact result is not achieved (for example, **scale**=0; 3.2/1), the result shall be truncated.

For all values of **obase** specified by this volume of IEEE Std 1003.1-2001, *bc* shall output numeric values by performing each of the following steps in order:

1. If the value is less than zero, a hyphen ('-') character shall be output.
2. One of the following is output, depending on the numerical value:
  - If the absolute value of the numerical value is greater than or equal to one, the integer portion of the value shall be output as a series of digits appropriate to **obase** (as described below), most significant digit first. The most significant non-zero digit shall be output next, followed by each successively less significant digit.
  - If the absolute value of the numerical value is less than one but greater than zero and the scale of the numerical value is greater than zero, it is unspecified whether the character 0 is output.
  - If the numerical value is zero, the character 0 shall be output.
3. If the scale of the value is greater than zero and the numeric value is not zero, a period character shall be output, followed by a series of digits appropriate to **obase** (as described below) representing the most significant portion of the fractional part of the value. If *s* represents the scale of the value being output, the number of digits output shall be *s* if **obase** is 10, less than or equal to *s* if **obase** is greater than 10, or greater than or equal to *s* if **obase** is less than 10. For **obase** values other than 10, this should be the number of digits needed to represent a precision of  $10^s$ .

For **obase** values from 2 to 16, valid digits are the first **obase** of the single characters:

0 1 2 3 4 5 6 7 8 9 A B C D E F

which represent the values zero to 15, inclusive, respectively.

For bases greater than 16, each digit shall be written as a separate multi-digit decimal number. Each digit except the most significant fractional digit shall be preceded by a single <space>. For bases from 17 to 100, *bc* shall write two-digit decimal numbers; for bases from 101 to 1000, three-digit decimal strings, and so on. For example, the decimal number 1 024 in base 25 would be written as:

Δ01Δ15Δ24

and in base 125, as:

Δ008Δ024

Very large numbers shall be split across lines with 70 characters per line in the POSIX locale; other locales may split at different character boundaries. Lines that are continued shall end with a backslash ('\').

A function call shall consist of a function name followed by parentheses containing a comma-separated list of expressions, which are the function arguments. A whole array passed as an argument shall be specified by the array name followed by empty square brackets. All function arguments shall be passed by value. As a result, changes made to the formal parameters shall have no effect on the actual arguments. If the function terminates by executing a **return**

7836 statement, the value of the function shall be the value of the expression in the parentheses of the  
7837 **return** statement or shall be zero if no expression is provided or if there is no **return** statement.

7838 The result of **sqrt(expression)** shall be the square root of the expression. The result shall be  
7839 truncated in the least significant decimal place. The scale of the result shall be the scale of the  
7840 expression or the value of **scale**, whichever is larger.

7841 The result of **length(expression)** shall be the total number of significant decimal digits in the  
7842 expression. The scale of the result shall be zero.

7843 The result of **scale(expression)** shall be the scale of the expression. The scale of the result shall be  
7844 zero.

7845 A numeric constant shall be an expression. The scale shall be the number of digits that follow the  
7846 radix point in the input representing the constant, or zero if no radix point appears.

7847 The sequence (*expression*) shall be an expression with the same value and scale as *expression*.  
7848 The parentheses can be used to alter the normal precedence.

7849 The semantics of the unary and binary operators are as follows:

7850 **-expression**

7851 The result shall be the negative of the *expression*. The scale of the result shall be the scale of  
7852 *expression*.

7853 The unary increment and decrement operators shall not modify the scale of the named  
7854 expression upon which they operate. The scale of the result shall be the scale of that named  
7855 expression.

7856 **++named-expression**

7857 The named expression shall be incremented by one. The result shall be the value of the  
7858 named expression after incrementing.

7859 **--named-expression**

7860 The named expression shall be decremented by one. The result shall be the value of the  
7861 named expression after decrementing.

7862 **named-expression++**

7863 The named expression shall be incremented by one. The result shall be the value of the  
7864 named expression before incrementing.

7865 **named-expression--**

7866 The named expression shall be decremented by one. The result shall be the value of the  
7867 named expression before decrementing.

7868 The exponentiation operator, circumflex ('^'), shall bind right to left.

7869 **expression^expression**

7870 The result shall be the first *expression* raised to the power of the second *expression*. If the  
7871 second expression is not an integer, the behavior is undefined. If *a* is the scale of the left  
7872 expression and *b* is the absolute value of the right expression, the scale of the result shall be:  
7873     if *b* >= 0 min(*a* \* *b*, max(scale, *a*)) if *b* < 0 scale

7874 The multiplicative operators ('\*', '/', '%') shall bind left to right.

7875 **expression\*expression**

7876 The result shall be the product of the two expressions. If *a* and *b* are the scales of the two  
7877 expressions, then the scale of the result shall be:

```

7878 min(a+b,max(scale,a,b))
7879 expression/expression
7880 The result shall be the quotient of the two expressions. The scale of the result shall be the
7881 value of scale.
7882 expression%expression
7883 For expressions a and b, a%b shall be evaluated equivalent to the steps:
7884 1. Compute a/b to current scale.
7885 2. Use the result to compute:
7886 a - (a / b) * b
7887 to scale:
7888 max(scale + scale(b), scale(a))
7889 The scale of the result shall be:
7890 max(scale + scale(b), scale(a))
7891 When scale is zero, the '%' operator is the mathematical remainder operator.
7892 The additive operators ('+', '-') shall bind left to right.
7893 expression+expression
7894 The result shall be the sum of the two expressions. The scale of the result shall be the
7895 maximum of the scales of the expressions.
7896 expression-expression
7897 The result shall be the difference of the two expressions. The scale of the result shall be the
7898 maximum of the scales of the expressions.
7899 The assignment operators ('=' , "+=" , "-=" , "*=" , "/=" , "%=" , "^=") shall bind right to left.
7900 named-expression=expression
7901 This expression shall result in assigning the value of the expression on the right to the
7902 named expression on the left. The scale of both the named expression and the result shall be
7903 the scale of expression.
7904 The compound assignment forms:
7905 named-expression <operator>= expression
7906 shall be equivalent to:
7907 named-expression=named-expression <operator> expression
7908 except that the named-expression shall be evaluated only once.
7909 Unlike all other operators, the relational operators ('<', '>', "<=", ">=", "==" , "!=")
7910 shall be only valid as the object of an if, while, or inside a for statement.
7911 expression1<expression2
7912 The relation shall be true if the value of expression1 is strictly less than the value of
7913 expression2.
7914 expression1>expression2
7915 The relation shall be true if the value of expression1 is strictly greater than the value of
7916 expression2.

```

7917 *expression1<=expression2*

The relation shall be true if the value of *expression1* is less than or equal to the value of *expression2*.

7920 *expression1>=expression2*

The relation shall be true if the value of *expression1* is greater than or equal to the value of *expression2*.

7923 *expression1==expression2*

The relation shall be true if the values of *expression1* and *expression2* are equal.

7925 *expression1!=expression2*

The relation shall be true if the values of *expression1* and *expression2* are unequal.

7927 There are only two storage classes in *bc*: global and automatic (local). Only identifiers that are local to a function need be declared with the **auto** command. The arguments to a function shall be local to the function. All other identifiers are assumed to be global and available to all functions. All identifiers, global and local, have initial values of zero. Identifiers declared as auto shall be allocated on entry to the function and released on returning from the function. They therefore do not retain values between function calls. Auto arrays shall be specified by the array name followed by empty square brackets. On entry to a function, the old values of the names that appear as parameters and as automatic variables shall be pushed onto a stack. Until the function returns, reference to these names shall refer only to the new values.

7936 References to any of these names from other functions that are called from this function also refer to the new value until one of those functions uses the same name for a local variable.

7938 When a statement is an expression, unless the main operator is an assignment, execution of the statement shall write the value of the expression followed by a <newline>.

7940 When a statement is a string, execution of the statement shall write the value of the string.

7941 Statements separated by semicolons or <newline>s shall be executed sequentially. In an interactive invocation of *bc*, each time a <newline> is read that satisfies the grammatical production:

7944 input\_item : semicolon\_list NEWLINE

7945 the sequential list of statements making up the **semicolon\_list** shall be executed immediately  
7946 and any output produced by that execution shall be written without any delay due to buffering.

7947 In an **if** statement (**if**(*relation*) *statement*), the *statement* shall be executed if the *relation* is true.

7948 The **while** statement (**while**(*relation*) *statement*) implements a loop in which the *relation* is tested;  
7949 each time the *relation* is true, the *statement* shall be executed and the *relation* retested. When the  
7950 *relation* is false, execution shall resume after *statement*.

7951 A **for** statement(**for**(*expression*; *relation*; *expression*) *statement*) shall be the same as:

7952 *first-expression*  
7953 while (*relation*) {  
7954     *statement*  
7955     *last-expression*  
7956 }

7957 The application shall ensure that all three expressions are present.

7958 The **break** statement shall cause termination of a **for** or **while** statement.

7959 The **auto** statement (**auto** *identifier* [,*identifier*] ...) shall cause the values of the *identifiers* to be  
7960 pushed down. The *identifiers* can be ordinary *identifiers* or *array identifiers*. *Array identifiers*

7961 shall be specified by following the array name by empty square brackets. The application shall  
 7962 ensure that the **auto** statement is the first statement in a function definition.

7963 A **define** statement:

```
7964 define LETTER (opt_parameter_list) {
7965 opt_auto_define_list
7966 statement_list
7967 }
```

7968 defines a function named **LETTER**. If a function named **LETTER** was previously defined, the  
 7969 **define** statement shall replace the previous definition. The expression:

```
7970 LETTER (opt_argument_list)
```

7971 shall invoke the function named **LETTER**. The behavior is undefined if the number of  
 7972 arguments in the invocation does not match the number of parameters in the definition.  
 7973 Functions shall be defined before they are invoked. A function shall be considered to be defined  
 7974 within its own body, so recursive calls are valid. The values of numeric constants within a  
 7975 function shall be interpreted in the base specified by the value of the **ibase** register when the  
 7976 function is invoked.

7977 The **return** statements (**return** and **return(expression)**) shall cause termination of a function,  
 7978 popping of its auto variables, and specification of the result of the function. The first form shall  
 7979 be equivalent to **return(0)**. The value and scale of the result returned by the function shall be the  
 7980 value and scale of the expression returned.

7981 The **quit** statement (**quit**) shall stop execution of a *bc* program at the point where the statement  
 7982 occurs in the input, even if it occurs in a function definition, or in an **if**, **for**, or **while** statement.

7983 The following functions shall be defined when the **-l** option is specified:

7984 **s(expression)**  
 7985     Sine of argument in radians.

7986 **c(expression)**  
 7987     Cosine of argument in radians.

7988 **a(expression)**  
 7989     Arctangent of argument.

7990 **l(expression)**  
 7991     Natural logarithm of argument.

7992 **e(expression)**  
 7993     Exponential function of argument.

7994 **j(expression, expression)**  
 7995     Bessel function of integer order.

7996 The scale of the result returned by these functions shall be the value of the **scale** register at the  
 7997 time the function is invoked. The value of the **scale** register after these functions have completed  
 7998 their execution shall be the same value it had upon invocation. The behavior is undefined if any  
 7999 of these functions is invoked with an argument outside the domain of the mathematical  
 8000 function.

## 8001 EXIT STATUS

8002 The following exit values shall be returned:

8003 0           All input files were processed successfully.

8004       *unspecified*   An error occurred.

## 8005 CONSEQUENCES OF ERRORS

8006       If any *file* operand is specified and the named file cannot be accessed, *bc* shall write a diagnostic  
8007       message to standard error and terminate without any further action.

8008       In an interactive invocation of *bc*, the utility should print an error message and recover following  
8009       any error in the input. In a non-interactive invocation of *bc*, invalid input causes undefined  
8010       behavior.

## 8011 APPLICATION USAGE

8012       Automatic variables in *bc* do not work in exactly the same way as in either C or PL/1.

8013       For historical reasons, the exit status from *bc* cannot be relied upon to indicate that an error has  
8014       occurred. Returning zero after an error is possible. Therefore, *bc* should be used primarily by  
8015       interactive users (who can react to error messages) or by application programs that can  
8016       somehow validate the answers returned as not including error messages.

8017       The *bc* utility always uses the period ('.') character to represent a radix point, regardless of any  
8018       decimal-point character specified as part of the current locale. In languages like C or awk, the  
8019       period character is used in program source, so it can be portable and unambiguous, while the  
8020       locale-specific character is used in input and output. Because there is no distinction between  
8021       source and input in *bc*, this arrangement would not be possible. Using the locale-specific  
8022       character in *bc*'s input would introduce ambiguities into the language; consider the following  
8023       example in a locale with a comma as the decimal-point character:

```
8024 define f(a,b) {
8025 ...
8026 }
8027 ...
8028 f(1,2,3)
```

8029       Because of such ambiguities, the period character is used in input. Having input follow different  
8030       conventions from output would be confusing in either pipeline usage or interactive usage, so the  
8031       period is also used in output.

## 8032 EXAMPLES

8033       In the shell, the following assigns an approximation of the first ten digits of ' $\pi$ ' to the variable  
8034       *x*:

```
8035 x=$(printf "%s\n" 'scale = 10; 104348/33215' | bc)
```

8036       The following *bc* program prints the same approximation of ' $\pi$ ', with a label, to standard  
8037       output:

```
8038 scale = 10
8039 "pi equals "
8040 104348 / 33215
```

8041       The following defines a function to compute an approximate value of the exponential function  
8042       (note that such a function is predefined if the **-l** option is specified):

```
8043 scale = 20
8044 define e(x){
8045 auto a, b, c, i, s
8046 a = 1
8047 b = 1
8048 s = 1
```

```

8049 for (i = 1; i == 1; i++) {
8050 a = a*x
8051 b = b*i
8052 c = a/b
8053 if (c == 0) {
8054 return(s)
8055 }
8056 s = s+c
8057 }
8058 }
```

8059 The following prints approximate values of the exponential function of the first ten integers:

```

8060 for (i = 1; i <= 10; ++i) {
8061 e(i)
8062 }
```

### 8063 RATIONALE

8064 The *bc* utility is implemented historically as a front-end processor for *dc*; *dc* was not selected to  
 8065 be part of this volume of IEEE Std 1003.1-2001 because *bc* was thought to have a more intuitive  
 8066 programmatic interface. Current implementations that implement *bc* using *dc* are expected to be  
 8067 compliant.

8068 The exit status for error conditions has been left unspecified for several reasons:

- 8069 • The *bc* utility is used in both interactive and non-interactive situations. Different exit codes  
 8070 may be appropriate for the two uses.
- 8071 • It is unclear when a non-zero exit should be given; divide-by-zero, undefined functions, and  
 8072 syntax errors are all possibilities.
- 8073 • It is not clear what utility the exit status has.
- 8074 • In the 4.3 BSD, System V, and Ninth Edition implementations, *bc* works in conjunction with  
 8075 *dc*. The *dc* utility is the parent, *bc* is the child. This was done to cleanly terminate *bc* if *dc*  
 8076 aborted.

8077 The decision to have *bc* exit upon encountering an inaccessible input file is based on the belief  
 8078 that *bc file1 file2* is used most often when at least *file1* contains data/function  
 8079 declarations/initializations. Having *bc* continue with prerequisite files missing is probably not  
 8080 useful. There is no implication in the CONSEQUENCES OF ERRORS section that *bc* must check  
 8081 all its files for accessibility before opening any of them.

8082 There was considerable debate on the appropriateness of the language accepted by *bc*. Several  
 8083 reviewers preferred to see either a pure subset of the C language or some changes to make the  
 8084 language more compatible with C. While the *bc* language has some obvious similarities to C, it  
 8085 has never claimed to be compatible with any version of C. An interpreter for a subset of C might  
 8086 be a very worthwhile utility, and it could potentially make *bc* obsolete. However, no such utility  
 8087 is known in historical practice, and it was not within the scope of this volume of  
 8088 IEEE Std 1003.1-2001 to define such a language and utility. If and when they are defined, it may  
 8089 be appropriate to include them in a future version of IEEE Std 1003.1. This left the following  
 8090 alternatives:

- 8091 1. Exclude any calculator language from this volume of IEEE Std 1003.1-2001.

8092 The consensus of the standard developers was that a simple programmatic calculator  
 8093 language is very useful for both applications and interactive users. The only arguments for  
 8094 excluding any calculator were that it would become obsolete if and when a C-compatible

8095 one emerged, or that the absence would encourage the development of such a C-  
8096 compatible one. These arguments did not sufficiently address the needs of current  
8097 application writers.

8098 2. Standardize the historical *dc*, possibly with minor modifications.

8099 The consensus of the standard developers was that *dc* is a fundamentally less usable  
8100 language and that that would be far too severe a penalty for avoiding the issue of being  
8101 similar to but incompatible with C.

8102 3. Standardize the historical *bc*, possibly with minor modifications.

8103 This was the approach taken. Most of the proponents of changing the language would not  
8104 have been satisfied until most or all of the incompatibilities with C were resolved. Since  
8105 most of the changes considered most desirable would break historical applications and  
8106 require significant modification to historical implementations, almost no modifications  
8107 were made. The one significant modification that was made was the replacement of the  
8108 historical *bc* assignment operators " $=+$ ", and so on, with the more modern " $+=$ ", and so  
8109 on. The older versions are considered to be fundamentally flawed because of the lexical  
8110 ambiguity in uses like  $a=-1$ .

8111 In order to permit implementations to deal with backwards-compatibility as they see fit,  
8112 the behavior of this one ambiguous construct was made undefined. (At least three  
8113 implementations have been known to support this change already, so the degree of change  
8114 involved should not be great.)

8115 The '%' operator is the mathematical remainder operator when **scale** is zero. The behavior of  
8116 this operator for other values of **scale** is from historical implementations of *bc*, and has been  
8117 maintained for the sake of historical applications despite its non-intuitive nature.

8118 Historical implementations permit setting **ibase** and **obase** to a broader range of values. This  
8119 includes values less than 2, which were not seen as sufficiently useful to standardize. These  
8120 implementations do not interpret input properly for values of **ibase** that are greater than 16. This  
8121 is because numeric constants are recognized syntactically, rather than lexically, as described in  
8122 this volume of IEEE Std 1003.1-2001. They are built from lexical tokens of single hexadecimal  
8123 digits and periods. Since <blank>s between tokens are not visible at the syntactic level, it is not  
8124 possible to recognize the multi-digit "digits" used in the higher bases properly. The ability to  
8125 recognize input in these bases was not considered useful enough to require modifying these  
8126 implementations. Note that the recognition of numeric constants at the syntactic level is not a  
8127 problem with conformance to this volume of IEEE Std 1003.1-2001, as it does not impact the  
8128 behavior of conforming applications (and correct *bc* programs). Historical implementations also  
8129 accept input with all of the digits '0'-'9' and 'A'-'F' regardless of the value of **ibase**; since  
8130 digits with value greater than or equal to **ibase** are not really appropriate, the behavior when  
8131 they appear is undefined, except for the common case of:

```
8132 ibase=8;
8133 /* Process in octal base. */
8134 ...
8135 ibase=A
8136 /* Restore decimal base. */
```

8137 In some historical implementations, if the expression to be written is an uninitialized array  
8138 element, a leading <space> and/or up to four leading 0 characters may be output before the  
8139 character zero. This behavior is considered a bug; it is unlikely that any currently conforming  
8140 application relies on:

8141        echo 'b[ 3 ]' | bc

8142        returning 00000 rather than 0.

8143        Exact calculation of the number of fractional digits to output for a given value in a base other  
8144        than 10 can be computationally expensive. Historical implementations use a faster  
8145        approximation, and this is permitted. Note that the requirements apply only to values of **obase**  
8146        that this volume of IEEE Std 1003.1-2001 requires implementations to support (in particular, not  
8147        to 1, 0, or negative bases, if an implementation supports them as an extension).

8148        Historical implementations of *bc* did not allow array parameters to be passed as the last  
8149        parameter to a function. New implementations are encouraged to remove this restriction even  
8150        though it is not required by the grammar.

8151 **FUTURE DIRECTIONS**

8152        None.

8153 **SEE ALSO**

8154        Section 1.10 (on page 19), *awk*

8155 **CHANGE HISTORY**

8156        First released in Issue 4.

8157 **Issue 5**

8158        The FUTURE DIRECTIONS section is added.

8159 **Issue 6**

8160        Updated to align with the IEEE P1003.2b draft standard, which included resolution of several  
8161        interpretations of the ISO POSIX-2:1993 standard.

8162        The normative text is reworded to avoid use of the term “must” for application requirements.

8163 **NAME**

8164 bg — run jobs in the background

8165 **SYNOPSIS**

8166 UP bg [ *job\_id* . . . ]

8167

8168 **DESCRIPTION**

8169 If job control is enabled (see the description of *set -m*), the *bg* utility shall resume suspended jobs  
8170 from the current environment (see Section 2.12 (on page 61)) by running them as background  
8171 jobs. If the job specified by *job\_id* is already a running background job, the *bg* utility shall have no  
8172 effect and shall exit successfully.

8173 Using *bg* to place a job into the background shall cause its process ID to become “known in the  
8174 current shell execution environment”, as if it had been started as an asynchronous list; see  
8175 Section 2.9.3.1 (on page 50).

8176 **OPTIONS**

8177 None.

8178 **OPERANDS**

8179 The following operand shall be supported:

8180 *job\_id* Specify the job to be resumed as a background job. If no *job\_id* operand is given,  
8181 the most recently suspended job shall be used. The format of *job\_id* is described in  
8182 the Base Definitions volume of IEEE Std 1003.1-2001, Section 3.203, Job Control Job  
8183 ID.

8184 **STDIN**

8185 Not used.

8186 **INPUT FILES**

8187 None.

8188 **ENVIRONMENT VARIABLES**

8189 The following environment variables shall affect the execution of *bg*:

8190 *LANG* Provide a default value for the internationalization variables that are unset or null.  
8191 (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2,  
8192 Internationalization Variables for the precedence of internationalization variables  
8193 used to determine the values of locale categories.)

8194 *LC\_ALL* If set to a non-empty string value, override the values of all the other  
8195 internationalization variables.

8196 *LC\_CTYPE* Determine the locale for the interpretation of sequences of bytes of text data as  
8197 characters (for example, single-byte as opposed to multi-byte characters in  
8198 arguments).

8199 *LC\_MESSAGES*

8200 Determine the locale that should be used to affect the format and contents of  
8201 diagnostic messages written to standard error.

8202 XSI *NLSPATH* Determine the location of message catalogs for the processing of *LC\_MESSAGES*.

8203 **ASYNCHRONOUS EVENTS**

8204 Default.

**8205 STDOUT**

8206 The output of *bg* shall consist of a line in the format:

8207 " [ %d ] %s\n", <job-number>, <command>

8208 where the fields are as follows:

8209 <job-number> A number that can be used to identify the job to the *wait*, *fg*, and *kill* utilities. Using  
8210 these utilities, the job can be identified by prefixing the job number with '%'.

8211 <command> The associated command that was given to the shell.

**8212 STDERR**

8213 The standard error shall be used only for diagnostic messages.

**8214 OUTPUT FILES**

8215 None.

**8216 EXTENDED DESCRIPTION**

8217 None.

**8218 EXIT STATUS**

8219 The following exit values shall be returned:

8220 0 Successful completion.

8221 >0 An error occurred.

**8222 CONSEQUENCES OF ERRORS**

8223 If job control is disabled, the *bg* utility shall exit with an error and no job shall be placed in the  
8224 background.

**8225 APPLICATION USAGE**

8226 A job is generally suspended by typing the SUSP character (<control>-Z on most systems); see  
8227 the Base Definitions volume of IEEE Std 1003.1-2001, Chapter 11, General Terminal Interface. At  
8228 that point, *bg* can put the job into the background. This is most effective when the job is  
8229 expecting no terminal input and its output has been redirected to non-terminal files. A  
8230 background job can be forced to stop when it has terminal output by issuing the command:

8231 stty tostop

8232 A background job can be stopped with the command:

8233 kill -s stop job ID

8234 The *bg* utility does not work as expected when it is operating in its own utility execution  
8235 environment because that environment has no suspended jobs. In the following examples:

8236 . . . | xargs bg  
8237 (bg)

8238 each *bg* operates in a different environment and does not share its parent shell's understanding  
8239 of jobs. For this reason, *bg* is generally implemented as a shell regular built-in.

**8240 EXAMPLES**

8241 None.

**8242 RATIONALE**

8243 The extensions to the shell specified in this volume of IEEE Std 1003.1-2001 have mostly been  
8244 based on features provided by the KornShell. The job control features provided by *bg*, *fg*, and *jobs*  
8245 are also based on the KornShell. The standard developers examined the characteristics of the C  
8246 shell versions of these utilities and found that differences exist. Despite widespread use of the C

8247 shell, the KornShell versions were selected for this volume of IEEE Std 1003.1-2001 to maintain a  
8248 degree of uniformity with the rest of the KornShell features selected (such as the very popular  
8249 command line editing features).

8250 The *bg* utility is expected to wrap its output if the output exceeds the number of display  
8251 columns.

8252 **FUTURE DIRECTIONS**

8253 None.

8254 **SEE ALSO**

8255 Section 2.9.3.1 (on page 50), *fg*, *kill*, *jobs*, *wait*

8256 **CHANGE HISTORY**

8257 First released in Issue 4.

8258 **Issue 6**

8259 This utility is marked as part of the User Portability Utilities option.

8260 The JC margin marker on the SYNOPSIS is removed since support for Job Control is mandatory  
8261 in this issue. This is a FIPS requirement.

8262 **NAME**

8263      c99 — compile standard C programs

8264 **SYNOPSIS**

```
8265 CD c99 [-c][-D name[=value]]...[-E][-g][-I directory] ... [-L directory]
8266 ... [-o outfile][-Oopt level][-s][-U name]... operand ...
```

8267

8268 **DESCRIPTION**

8269      The *c99* utility is an interface to the standard C compilation system; it shall accept source code  
 8270      conforming to the ISO C standard. The system conceptually consists of a compiler and link  
 8271      editor. The files referenced by *operands* shall be compiled and linked to produce an executable  
 8272      file. (It is unspecified whether the linking occurs entirely within the operation of *c99*; some  
 8273      implementations may produce objects that are not fully resolved until the file is executed.)

8274      If the **-c** option is specified, for all pathname operands of the form *file.c*, the files:

8275      \$(basename *pathname* .c).o

8276      shall be created as the result of successful compilation. If the **-c** option is not specified, it is  
 8277      unspecified whether such **.o** files are created or deleted for the *file.c* operands.

8278      If there are no options that prevent link editing (such as **-c** or **-E**), and all operands compile and  
 8279      link without error, the resulting executable file shall be written according to the **-o** *outfile* option  
 8280      (if present) or to the file **a.out**.

8281      The executable file shall be created as specified in Section 1.7.1.4 (on page 4), except that the file  
 8282      permission bits shall be set to:

8283      S\_IRWXO | S\_IRWXG | S\_IROWXU

8284      and the bits specified by the *umask* of the process shall be cleared.

8285 **OPTIONS**

8286      The *c99* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section  
 8287      12.2, Utility Syntax Guidelines, except that:

- 8288      • The **-l** *library* operands have the format of options, but their position within a list of  
 8289      operands affects the order in which libraries are searched.
- 8290      • The order of specifying the **-I** and **-L** options is significant.
- 8291      • Conforming applications shall specify each option separately; that is, grouping option letters  
 8292      (for example, **-cO**) need not be recognized by all implementations.

8293      The following options shall be supported:

- |                               |                                                                                                                                                                                                                                                                                                                                                                                 |
|-------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 8294 <b>-c</b>                | Suppress the link-edit phase of the compilation, and do not remove any object files<br>8295      that are produced.                                                                                                                                                                                                                                                             |
| 8296 <b>-g</b>                | Produce symbolic information in the object or executable files; the nature of this<br>8297      information is unspecified, and may be modified by implementation-defined<br>8298      interactions with other options.                                                                                                                                                         |
| 8299 <b>-s</b>                | Produce object or executable files, or both, from which symbolic and other<br>8300      information not required for proper execution using the <i>exec</i> family defined in the<br>8301      System Interfaces volume of IEEE Std 1003.1-2001 has been removed (stripped). If<br>8302      both <b>-g</b> and <b>-s</b> options are present, the action taken is unspecified. |
| 8303 <b>-o</b> <i>outfile</i> | Use the pathname <i>outfile</i> , instead of the default <b>a.out</b> , for the executable file<br>8304      produced. If the <b>-o</b> option is present with <b>-c</b> or <b>-E</b> , the result is unspecified.                                                                                                                                                              |

|      |                                                                                                       |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
|------|-------------------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 8305 | <b>-D</b> <i>name[=value]</i>                                                                         | Define <i>name</i> as if by a C-language <b>#define</b> directive. If no <i>=value</i> is given, a value of 1 shall be used. The <b>-D</b> option has lower precedence than the <b>-U</b> option. That is, if <i>name</i> is used in both a <b>-U</b> and a <b>-D</b> option, <i>name</i> shall be undefined regardless of the order of the options. Additional implementation-defined <i>names</i> may be provided by the compiler. Implementations shall support at least 2 048 bytes of <b>-D</b> definitions and 256 <i>names</i> .                                                                                                                                                                                                                                                                    |
| 8312 | <b>-E</b>                                                                                             | Copy C-language source files to standard output, expanding all preprocessor directives; no compilation shall be performed. If any operand is not a text file, the effects are unspecified.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
| 8315 | <b>-I</b> <i>directory</i>                                                                            | Change the algorithm for searching for headers whose names are not absolute pathnames to look in the directory named by the <i>directory</i> pathname before looking in the usual places. Thus, headers whose names are enclosed in double-quotes (" ") shall be searched for first in the directory of the file with the <b>#include</b> line, then in directories named in <b>-I</b> options, and last in the usual places. For headers whose names are enclosed in angle brackets ("<>"), the header shall be searched for only in directories named in <b>-I</b> options and then in the usual places. Directories named in <b>-I</b> options shall be searched in the order specified. Implementations shall support at least ten instances of this option in a single <i>c99</i> command invocation. |
| 8325 | <b>-L</b> <i>directory</i>                                                                            | Change the algorithm of searching for the libraries named in the <b>-l</b> objects to look in the directory named by the <i>directory</i> pathname before looking in the usual places. Directories named in <b>-L</b> options shall be searched in the order specified. Implementations shall support at least ten instances of this option in a single <i>c99</i> command invocation. If a directory specified by a <b>-L</b> option contains files with names starting with any of the strings "libc.", "libl.", "libpthread.", "libm.", "librt.", "libtrace.", "libxnet.", or "liby.", the results are unspecified.                                                                                                                                                                                     |
| 8332 | <b>-O</b> <i>optlevel</i>                                                                             | Specify the level of code optimization. If the <i>optlevel</i> option-argument is the digit '0', all special code optimizations shall be disabled. If it is the digit '1', the nature of the optimization is unspecified. If the <b>-O</b> option is omitted, the nature of the system's default optimization is unspecified. It is unspecified whether code generated in the presence of the <b>-O 0</b> option is the same as that generated when <b>-O</b> is omitted. Other <i>optlevel</i> values may be supported.                                                                                                                                                                                                                                                                                   |
| 8339 | <b>-U</b> <i>name</i>                                                                                 | Remove any initial definition of <i>name</i> .                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| 8340 | Multiple instances of the <b>-D</b> , <b>-I</b> , <b>-U</b> , and <b>-L</b> options can be specified. |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |

## 8341 OPERANDS

An *operand* is either in the form of a pathname or the form **-l** *library*. The application shall ensure that at least one operand of the pathname form is specified. The following operands shall be supported:

|      |               |                                                                                                                                                                                                                                      |
|------|---------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 8345 | <i>file.c</i> | A C-language source file to be compiled and optionally linked. The application shall ensure that the operand is of this form if the <b>-c</b> option is used.                                                                        |
| 8347 | <i>file.a</i> | A library of object files typically produced by the <i>ar</i> utility, and passed directly to the link editor. Implementations may recognize implementation-defined suffixes other than <b>.a</b> as denoting object file libraries. |
| 8350 | <i>file.o</i> | An object file produced by <i>c99 -c</i> and passed directly to the link editor. Implementations may recognize implementation-defined suffixes other than <b>.o</b> as                                                               |

8360 STDIN

8361 Not used.

8362 INPUT FILES

8363 The input file shall be one of the following: a text file containing a C-language source program,  
8364 an object file in the format produced by *c99 -c*, or a library of object files, in the format produced  
8365 by archiving zero or more object files, using *ar*. Implementations may supply additional utilities  
8366 that produce files in these formats. Additional input file formats are implementation-defined.

## 8367 ENVIRONMENT VARIABLES

8368 The following environment variables shall affect the execution of *c99*:

8369        ***LANG***      Provide a default value for the internationalization variables that are unset or null.  
8370                          (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2,  
8371                          Internationalization Variables for the precedence of internationalization variables  
8372                          used to determine the values of locale categories.)

8373        ***LC\_ALL***     If set to a non-empty string value, override the values of all the other  
8374              internationalization variables.

8378 *LC\_MESSAGES*

8379 Determine the locale that should be used to affect the format and contents of  
8380 diagnostic messages written to standard error.

**8381 XSI NLSPATH** Determine the location of message catalogs for the processing of *LC\_MESSAGES*.

8382           **TMPDIR**   Provide a pathname that should override the default directory for temporary files,  
8383   XSI           if any. On XSI-conforming systems, provide a pathname that shall override the  
8384                 default directory for temporary files, if any.

8385 ASYNCHRONOUS EVENTS

8386 Default.

8387 STDOUT

If more than one *file* operand ending in .c (or possibly other unspecified suffixes) is given, for each such file:

8390 " %s :\n" , <file>

8391 may be written. These messages, if written, shall precede the processing of each input file; they  
8392 shall not be written to the standard output if they are written to the standard error, as described  
8393 in the STDERR section.

8394 If the **-E** option is specified, the standard output shall be a text file that represents the results of  
 8395 the preprocessing stage of the language; it may contain extra information appropriate for  
 8396 subsequent compilation passes.

## 8397 **STDERR**

8398 The standard error shall be used only for diagnostic messages. If more than one *file* operand  
 8399 ending in **.c** (or possibly other unspecified suffixes) is given, for each such file:

8400 " %s :\n" , <*file*>

8401 may be written to allow identification of the diagnostic and warning messages with the  
 8402 appropriate input file. These messages, if written, shall precede the processing of each input file;  
 8403 they shall not be written to the standard error if they are written to the standard output, as  
 8404 described in the STDOUT section.

8405 This utility may produce warning messages about certain conditions that do not warrant  
 8406 returning an error (non-zero) exit value.

## 8407 **OUTPUT FILES**

8408 Object files or executable files or both are produced in unspecified formats.

## 8409 **EXTENDED DESCRIPTION**

### 8410 **Standard Libraries**

8411 The *c99* utility shall recognize the following **-l** operands for standard libraries:

- 8412   **-l c**       This operand shall make visible all functions referenced in the System Interfaces  
                  volume of IEEE Std 1003.1-2001, with the possible exception of those functions  
                  listed as residing in **<aio.h>**, **<arpa/inet.h>**, **<complex.h>**, **<fenv.h>**, **<math.h>**,  
                  **<mqueue.h>**,   **<netdb.h>**,   **<netinet/in.h>**,   **<pthread.h>**,   **<sched.h>**,  
                  **<semaphore.h>**, **<spawn.h>**, **<sys/socket.h>**, **pthread\_kill()**, and **pthread\_sigmask()**  
                  in **<signal.h>**, **<trace.h>**, functions marked as extensions other than as part of the  
                  MF or MPR extensions in **<sys/mman.h>**, functions marked as ADV in **<fcntl.h>**,  
                  and functions marked as CS, CPT, and TMR in **<time.h>**. This operand shall not  
                  be required to be present to cause a search of this library.
- 8421   **-l l**        This operand shall make visible all functions required by the C-language output of  
                  *lex* that are not made available through the **-l c** operand.
- 8423   **-l pthread**   This operand shall make visible all functions referenced in **<pthread.h>** and  
                  **pthread\_kill()** and **pthread\_sigmask()** referenced in **<signal.h>**. An implementation  
                  may search this library in the absence of this operand.
- 8426   **-l m**       This operand shall make visible all functions referenced in **<math.h>**,  
                  **<complex.h>**, and **<fenv.h>**. An implementation may search this library in the  
                  absence of this operand.
- 8429   **-l rt**       This operand shall make visible all functions referenced in **<aio.h>**, **<mqueue.h>**,  
                  **<sched.h>**, **<semaphore.h>**, and **<spawn.h>**, functions marked as extensions other  
                  than as part of the MF or MPR extensions in **<sys/mman.h>**, functions marked as  
                  ADV in **<fcntl.h>**, and functions marked as CS, CPT, and TMR in **<time.h>**. An  
                  implementation may search this library in the absence of this operand.
- 8434   **-l trace**     This operand shall make visible all functions referenced in **<trace.h>**. An  
                  implementation may search this library in the absence of this operand.
- 8436   **-l xnet**      This operand makes visible all functions referenced in **<arpa/inet.h>**, **<netdb.h>**,  
                  **<netinet/in.h>**, and **<sys/socket.h>**. An implementation may search this library in

the absence of this operand.

**-ly** This operand shall make visible all functions required by the C-language output of yacc that are not made available through the **-l c** operand.

8441 In the absence of options that inhibit invocation of the link editor, such as **-c** or **-E**, the *c99* utility  
8442 shall cause the equivalent of a **-l c** operand to be passed to the link editor as the last **-l** operand,  
8443 causing it to be searched after all other object files and libraries are loaded.

8444 It is unspecified whether the libraries **libc.a**, **libm.a**, **librt.a**, **libpthread.a**, **libl.a**, **liby.a**, or 1  
8445 **libxnet.a** exist as regular files. The implementation may accept as **-l** operands names of objects 1  
8446 that do not exist as regular files.

## 8447 External Symbols

8448 The C compiler and link editor shall support the significance of external symbols up to a length  
8449 of at least 31 bytes; the action taken upon encountering symbols exceeding the implementation-  
8450 defined maximum symbol length is unspecified.

8451 The compiler and link editor shall support a minimum of 511 external symbols per source or  
8452 object file, and a minimum of 4 095 external symbols in total. A diagnostic message shall be  
8453 written to the standard output if the implementation-defined limit is exceeded; other actions are  
8454 unspecified.

8455 Programming Environments

All implementations shall support one of the following programming environments as a default. Implementations may support more than one of the following programming environments. Applications can use `sysconf()` or `getconf` to determine which programming environments are supported.

**Table 4-4** Programming Environments: Type Sizes

| Programming Environment<br><i>getconf Name</i> | Bits in<br>int | Bits in<br>long | Bits in<br>pointer | Bits in<br>off_t |
|------------------------------------------------|----------------|-----------------|--------------------|------------------|
| _POSIX_V6_ILP32_OFF32                          | 32             | 32              | 32                 | 32               |
| _POSIX_V6_ILP32_OFFBIG                         | 32             | 32              | 32                 | ≥64              |
| _POSIX_V6_LP64_OFF64                           | 32             | 64              | 64                 | 64               |
| _POSIX_V6_LPBIG_OFFBIG                         | ≥32            | ≥64             | ≥64                | ≥64              |

8467 All implementations shall support one or more environments where the widths of the following  
8468 types are no greater than the width of type **long**:

8469       **blksize\_t, cc\_t, mode\_t, nfds\_t, pid\_t, ptrdiff\_t, size\_t, speed\_t, ssize\_t, suseconds\_t,**  
8470        **tcflag\_t, useconds\_t, wchar\_t, wint\_t**

The executable files created when these environments are selected shall be in a proper format for execution by the *exec* family of functions. Each environment may be one of the ones in Table 4-4, or it may be another environment. The names for the environments that meet this requirement shall be output by a *getconf* command using the *POSIX\_V6\_WIDTH\_RESTRICTED\_ENVS* argument, as a <newline>-separated list of names suitable for use with the *getconf -v* option. If more than one environment meets the requirement, the names of all such environments shall be output on separate lines. Any of these names can then be used in a subsequent *getconf* command to obtain the flags specific to that environment with the following suffixes added as appropriate:

8479 \_CFLAGS To get the C compiler flags.

8480        \_LDFLAGS To get the linker/loader flags.  
 8481        \_LIBS      To get the libraries.  
 8482        This requirement may be removed in a future version.  
 8483        When this utility processes a file containing a function called *main()*, it shall be defined with a  
 8484        return type equivalent to **int**. Using return from the initial call to *main()* shall be equivalent  
 8485        (other than with respect to language scope issues) to calling *exit()* with the returned value.  
 8486        Reaching the end of the initial call to *main()* shall be equivalent to calling *exit(0)*. The  
 8487        implementation shall not declare a prototype for this function.  
 8488        Implementations provide configuration strings for C compiler flags, linker/loader flags, and  
 8489        libraries for each supported environment. When an application needs to use a specific  
 8490        programming environment rather than the implementation default programming environment  
 8491        while compiling, the application shall first verify that the implementation supports the desired  
 8492        environment. If the desired programming environment is supported, the application shall then  
 8493        invoke *c99* with the appropriate C compiler flags as the first options for the compile, the  
 8494        appropriate linker/loader flags after any other options but before any operands, and the  
 8495        appropriate libraries at the end of the operands.  
 8496        Conforming applications shall not attempt to link together object files compiled for different  
 8497        programming models. Applications shall also be aware that binary data placed in shared  
 8498        memory or in files might not be recognized by applications built for other programming models.

8499            **Table 4-5** Programming Environments: *c99* and *cc* Arguments

| 8500 <b>Programming Environment</b><br>8501 <i>getconf Name</i> | 8502 <b>Use</b>                                                  | 8503 <b><i>c99</i> and <i>cc</i> Arguments</b><br>8504 <i>getconf Name</i>                              |
|-----------------------------------------------------------------|------------------------------------------------------------------|---------------------------------------------------------------------------------------------------------|
| 8502        _POSIX_V6_ILP32_OFF32                               | 8503        C Compiler Flags<br>Linker/Loader Flags<br>Libraries | 8504        POSIX_V6_ILP32_OFF32_CFLAGS<br>POSIX_V6_ILP32_OFF32_LDFLAGS<br>POSIX_V6_ILP32_OFF32_LIBS    |
| 8505        _POSIX_V6_ILP32_OFFBIG                              | 8506        C Compiler Flags<br>Linker/Loader Flags<br>Libraries | 8507        POSIX_V6_ILP32_OFFBIG_CFLAGS<br>POSIX_V6_ILP32_OFFBIG_LDFLAGS<br>POSIX_V6_ILP32_OFFBIG_LIBS |
| 8508        _POSIX_V6_LP64_OFF64                                | 8509        C Compiler Flags<br>Linker/Loader Flags<br>Libraries | 8510        POSIX_V6_LP64_OFF64_CFLAGS<br>POSIX_V6_LP64_OFF64_LDFLAGS<br>POSIX_V6_LP64_OFF64_LIBS       |
| 8511        _POSIX_V6_LPBIG_OFFBIG                              | 8512        C Compiler Flags<br>Linker/Loader Flags<br>Libraries | 8513        POSIX_V6_LPBIG_OFFBIG_CFLAGS<br>POSIX_V6_LPBIG_OFFBIG_LDFLAGS<br>POSIX_V6_LPBIG_OFFBIG_LIBS |

8514        **EXIT STATUS**

8515        The following exit values shall be returned:  
 8516        0    Successful compilation or link edit.  
 8517        >0   An error occurred.

8518        **CONSEQUENCES OF ERRORS**

8519        When *c99* encounters a compilation error that causes an object file not to be created, it shall write  
 8520        a diagnostic to standard error and continue to compile other source code operands, but it shall  
 8521        not perform the link phase and return a non-zero exit status. If the link edit is unsuccessful, a  
 8522        diagnostic message shall be written to standard error and *c99* exits with a non-zero status. A  
 8523        conforming application shall rely on the exit status of *c99*, rather than on the existence or mode  
 8524        of the executable file.

8525 **APPLICATION USAGE**

8526 Since the *c99* utility usually creates files in the current directory during the compilation process,  
8527 it is typically necessary to run the *c99* utility in a directory in which a file can be created.

8528 On systems providing POSIX Conformance (see the Base Definitions volume of  
8529 IEEE Std 1003.1-2001, Chapter 2, Conformance), *c99* is required only with the C-Language  
8530 Development option; XSI-conformant systems always provide *c99*.

8531 Some historical implementations have created .o files when –c is not specified and more than  
8532 one source file is given. Since this area is left unspecified, the application cannot rely on .o files  
8533 being created, but it also must be prepared for any related .o files that already exist being deleted  
8534 at the completion of the link edit.

8535 Some historical implementations have permitted –L options to be interspersed with –I operands  
8536 on the command line. For an application to compile consistently on systems that do not behave  
8537 like this, it is necessary for a conforming application to supply all –L options before any of the –I  
8538 options.

8539 There is the possible implication that if a user supplies versions of the standard functions (before  
8540 they would be encountered by an implicit –I c or explicit –I m), that those versions would be  
8541 used in place of the standard versions. There are various reasons this might not be true  
8542 (functions defined as macros, manipulations for clean name space, and so on), so the existence of  
8543 files named in the same manner as the standard libraries within the –L directories is explicitly  
8544 stated to produce unspecified behavior.

8545 All of the functions specified in the System Interfaces volume of IEEE Std 1003.1-2001 may be  
8546 made visible by implementations when the Standard C Library is searched. Conforming  
8547 applications must explicitly request searching the other standard libraries when functions made  
8548 visible by those libraries are used.

8549 **EXAMPLES**

- 8550 1. The following usage example compiles **foo.c** and creates the executable file **foo**:

8551     *c99 -o foo foo.c*

8552 The following usage example compiles **foo.c** and creates the object file **foo.o**:

8553     *c99 -c foo.c*

8554 The following usage example compiles **foo.c** and creates the executable file **a.out**:

8555     *c99 foo.c*

8556 The following usage example compiles **foo.c**, links it with **bar.o**, and creates the executable  
8557 file **a.out**. It may also create and leave **foo.o**:

8558     *c99 foo.c bar.o*

- 8559 2. The following example shows how an application using threads interfaces can test for  
8560 support of and use a programming environment supporting 32-bit **int**, **long**, and **pointer**  
8561 types and an **off\_t** type using at least 64 bits:

```
8562 if [$(getconf _POSIX_V6_ILP32_OFFBIG) != "-1"]
8563 then
8564 c99 $(getconf POSIX_V6_ILP32_OFFBIG_CFLAGS) -D_XOPEN_SOURCE=600 \
8565 $(getconf POSIX_V6_ILP32_OFFBIG_LDFLAGS) foo.c -o foo \
8566 $(getconf POSIX_V6_ILP32_OFFBIG_LIBS) -l pthread
8567 else
8568 echo ILP32_OFFBIG programming environment not supported
```

```
8569 exit 1
8570 fi
```

3. The following examples clarify the use and interactions of **-L** options and **-l** operands.

Consider the case in which module **a.c** calls function *f()* in library **libQ.a**, and module **b.c** calls function *g()* in library **libp.a**. Assume that both libraries reside in **/a/b/c**. The command line to compile and link in the desired way is:

```
c99 -L /a/b/c main.o a.c -l Q b.c -l p
```

In this case the **-l Q** operand need only precede the first **-l p** operand, since both **libQ.a** and **libp.a** reside in the same directory.

Multiple **-L** operands can be used when library name collisions occur. Building on the previous example, suppose that the user wants to use a new **libp.a**, in **/a/a/a**, but still wants *f()* from **/a/b/c/libQ.a**:

```
c99 -L /a/a/a -L /a/b/c main.o a.c -l Q b.c -l p
```

In this example, the linker searches the **-L** options in the order specified, and finds **/a/a/a/libp.a** before **/a/b/c/libp.a** when resolving references for **b.c**. The order of the **-l** operands is still important, however.

4. The following example shows how an application can use a programming environment where the widths of the following types:

```
blksize_t, cc_t, mode_t, nfds_t, pid_t, ptrdiff_t, size_t, speed_t, ssize_t, suseconds_t,
tcflag_t, useconds_t, wchar_t, wint_t
```

are no greater than the width of type **long**:

```
First choose one of the listed environments ...
... if there are no additional constraints, the first one will do:
CENV=$(getconf POSIX_V6_WIDTH_RESTRICTED_ENVS | head -n 1) 2
... or, if an environment that supports large files is preferred, 2
look for names that contain "OFF64" or "OFFBIG". (This chooses
the last one in the list if none match.)
for CENV in $(getconf POSIX_V6_WIDTH_RESTRICTED_ENVS) 2
do
 case $CENV in
 OFF64 | *OFFBIG*) break ;;
 esac
done
The chosen environment name can now be used like this:
c99 ${getconf ${CENV}_CFLAGS} -D _POSIX_C_SOURCE=200112L \
${getconf ${CENV}_LDFLAGS} foo.c -o foo \
${getconf ${CENV}_LIBS}
```

## 606 RATIONALE

The **c99** utility is based on the **c89** utility originally introduced in the ISO POSIX-2:1993 standard.

Some of the changes from **c89** include the modification to the contents of the Standard Libraries section to account for new headers and options; for example, **<spawn.h>** added to the **-l rt** operand, and the **-l trace** operand added for the Tracing functions.

8611 FUTURE DIRECTIONS

8612 None.

**8613 SEE ALSO**

8614 Section 1.7.1.4 (on page 4), *ar*, *getconf*, *make*, *nm*, *strip*, *umask*, the System Interfaces volume of  
8615 IEEE Std 1003.1-2001, *exec*, *sysconf()*, the Base Definitions volume of IEEE Std 1003.1-2001,  
8616 Chapter 13, Headers

8617 CHANGE HISTORY

8618 First released in Issue 6. Included for alignment with the ISO/IEC 9899:1999 standard.

8619 IEEE Std 1003.1-2001/Cor 1-2002, item XCU/TC1/D6/12 is applied, correcting the EXTENDED  
8620 DESCRIPTION of **-l c** and **-l m**. Previously, the text did not take into account the presence of  
8621 the *c99* math headers. 1 1 1

8622 IEEE Std 1003.1-2001/Cor 1-2002, item XCU/TC1/D6/13 is applied, changing the reference to 1  
8623 the libxnet library to libxnet.a. 1

IEEE Std 1003.1-2001/Cor 2-2004, item XCU/TC2/D6/5 is applied, updating the OPTIONS section, so that the names of files contained in the directory specified by the -L option are not assumed to end in the .a suffix. The set of library prefixes is also updated.

IEEE Std 1003.1-2001/Cor 2-2004, item XCU/TC2/D6/6 is applied, removing the lead underscore from the POSIX\_V6\_WIDTH\_RESTRICTED\_ENVS variable in the EXTENDED DESCRIPTION and the EXAMPLES sections.

8630 **NAME**

8631        cal — print a calendar

8632 **SYNOPSIS**

8633 XSI        `cal [[month] year ]`

8634

8635 **DESCRIPTION**

8636        The *cal* utility shall write a calendar to standard output using the Julian calendar for dates from  
8637        January 1, 1 through September 2, 1752 and the Gregorian calendar for dates from September 14,  
8638        1752 through December 31, 9999 as though the Gregorian calendar had been adopted on  
8639        September 14, 1752.

8640 **OPTIONS**

8641        None.

8642 **OPERANDS**

8643        The following operands shall be supported:

8644        *month*        Specify the month to be displayed, represented as a decimal integer from 1  
8645        (January) to 12 (December). The default shall be the current month.

8646        *year*        Specify the year for which the calendar is displayed, represented as a decimal  
8647        integer from 1 to 9999. The default shall be the current year.

8648 **STDIN**

8649        Not used.

8650 **INPUT FILES**

8651        None.

8652 **ENVIRONMENT VARIABLES**

8653        The following environment variables shall affect the execution of *cal*:

8654        *LANG*        Provide a default value for the internationalization variables that are unset or null.  
8655        (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2,  
8656        Internationalization Variables for the precedence of internationalization variables  
8657        used to determine the values of locale categories.)

8658        *LC\_ALL*      If set to a non-empty string value, override the values of all the other  
8659        internationalization variables.

8660        *LC\_CTYPE*     Determine the locale for the interpretation of sequences of bytes of text data as  
8661        characters (for example, single-byte as opposed to multi-byte characters in  
8662        arguments).

8663        *LC\_MESSAGES*

8664        Determine the locale that should be used to affect the format and contents of  
8665        diagnostic messages written to standard error, and informative messages written  
8666        to standard output.

8667        *LC\_TIME*      Determine the format and contents of the calendar.

8668        *NLSPATH*     Determine the location of message catalogs for the processing of *LC\_MESSAGES*.

8669        *TZ*        Determine the timezone used to calculate the value of the current month.

**8670 ASYNCHRONOUS EVENTS**

8671 Default.

**8672 STDOUT**

8673 The standard output shall be used to display the calendar, in an unspecified format.

**8674 STDERR**

8675 The standard error shall be used only for diagnostic messages.

**8676 OUTPUT FILES**

8677 None.

**8678 EXTENDED DESCRIPTION**

8679 None.

**8680 EXIT STATUS**

8681 The following exit values shall be returned:

8682 0 Successful completion.

8683 >0 An error occurred.

**8684 CONSEQUENCES OF ERRORS**

8685 Default.

**8686 APPLICATION USAGE**

8687 Note that:

8688 cal 83

8689 refers to A.D. 83, not 1983.

**8690 EXAMPLES**

8691 None.

**8692 RATIONALE**

8693 None.

**8694 FUTURE DIRECTIONS**

8695 A future version of IEEE Std 1003.1-2001 may support locale-specific recognition of the date of  
8696 adoption of the Gregorian calendar.

**8697 SEE ALSO**

8698 None.

**8699 CHANGE HISTORY**

8700 First released in Issue 2.

**8701 Issue 6**

8702 The DESCRIPTION is updated to allow for traditional behavior for years before the adoption of  
8703 the Gregorian calendar.

**8704 NAME**

8705        cat — concatenate and print files

**8706 SYNOPSIS**

8707        cat [-u][*file* ...]

**8708 DESCRIPTION**

8709        The *cat* utility shall read files in sequence and shall write their contents to the standard output in  
8710        the same sequence.

**8711 OPTIONS**

8712        The *cat* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section 12.2,  
8713        Utility Syntax Guidelines.

8714        The following option shall be supported:

8715        **-u**        Write bytes from the input file to the standard output without delay as each is  
8716        read.

**8717 OPERANDS**

8718        The following operand shall be supported:

8719        *file*        A pathname of an input file. If no *file* operands are specified, the standard input  
8720        shall be used. If a *file* is ‘-’, the *cat* utility shall read from the standard input at  
8721        that point in the sequence. The *cat* utility shall not close and reopen standard input  
8722        when it is referenced in this way, but shall accept multiple occurrences of ‘-’ as a  
8723        *file* operand.

**8724 STDIN**

8725        The standard input shall be used only if no *file* operands are specified, or if a *file* operand is ‘-’.  
8726        See the INPUT FILES section.

**8727 INPUT FILES**

8728        The input files can be any file type.

**8729 ENVIRONMENT VARIABLES**

8730        The following environment variables shall affect the execution of *cat*:

8731        **LANG**        Provide a default value for the internationalization variables that are unset or null.  
8732        (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2,  
8733        Internationalization Variables for the precedence of internationalization variables  
8734        used to determine the values of locale categories.)

8735        **LC\_ALL**      If set to a non-empty string value, override the values of all the other  
8736        internationalization variables.

8737        **LC\_CTYPE**     Determine the locale for the interpretation of sequences of bytes of text data as  
8738        characters (for example, single-byte as opposed to multi-byte characters in  
8739        arguments).

**8740 LC\_MESSAGES**

8741        Determine the locale that should be used to affect the format and contents of  
8742        diagnostic messages written to standard error.

8743        **xsi NLSPATH**    Determine the location of message catalogs for the processing of *LC\_MESSAGES*.

**8744 ASYNCHRONOUS EVENTS**

8745        Default.

**8746 STDOUT**

8747     The standard output shall contain the sequence of bytes read from the input files. Nothing else  
8748     shall be written to the standard output.

**8749 STDERR**

8750     The standard error shall be used only for diagnostic messages.

**8751 OUTPUT FILES**

8752     None.

**8753 EXTENDED DESCRIPTION**

8754     None.

**8755 EXIT STATUS**

8756     The following exit values shall be returned:

8757         0 All input files were output successfully.

8758         >0 An error occurred.

**8759 CONSEQUENCES OF ERRORS**

8760     Default.

**8761 APPLICATION USAGE**

8762     The **-u** option has value in prototyping non-blocking reads from FIFOs. The intent is to support  
8763     the following sequence:

```
8764 mkfifo foo
8765 cat -u foo > /dev/tty13 &
8766 cat -u > foo
```

8767     It is unspecified whether standard output is or is not buffered in the default case. This is  
8768     sometimes of interest when standard output is associated with a terminal, since buffering may  
8769     delay the output. The presence of the **-u** option guarantees that unbuffered I/O is available. It is  
8770     implementation-defined whether the **cat** utility buffers output if the **-u** option is not specified.  
8771     Traditionally, the **-u** option is implemented using the equivalent of the **setvbuf()** function  
8772     defined in the System Interfaces volume of IEEE Std 1003.1-2001.

**8773 EXAMPLES**

8774     The following command:

```
8775 cat myfile
```

8776     writes the contents of the file **myfile** to standard output.

8777     The following command:

```
8778 cat doc1 doc2 > doc.all
```

8779     concatenates the files **doc1** and **doc2** and writes the result to **doc.all**.

8780     Because of the shell language mechanism used to perform output redirection, a command such  
8781     as this:

```
8782 cat doc doc.end > doc
```

8783     causes the original data in **doc** to be lost.

8784     The command:

```
8785 cat start - middle - end > file
```

8786 when standard input is a terminal, gets two arbitrary pieces of input from the terminal with a  
8787 single invocation of *cat*. Note, however, that if standard input is a regular file, this would be  
8788 equivalent to the command:

8789    *cat start - middle /dev/null end > file*

8790 because the entire contents of the file would be consumed by *cat* the first time '*-*' was used as a  
8791 *file* operand and an end-of-file condition would be detected immediately when '*-*' was  
8792 referenced the second time.

8793 **RATIONALE**

8794 Historical versions of the *cat* utility include the options **-e**, **-t**, and **-v**, which permit the ends of  
8795 lines, *<tab>*s, and invisible characters, respectively, to be rendered visible in the output. The  
8796 standard developers omitted these options because they provide too fine a degree of control  
8797 over what is made visible, and similar output can be obtained using a command such as:

8798    *sed -n -e 's/\$/\$/' -e l pathname*

8799 The **-s** option was omitted because it corresponds to different functions in BSD and System V-  
8800 based systems. The BSD **-s** option to squeeze blank lines can be accomplished by the shell script  
8801 shown in the following example:

```
8802 sed -n '
8803 # Write non-empty lines.
8804 ./ {
8805 p
8806 d
8807 }
8808 # Write a single empty line, then look for more empty lines.
8809 /^$/ p
8810 # Get next line, discard the held <newline> (empty line),
8811 # and look for more empty lines.
8812 :Empty
8813 /^$/ {
8814 N
8815 s/.//'
8816 b Empty
8817 }
8818 # Write the non-empty line before going back to search
8819 # for the first in a set of empty lines.
8820 p
8821 '
```

8822 The System V **-s** option to silence error messages can be accomplished by redirecting the  
8823 standard error. Note that the BSD documentation for *cat* uses the term "blank line" to mean the  
8824 same as the POSIX "empty line": a line consisting only of a *<newline>*.

8825 The BSD **-n** option was omitted because similar functionality can be obtained from the **-n**  
8826 option of the *pr* utility.

8827 **FUTURE DIRECTIONS**

8828 None.

8829 **SEE ALSO**

8830    *more*, the System Interfaces volume of IEEE Std 1003.1-2001, *setvbuf()*

8831 **CHANGE HISTORY**

8832 First released in Issue 2.

## 8833 NAME

8834        cd — change the working directory

## 8835 SYNOPSIS

8836        cd [-L | -P] [directory]

1

8837        cd -

1

## 8838 DESCRIPTION

8839        The *cd* utility shall change the working directory of the current shell execution environment (see  
8840        Section 2.12 (on page 61)) by executing the following steps in sequence. (In the following steps,  
8841        the symbol **curpath** represents an intermediate value used to simplify the description of the  
8842        algorithm used by *cd*. There is no requirement that **curpath** be made visible to the application.)

- 8843        1. If no *directory* operand is given and the *HOME* environment variable is empty or  
8844            undefined, the default behavior is implementation-defined and no further steps shall be  
8845            taken.
- 8846        2. If no *directory* operand is given and the *HOME* environment variable is set to a non-empty  
8847            value, the *cd* utility shall behave as if the directory named in the *HOME* environment  
8848            variable was specified as the *directory* operand.
- 8849        3. If the *directory* operand begins with a slash character, set **curpath** to the operand and  
8850            proceed to step 7.
- 8851        4. If the first component of the *directory* operand is dot or dot-dot, proceed to step 6.
- 8852        5. Starting with the first pathname in the colon-separated pathnames of *CDPATH* (see the  
8853            ENVIRONMENT VARIABLES section) if the pathname is non-null, test if the  
8854            concatenation of that pathname, a slash character, and the *directory* operand names a  
8855            directory. If the pathname is null, test if the concatenation of dot, a slash character, and the  
8856            operand names a directory. In either case, if the resulting string names an existing  
8857            directory, set **curpath** to that string and proceed to step 7. Otherwise, repeat this step with  
8858            the next pathname in *CDPATH* until all pathnames have been tested.
- 8859        6. Set **curpath** to the string formed by the concatenation of the value of *PWD*, a slash  
8860            character, and the operand.
- 8861        7. If the **-P** option is in effect, the *cd* utility shall perform actions equivalent to the *chdir()*  
8862            function, called with **curpath** as the *path* argument. If these actions succeed, the *PWD*  
8863            environment variable shall be set to an absolute pathname for the current working  
8864            directory and shall not contain filename components that, in the context of pathname  
8865            resolution, refer to a file of type symbolic link. If there is insufficient permission on the new  
8866            directory, or on any parent of that directory, to determine the current working directory,  
8867            the value of the *PWD* environment variable is unspecified. If the actions equivalent to  
8868            *chdir()* fail for any reason, the *cd* utility shall display an appropriate error message and not  
8869            alter the *PWD* environment variable. Whether the actions equivalent to *chdir()* succeed or  
8870            fail, no further steps shall be taken.
- 8871        8. The **curpath** value shall then be converted to canonical form as follows, considering each  
8872            component from beginning to end, in sequence:
  - 8873            a. Dot components and any slashes that separate them from the next component shall  
8874              be deleted.
  - 8875            b. For each dot-dot component, if there is a preceding component and it is neither root  
8876              nor dot-dot, the preceding component, all slashes separating the preceding  
8877              component from dot-dot, dot-dot and all slashes separating dot-dot from the  
8878              following component shall be deleted.

- 8879           c. An implementation may further simplify **curpath** by removing any trailing slash  
8880            characters that are not also leading slashes, replacing multiple non-leading  
8881            consecutive slashes with a single slash, and replacing three or more leading slashes  
8882            with a single slash. If, as a result of this canonicalization, the **curpath** variable is null,  
8883            no further steps shall be taken.
- 8884       9. The *cd* utility shall then perform actions equivalent to the *chdir()* function called with  
8885           **curpath** as the *path* argument. If these actions failed for any reason, the *cd* utility shall  
8886           display an appropriate error message and no further steps shall be taken. The *PWD*  
8887           environment variable shall be set to **curpath**.

8888       If, during the execution of the above steps, the *PWD* environment variable is changed, the  
8889           *OLDPWD* environment variable shall also be changed to the value of the old working directory  
8890           (that is the current working directory immediately prior to the call to *cd*).

## 8891     OPTIONS

8892       The *cd* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section 12.2,  
8893           Utility Syntax Guidelines.

8894       The following options shall be supported by the implementation:

- 8895       **-L**           Handle the operand dot-dot logically; symbolic link components shall not be  
8896            resolved before dot-dot components are processed (see steps 8. and 9. in the  
8897            DESCRIPTION).
- 8898       **-P**           Handle the operand dot-dot physically; symbolic link components shall be  
8899            resolved before dot-dot components are processed (see step 7. in the  
8900            DESCRIPTION).

8901       If both **-L** and **-P** options are specified, the last of these options shall be used and all others  
8902           ignored. If neither **-L** nor **-P** is specified, the operand shall be handled dot-dot logically; see the  
8903           DESCRIPTION.

## 8904     OPERANDS

8905       The following operands shall be supported:

- 8906       **directory**   An absolute or relative pathname of the directory that shall become the new  
8907            working directory. The interpretation of a relative pathname by *cd* depends on the  
8908           **-L** option and the *CDPATH* and *PWD* environment variables. If *directory* is an  
8909           empty string, the results are unspecified.
- 8910       **-**           When a hyphen is used as the operand, this shall be equivalent to the command:  
8911            

```
cd "$OLDPWD" && pwd
```

  
8912           which changes to the previous working directory and then writes its name.

## 8913     STDIN

8914       Not used.

## 8915     INPUT FILES

8916       None.

## 8917     ENVIRONMENT VARIABLES

8918       The following environment variables shall affect the execution of *cd*:

- 8919       **CDPATH**    A colon-separated list of pathnames that refer to directories. The *cd* utility shall use  
8920            this list in its attempt to change the directory, as described in the DESCRIPTION.  
8921            An empty string in place of a directory pathname represents the current directory.  
8922            If *CDPATH* is not set, it shall be treated as if it were an empty string.

|          |                               |                                                                                                                                                                                                                                                                                                       |
|----------|-------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 8923     | <i>HOME</i>                   | The name of the directory, used when no <i>directory</i> operand is specified.                                                                                                                                                                                                                        |
| 8924     | <i>LANG</i>                   | Provide a default value for the internationalization variables that are unset or null. (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2, Internationalization Variables for the precedence of internationalization variables used to determine the values of locale categories.) |
| 8928     | <i>LC_ALL</i>                 | If set to a non-empty string value, override the values of all the other internationalization variables.                                                                                                                                                                                              |
| 8930     | <i>LC_CTYPE</i>               | Determine the locale for the interpretation of sequences of bytes of text data as characters (for example, single-byte as opposed to multi-byte characters in arguments).                                                                                                                             |
| 8933     | <i>LC_MESSAGES</i>            | Determine the locale that should be used to affect the format and contents of diagnostic messages written to standard error.                                                                                                                                                                          |
| 8936 XSI | <i>NLSPATH</i>                | Determine the location of message catalogs for the processing of <i>LC_MESSAGES</i> .                                                                                                                                                                                                                 |
| 8937     | <i>OLDPWD</i>                 | A pathname of the previous working directory, used by <i>cd -</i> .                                                                                                                                                                                                                                   |
| 8938     | <i>PWD</i>                    | This variable shall be set as specified in the DESCRIPTION. If an application sets or unsets the value of <i>PWD</i> , the behavior of <i>cd</i> is unspecified.                                                                                                                                      |
| 8940     | <b>ASYNCHRONOUS EVENTS</b>    |                                                                                                                                                                                                                                                                                                       |
| 8941     | Default.                      |                                                                                                                                                                                                                                                                                                       |
| 8942     | <b>STDOUT</b>                 |                                                                                                                                                                                                                                                                                                       |
| 8943     |                               | If a non-empty directory name from <i>CDPATH</i> is used, or if <i>cd -</i> is used, an absolute pathname of the new working directory shall be written to the standard output as follows:                                                                                                            |
| 8945     |                               | "%s\n", <new directory>                                                                                                                                                                                                                                                                               |
| 8946     |                               | Otherwise, there shall be no output.                                                                                                                                                                                                                                                                  |
| 8947     | <b>STDERR</b>                 |                                                                                                                                                                                                                                                                                                       |
| 8948     |                               | The standard error shall be used only for diagnostic messages.                                                                                                                                                                                                                                        |
| 8949     | <b>OUTPUT FILES</b>           |                                                                                                                                                                                                                                                                                                       |
| 8950     |                               | None.                                                                                                                                                                                                                                                                                                 |
| 8951     | <b>EXTENDED DESCRIPTION</b>   |                                                                                                                                                                                                                                                                                                       |
| 8952     |                               | None.                                                                                                                                                                                                                                                                                                 |
| 8953     | <b>EXIT STATUS</b>            |                                                                                                                                                                                                                                                                                                       |
| 8954     |                               | The following exit values shall be returned:                                                                                                                                                                                                                                                          |
| 8955     | 0                             | The directory was successfully changed.                                                                                                                                                                                                                                                               |
| 8956     | >0                            | An error occurred.                                                                                                                                                                                                                                                                                    |
| 8957     | <b>CONSEQUENCES OF ERRORS</b> |                                                                                                                                                                                                                                                                                                       |
| 8958     |                               | The working directory shall remain unchanged.                                                                                                                                                                                                                                                         |

**8959 APPLICATION USAGE**

8960 Since *cd* affects the current shell execution environment, it is always provided as a shell regular  
8961 built-in. If it is called in a subshell or separate utility execution environment, such as one of the  
8962 following:

```
8963 (cd /tmp)
8964 nohup cd
8965 find . -exec cd {} \;
```

8966 it does not affect the working directory of the caller's environment.

8967 The user must have execute (search) permission in *directory* in order to change to it.

**8968 EXAMPLES**

8969 None.

**8970 RATIONALE**

8971 The use of the *CDPATH* was introduced in the System V shell. Its use is analogous to the use of the  
8972 *PATH* variable in the shell. The BSD C shell used a shell parameter *cpath* for this purpose.

8973 A common extension when *HOME* is undefined is to get the login directory from the user  
8974 database for the invoking user. This does not occur on System V implementations.

8975 Some historical shells, such as the KornShell, took special actions when the directory name  
8976 contained a dot-dot component, selecting the logical parent of the directory, rather than the  
8977 actual parent directory; that is, it moved up one level toward the ‘‘/’’ in the pathname,  
8978 remembering what the user typed, rather than performing the equivalent of:

```
8979 chdir("../");
```

8980 In such a shell, the following commands would not necessarily produce equivalent output for all  
8981 directories:

```
8982 cd .. && ls ls ..
```

8983 This behavior is now the default. It is not consistent with the definition of dot-dot in most  
8984 historical practice; that is, while this behavior has been optionally available in the KornShell,  
8985 other shells have historically not supported this functionality. The logical pathname is stored in  
8986 the *PWD* environment variable when the *cd* utility completes and this value is used to construct  
8987 the next directory name if *cd* is invoked with the *-L* option.

**8988 FUTURE DIRECTIONS**

8989 None.

**8990 SEE ALSO**

8991 Section 2.12 (on page 61), *pwd*, the System Interfaces volume of IEEE Std 1003.1-2001, *chdir()*

**8992 CHANGE HISTORY**

8993 First released in Issue 2.

**8994 Issue 6**

8995 The following new requirements on POSIX implementations derive from alignment with the  
8996 Single UNIX Specification:

- 8997 • The *cd* – operand, *PWD*, and *OLDPWD* are added.

8998 The *-L* and *-P* options are added to align with the IEEE P1003.2b draft standard. This also  
8999 includes the introduction of a new description to include the effect of these options.

9000 IEEE Std 1003.1-2001/Cor 1-2002, item XCU/TC1/D6/14 is applied, changing the SYNOPSIS to 1  
9001 make it clear that the **-L** and **-P** options are mutually-exclusive. 1

9002 **NAME**9003        cflow — generate a C-language flowgraph (**DEVELOPMENT**)9004 **SYNOPSIS**9005 XSI        cflow [-r][-d num][-D name[=def]] ... [-i incl][-I dir] ...  
9006            [-U dir] ... file ...

9007

9008 **DESCRIPTION**9009        The *cflow* utility shall analyze a collection of object files or assembler, C-language, *lex*, or *yacc*  
9010        source files, and attempt to build a graph, written to standard output, charting the external  
9011        references.9012 **OPTIONS**9013        The *cflow* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section  
9014        12.2, Utility Syntax Guidelines, except that the order of the **-D**, **-I**, and **-U** options (which are  
9015        identical to their interpretation by *c99*) is significant.

9016        The following options shall be supported:

9017        **-d num**        Indicate the depth at which the flowgraph is cut off. The application shall ensure  
9018        that the argument *num* is a decimal integer. By default this is a very large number  
9019        (typically greater than 32 000). Attempts to set the cut-off depth to a non-positive  
9020        integer shall be ignored.9021        **-i incl**        Increase the number of included symbols. The *incl* option-argument is one of the  
9022        following characters:9023            **x**        Include external and static data symbols. The default shall be to include only  
9024        functions in the flowgraph.  
9025            **\_**        (Underscore) Include names that begin with an underscore. The default shall  
9026        be to exclude these functions (and data if **-i x** is used).9027        **-r**        Reverse the caller:callee relationship, producing an inverted listing showing the  
9028        callers of each function. The listing shall also be sorted in lexicographical order by  
9029        callee.9030 **OPERANDS**

9031        The following operand is supported:

9032        **file**        The pathname of a file for which a graph is to be generated. Filenames suffixed by  
9033        **.l** shall shall be taken to be *lex* input, **.y** as *yacc* input, **.c** as *c99* input, and **.i** as the  
9034        output of *c99 -E*. Such files shall be processed as appropriate, determined by their  
9035        suffix.9036        Files suffixed by **.s** (conventionally assembler source) may have more limited  
9037        information extracted from them.9038 **STDIN**

9039        Not used.

9040 **INPUT FILES**9041        The input files shall be object files or assembler, C-language, *lex*, or *yacc* source files.9042 **ENVIRONMENT VARIABLES**9043        The following environment variables shall affect the execution of *cflow*:9044        **LANG**        Provide a default value for the internationalization variables that are unset or null.  
9045        (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2,

9046                   Internationalization Variables for the precedence of internationalization variables  
9047                   used to determine the values of locale categories.)

9048     *LC\_ALL*     If set to a non-empty string value, override the values of all the other  
9049                   internationalization variables.

9050     *LC\_COLLATE*  
9051                   Determine the locale for the ordering of the output when the **-r** option is used.

9052     *LC\_CTYPE*    Determine the locale for the interpretation of sequences of bytes of text data as  
9053                   characters (for example, single-byte as opposed to multi-byte characters in  
9054                   arguments and input files).

9055     *LC\_MESSAGES*  
9056                   Determine the locale that should be used to affect the format and contents of  
9057                   diagnostic messages written to standard error.

9058     *NLSPATH*    Determine the location of message catalogs for the processing of *LC\_MESSAGES*.

9059     **ASYNCHRONOUS EVENTS**  
9060                   Default.

9061     **STDOUT**  
9062                   The flowgraph written to standard output shall be formatted as follows:  
9063                   "*%d %s:%s\n*", *<reference number>*, *<global>*, *<definition>*

9064                   Each line of output begins with a reference (that is, line) number, followed by indentation of at  
9065                   least one column position per level. This is followed by the name of the global, a colon, and its  
9066                   definition. Normally globals are only functions not defined as an external or beginning with an  
9067                   underscore; see the OPTIONS section for the **-i** inclusion option. For information extracted from  
9068                   C-language source, the definition consists of an abstract type declaration (for example, **char \***)  
9069                   and, delimited by angle brackets, the name of the source file and the line number where the  
9070                   definition was found. Definitions extracted from object files indicate the filename and location  
9071                   counter under which the symbol appeared (for example, *text*).

9072                   Once a definition of a name has been written, subsequent references to that name contain only  
9073                   the reference number of the line where the definition can be found. For undefined references,  
9074                   only "*< >*" shall be written.

9075     **STDERR**  
9076                   The standard error shall be used only for diagnostic messages.

9077     **OUTPUT FILES**  
9078                   None.

9079     **EXTENDED DESCRIPTION**  
9080                   None.

9081     **EXIT STATUS**  
9082                   The following exit values shall be returned:  
9083                   0   Successful completion.  
9084                   >0   An error occurred.

9085     **CONSEQUENCES OF ERRORS**  
9086                   Default.

9087 **APPLICATION USAGE**

9088     Files produced by *lex* and *yacc* cause the reordering of line number declarations, and this can  
9089     confuse *cflow*. To obtain proper results, the input of *yacc* or *lex* must be directed to *cflow*.

9090 **EXAMPLES**

9091     Given the following in **file.c**:

```
9092 int i;
9093 int f();
9094 int g();
9095 int h();
9096 int
9097 main()
9098 {
9099 f();
9100 g();
9101 f();
9102 }
9103 int
9104 f()
9105 {
9106 i = h();
9107 }
```

9108     The command:

```
9109 cflow -i x file.c
```

9110     produces the output:

```
9111 1 main: int(), <file.c 6>
9112 2 f: int(), <file.c 13>
9113 3 h: <>
9114 4 i: int, <file.c 1>
9115 5 g: <>
```

9116 **RATIONALE**

9117     None.

9118 **FUTURE DIRECTIONS**

9119     None.

9120 **SEE ALSO**

9121     *c99*, *lex*, *yacc*

9122 **CHANGE HISTORY**

9123     First released in Issue 2.

9124 **Issue 6**

9125     The normative text is reworded to avoid use of the term “must” for application requirements.

9126 **NAME**

9127        chgrp — change the file group ownership

9128 **SYNOPSIS**9129        chgrp [-hR] *group file ...*

1

9130        chgrp -R [-H | -L | -P] *group file ...*

1

9131 **DESCRIPTION**9132        The *chgrp* utility shall set the group ID of the file named by each *file* operand to the group ID specified by the *group* operand.9134        For each *file* operand, or, if the **-R** option is used, each file encountered while walking the  
9135        directory trees specified by the *file* operands, the *chgrp* utility shall perform actions equivalent to  
9136        the *chown()* function defined in the System Interfaces volume of IEEE Std 1003.1-2001, called  
9137        with the following arguments:

- 9138        • The *file* operand shall be used as the *path* argument.
- 9139        • The user ID of the file shall be used as the *owner* argument.
- 9140        • The specified group ID shall be used as the *group* argument.

9141        Unless *chgrp* is invoked by a process with appropriate privileges, the set-user-ID and set-group-ID bits of a regular file shall be cleared upon successful completion; the set-user-ID and set-group-ID bits of other file types may be cleared.9144 **OPTIONS**9145        The *chgrp* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section  
9146        12.2, Utility Syntax Guidelines.

9147        The following options shall be supported by the implementation:

- 9148        **-h**        If the system supports group IDs for symbolic links, for each *file* operand that  
9149        names a file of type symbolic link, *chgrp* shall attempt to set the group ID of the  
9150        symbolic link instead of the file referenced by the symbolic link. If the system does  
9151        not support group IDs for symbolic links, for each *file* operand that names a file of  
9152        type symbolic link, *chgrp* shall do nothing more with the current file and shall go  
9153        on to any remaining files.
- 9154        **-H**        If the **-R** option is specified and a symbolic link referencing a file of type directory  
9155        is specified on the command line, *chgrp* shall change the group of the directory  
9156        referenced by the symbolic link and all files in the file hierarchy below it.
- 9157        **-L**        If the **-R** option is specified and a symbolic link referencing a file of type directory  
9158        is specified on the command line or encountered during the traversal of a file  
9159        hierarchy, *chgrp* shall change the group of the directory referenced by the symbolic  
9160        link and all files in the file hierarchy below it.
- 9161        **-P**        If the **-R** option is specified and a symbolic link is specified on the command line  
9162        or encountered during the traversal of a file hierarchy, *chgrp* shall change the  
9163        group ID of the symbolic link if the system supports this operation. The *chgrp*  
9164        utility shall not follow the symbolic link to any other part of the file hierarchy.
- 9165        **-R**        Recursively change file group IDs. For each *file* operand that names a directory,  
9166        *chgrp* shall change the group of the directory and all files in the file hierarchy below  
9167        it. Unless a **-H**, **-L**, or **-P** option is specified, it is unspecified which of these  
9168        options will be used as the default.

9169 Specifying more than one of the mutually-exclusive options **-H**, **-L**, and **-P** shall not be  
9170 considered an error. The last option specified shall determine the behavior of the utility.

## 9171 OPERANDS

9172 The following operands shall be supported:

9173 *group* A group name from the group database or a numeric group ID. Either specifies a  
9174 group ID to be given to each file named by one of the *file* operands. If a numeric  
9175 *group* operand exists in the group database as a group name, the group ID number  
9176 associated with that group name is used as the group ID.

9177 *file* A pathname of a file whose group ID is to be modified.

## 9178 STDIN

9179 Not used.

## 9180 INPUT FILES

9181 None.

## 9182 ENVIRONMENT VARIABLES

9183 The following environment variables shall affect the execution of *chgrp*:

9184 *LANG* Provide a default value for the internationalization variables that are unset or null.  
9185 (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2,  
9186 Internationalization Variables for the precedence of internationalization variables  
9187 used to determine the values of locale categories.)

9188 *LC\_ALL* If set to a non-empty string value, override the values of all the other  
9189 internationalization variables.

9190 *LC\_CTYPE* Determine the locale for the interpretation of sequences of bytes of text data as  
9191 characters (for example, single-byte as opposed to multi-byte characters in  
9192 arguments).

## 9193 *LC\_MESSAGES*

9194 Determine the locale that should be used to affect the format and contents of  
9195 diagnostic messages written to standard error.

9196 XSI *NLSPATH* Determine the location of message catalogs for the processing of *LC\_MESSAGES*.

## 9197 ASYNCHRONOUS EVENTS

9198 Default.

## 9199 STDOUT

9200 Not used.

## 9201 STDERR

9202 The standard error shall be used only for diagnostic messages.

## 9203 OUTPUT FILES

9204 None.

## 9205 EXTENDED DESCRIPTION

9206 None.

## 9207 EXIT STATUS

9208 The following exit values shall be returned:

9209 0 The utility executed successfully and all requested changes were made.

9210 >0 An error occurred.

9211 **CONSEQUENCES OF ERRORS**

9212 Default.

9213 **APPLICATION USAGE**

9214 Only the owner of a file or the user with appropriate privileges may change the owner or group of a file.

9216 Some implementations restrict the use of *chgrp* to a user with appropriate privileges when the group specified is not the effective group ID or one of the supplementary group IDs of the calling process.9219 **EXAMPLES**

9220 None.

9221 **RATIONALE**

9222 The System V and BSD versions use different exit status codes. Some implementations used the exit status as a count of the number of errors that occurred; this practice is unworkable since it can overflow the range of valid exit status values. The standard developers chose to mask these by specifying only 0 and &gt;0 as exit values.

9226 The functionality of *chgrp* is described substantially through references to *chown()*. In this way, there is no duplication of effort required for describing the interactions of permissions, multiple groups, and so on.9229 **FUTURE DIRECTIONS**

9230 None.

9231 **SEE ALSO**9232 *chmod*, *chown*, the System Interfaces volume of IEEE Std 1003.1-2001, *chown()*9233 **CHANGE HISTORY**

9234 First released in Issue 2.

9235 **Issue 6**9236 New options **-H**, **-L**, and **-P** are added to align with the IEEE P1003.2b draft standard. These options affect the processing of symbolic links.

9238 IEEE PASC Interpretation 1003.2 #172 is applied, changing the CONSEQUENCES OF ERRORS section to "Default."

9240 IEEE Std 1003.1-2001/Cor 1-2002, item XCU/TC1/D6/15 is applied, changing the SYNOPSIS to 1  
9241 make it clear that **-h** and **-R** are optional. 1

**9242 NAME**

9243        chmod — change the file modes

**9244 SYNOPSIS**

9245        chmod [-R] mode file ...

**9246 DESCRIPTION**

9247        The *chmod* utility shall change any or all of the file mode bits of the file named by each *file* operand in the way specified by the *mode* operand.

9249        It is implementation-defined whether and how the *chmod* utility affects any alternate or  
9250        additional file access control mechanism (see the Base Definitions volume of  
9251        IEEE Std 1003.1-2001, Section 4.4, File Access Permissions) being used for the specified file.

9252        Only a process whose effective user ID matches the user ID of the file, or a process with the  
9253        appropriate privileges, shall be permitted to change the file mode bits of a file.

**9254 OPTIONS**

9255        The *chmod* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section  
9256        12.2, Utility Syntax Guidelines.

9257        The following option shall be supported:

9258        **-R**        Recursively change file mode bits. For each *file* operand that names a directory,  
9259        *chmod* shall change the file mode bits of the directory and all files in the file  
9260        hierarchy below it.

**9261 OPERANDS**

9262        The following operands shall be supported:

9263        *mode*      Represents the change to be made to the file mode bits of each file named by one of  
9264        the *file* operands; see the EXTENDED DESCRIPTION section.

9265        *file*      A pathname of a file whose file mode bits shall be modified.

**9266 STDIN**

9267        Not used.

**9268 INPUT FILES**

9269        None.

**9270 ENVIRONMENT VARIABLES**

9271        The following environment variables shall affect the execution of *chmod*:

9272        *LANG*     Provide a default value for the internationalization variables that are unset or null.  
9273        (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2,  
9274        Internationalization Variables for the precedence of internationalization variables  
9275        used to determine the values of locale categories.)

9276        *LC\_ALL*   If set to a non-empty string value, override the values of all the other  
9277        internationalization variables.

9278        *LC\_CTYPE* Determine the locale for the interpretation of sequences of bytes of text data as  
9279        characters (for example, single-byte as opposed to multi-byte characters in  
9280        arguments).

9281        *LC\_MESSAGES*

9282        Determine the locale that should be used to affect the format and contents of  
9283        diagnostic messages written to standard error.

9284 XSI **NLSPATH** Determine the location of message catalogs for the processing of *LC\_MESSAGES*.

9285 **ASYNCHRONOUS EVENTS**

9286 Default.

9287 **STDOUT**

9288 Not used.

9289 **STDERR**

9290 The standard error shall be used only for diagnostic messages.

9291 **OUTPUT FILES**

9292 None.

9293 **EXTENDED DESCRIPTION**

9294 The *mode* operand shall be either a *symbolic\_mode* expression or a non-negative octal integer. The  
9295 *symbolic\_mode* form is described by the grammar later in this section.

9296 Each **clause** shall specify an operation to be performed on the current file mode bits of each *file*.  
9297 The operations shall be performed on each *file* in the order in which the **clauses** are specified.

9298 The **who** symbols **u**, **g**, and **o** shall specify the *user*, *group*, and *other* parts of the file mode bits,  
9299 respectively. A **who** consisting of the symbol **a** shall be equivalent to **ugo**.

9300 The **perm** symbols **r**, **w**, and **x** represent the *read*, *write*, and *execute/search* portions of file mode  
9301 bits, respectively. The **perm** symbol **s** shall represent the *set-user-ID-on-execution* (when **who**  
9302 contains or implies **u**) and *set-group-ID-on-execution* (when **who** contains or implies **g**) bits.

9303 The **perm** symbol **X** shall represent the execute/search portion of the file mode bits if the file is a  
9304 directory or if the current (unmodified) file mode bits have at least one of the execute bits  
9305 (**S\_IXUSR**, **S\_IXGRP**, or **S\_IOTH**) set. It shall be ignored if the file is not a directory and none of  
9306 the execute bits are set in the current file mode bits.

9307 The **permcopy** symbols **u**, **g**, and **o** shall represent the current permissions associated with the  
9308 *user*, *group*, and *other* parts of the file mode bits, respectively. For the remainder of this section,  
9309 **perm** refers to the non-terminals **perm** and **permcopy** in the grammar.

9310 If multiple **actionlists** are grouped with a single **wholist** in the grammar, each **actionlist** shall be  
9311 applied in the order specified with that **wholist**. The *op* symbols shall represent the operation  
9312 performed, as follows:

- 9313 + If **perm** is not specified, the '+' operation shall not change the file mode bits.

9314 If **who** is not specified, the file mode bits represented by **perm** for the owner, group, and  
9315 other permissions, except for those with corresponding bits in the file mode creation mask  
9316 of the invoking process, shall be set.

9317 Otherwise, the file mode bits represented by the specified **who** and **perm** values shall be set.

- 9318 - If **perm** is not specified, the '-' operation shall not change the file mode bits.

9319 If **who** is not specified, the file mode bits represented by **perm** for the owner, group, and  
9320 other permissions, except for those with corresponding bits in the file mode creation mask  
9321 of the invoking process, shall be cleared.

9322 Otherwise, the file mode bits represented by the specified **who** and **perm** values shall be  
9323 cleared.

- 9324 = Clear the file mode bits specified by the **who** value, or, if no **who** value is specified, all of the  
9325 file mode bits specified in this volume of IEEE Std 1003.1-2001.

9326 If **perm** is not specified, the '=' operation shall make no further modifications to the file  
 9327 mode bits.

9328 If **who** is not specified, the file mode bits represented by **perm** for the owner, group, and  
 9329 other permissions, except for those with corresponding bits in the file mode creation mask  
 9330 of the invoking process, shall be set.

9331 Otherwise, the file mode bits represented by the specified **who** and **perm** values shall be set.

9332 When using the symbolic mode form on a regular file, it is implementation-defined whether or  
 9333 not:

- 9334 • Requests to set the set-user-ID-on-execution or set-group-ID-on-execution bit when all  
 9335 execute bits are currently clear and none are being set are ignored.
- 9336 • Requests to clear all execute bits also clear the set-user-ID-on-execution and set-group-ID-  
 9337 on-execution bits.
- 9338 • Requests to clear the set-user-ID-on-execution or set-group-ID-on-execution bits when all  
 9339 execute bits are currently clear are ignored. However, if the command *ls -l file* writes an *s* in  
 9340 the position indicating that the set-user-ID-on-execution or set-group-ID-on-execution is set,  
 9341 the commands *chmod u-s file* or *chmod g-s file*, respectively, shall not be ignored.

9342 When using the symbolic mode form on other file types, it is implementation-defined whether  
 9343 or not requests to set or clear the set-user-ID-on-execution or set-group-ID-on-execution bits are  
 9344 honored.

9345 If the **who** symbol **o** is used in conjunction with the **perm** symbol **s** with no other **who** symbols  
 9346 being specified, the set-user-ID-on-execution and set-group-ID-on-execution bits shall not be  
 9347 modified. It shall not be an error to specify the **who** symbol **o** in conjunction with the **perm**  
 9348 symbol **s**.

9349 XSI The **perm** symbol **t** shall specify the S\_ISVTX bit. When used with a file of type directory, it can 1  
 9350 be used with the **who** symbol **a**, or with no **who** symbol. It shall not be an error to specify a **who** 1  
 9351 symbol of **u**, **g**, or **o** in conjunction with the **perm** symbol **t**, but the meaning of these 1  
 9352 combinations is unspecified. The effect when using the **perm** symbol **t** with any file type other 1  
 9353 than directory is unspecified. 1

9354 For an octal integer *mode* operand, the file mode bits shall be set absolutely.

9355 For each bit set in the octal number, the corresponding file permission bit shown in the following 1  
 9356 table shall be set; all other file permission bits shall be cleared. For regular files, for each bit set in 1  
 9357 the octal number corresponding to the set-user-ID-on-execution or the set-group-ID-on- 1  
 9358 execution, bits shown in the following table shall be set; if these bits are not set in the octal 1  
 9359 number, they are cleared. For other file types, it is implementation-defined whether or not 1  
 9360 requests to set or clear the set-user-ID-on-execution or set-group-ID-on-execution bits are 1  
 9361 honored.

| Octal           | Mode Bit | Octal       | Mode Bit | Octal       | Mode Bit | Octal       | Mode Bit |
|-----------------|----------|-------------|----------|-------------|----------|-------------|----------|
| <b>4000</b>     | S_ISUID  | <b>0400</b> | S_IRUSR  | <b>0040</b> | S_IRGRP  | <b>0004</b> | S_IROTH  |
| <b>2000</b>     | S_ISGID  | <b>0200</b> | S_IWUSR  | <b>0020</b> | S_IWGRP  | <b>0002</b> | S_IWOTH  |
| XSI <b>1000</b> | S_ISVTX  | <b>0100</b> | S_IXUSR  | <b>0010</b> | S_IXGRP  | <b>0001</b> | S_IXOTH  |

9366 When bits are set in the octal number other than those listed in the table above, the behavior is  
 9367 unspecified.

9368       **Grammar for chmod**

9369       The grammar and lexical conventions in this section describe the syntax for the *symbolic\_mode*  
9370       operand. The general conventions for this style of grammar are described in Section 1.10 (on  
9371       page 19). A valid *symbolic\_mode* can be represented as the non-terminal symbol *symbolic\_mode* in  
9372       the grammar. This formal syntax shall take precedence over the preceding text syntax  
9373       description.

9374       The lexical processing is based entirely on single characters. Implementations need not allow  
9375       <blank>s within the single argument being processed.

```
9376 %start symbolic_mode
9377 %%
9378 symbolic_mode : clause
9379 | symbolic_mode ',' clause
9380 ;
9381 clause : actionlist
9382 | wholist actionlist
9383 ;
9384 wholist : who
9385 | wholist who
9386 ;
9387 who : 'u' | 'g' | 'o' | 'a'
9388 ;
9389 actionlist : action
9390 | actionlist action
9391 ;
9392 action : op
9393 | op permplist
9394 | op permcopy
9395 ;
9396 permcopy : 'u' | 'g' | 'o'
9397 ;
9398 op : '+' | '-' | '='
9399 ;
9400 permplist : perm
9401 | perm permplist
9402 ;
9403 XSI perm : 'r' | 'w' | 'x' | 'X' | 's' | 't'
9404 ;
```

9405       **EXIT STATUS**

9406       The following exit values shall be returned:

- 9407       0    The utility executed successfully and all requested changes were made.  
9408       >0    An error occurred.

## 9409 CONSEQUENCES OF ERRORS

9410 Default.

## 9411 APPLICATION USAGE

9412 Some implementations of the *chmod* utility change the mode of a directory before the files in the  
 9413 directory when performing a recursive (-R option) change; others change the directory mode  
 9414 after the files in the directory. If an application tries to remove read or search permission for a  
 9415 file hierarchy, the removal attempt fails if the directory is changed first; on the other hand, trying  
 9416 to re-enable permissions to a restricted hierarchy fails if directories are changed last. Users  
 9417 should not try to make a hierarchy inaccessible to themselves.

9418 Some implementations of *chmod* never used the process' *umask* when changing modes; systems  
 9419 conformant with this volume of IEEE Std 1003.1-2001 do so when **who** is not specified. Note the  
 9420 difference between:

9421 `chmod a-w file`

9422 which removes all write permissions, and:

9423 `chmod -- -w file`

9424 which removes write permissions that would be allowed if **file** was created with the same  
 9425 *umask*.

9426 Conforming applications should never assume that they know how the set-user-ID and set-  
 9427 group-ID bits on directories are interpreted.

## 9428 EXAMPLES

| Mode         | Results                                                                                            |
|--------------|----------------------------------------------------------------------------------------------------|
| <i>a+=</i>   | Equivalent to <i>a+,a=</i> ; clears all file mode bits.                                            |
| <i>go+-w</i> | Equivalent to <i>go+,go-w</i> ; clears group and other write bits.                                 |
| <i>g=o-w</i> | Equivalent to <i>g=o,g-w</i> ; sets group bit to match other bits and then clears group write bit. |
| <i>g-r+w</i> | Equivalent to <i>g-r,g+w</i> ; clears group read bit and sets group write bit.                     |
| <i>uo=g</i>  | Sets owner bits to match group bits and sets other bits to match group bits.                       |

## 9439 RATIONALE

9440 The functionality of *chmod* is described substantially through references to concepts defined in  
 9441 the System Interfaces volume of IEEE Std 1003.1-2001. In this way, there is less duplication of  
 9442 effort required for describing the interactions of permissions. However, the behavior of this  
 9443 utility is not described in terms of the *chmod()* function from the System Interfaces volume of  
 9444 IEEE Std 1003.1-2001 because that specification requires certain side effects upon alternate file  
 9445 access control mechanisms that might not be appropriate, depending on the implementation.

9446 Implementations that support mandatory file and record locking as specified by the 1984  
 9447 /usr/group standard historically used the combination of set-group-ID bit set and group  
 9448 execute bit clear to indicate mandatory locking. This condition is usually set or cleared with the  
 9449 symbolic mode **perm** symbol **l** instead of the **perm** symbols **s** and **x** so that the mandatory  
 9450 locking mode is not changed without explicit indication that that was what the user intended.  
 9451 Therefore, the details on how the implementation treats these conditions must be defined in the  
 9452 documentation. This volume of IEEE Std 1003.1-2001 does not require mandatory locking (nor  
 9453 does the System Interfaces volume of IEEE Std 1003.1-2001), but does allow it as an extension.  
 9454 However, this volume of IEEE Std 1003.1-2001 does require that the *ls* and *chmod* utilities work

9455 consistently in this area. If *ls -l file* indicates that the set-group-ID bit is set, *chmod g-s file* must  
9456 clear it (assuming appropriate privileges exist to change modes).

9457 The System V and BSD versions use different exit status codes. Some implementations used the  
9458 exit status as a count of the number of errors that occurred; this practice is unworkable since it  
9459 can overflow the range of valid exit status values. This problem is avoided here by specifying  
9460 only 0 and >0 as exit values.

9461 The System Interfaces volume of IEEE Std 1003.1-2001 indicates that implementation-defined  
9462 restrictions may cause the S\_ISUID and S\_ISGID bits to be ignored. This volume of  
9463 IEEE Std 1003.1-2001 allows the *chmod* utility to choose to modify these bits before calling  
9464 *chmod()* (or some function providing equivalent capabilities) for non-regular files. Among other  
9465 things, this allows implementations that use the set-user-ID and set-group-ID bits on directories  
9466 to enable extended features to handle these extensions in an intelligent manner.

9467 The **X perm** symbol was adopted from BSD-based systems because it provides commonly  
9468 desired functionality when doing recursive (-R option) modifications. Similar functionality is  
9469 not provided by the *find* utility. Historical BSD versions of *chmod*, however, only supported **X**  
9470 with *op+*; it has been extended in this volume of IEEE Std 1003.1-2001 because it is also useful  
9471 with *op-*. (It has also been added for *op-* even though it duplicates **x**, in this case, because it is  
9472 intuitive and easier to explain.)

9473 The grammar was extended with the *permcopy* non-terminal to allow historical-practice forms of  
9474 symbolic modes like **o=u -g** (that is, set the “other” permissions to the permissions of “owner”  
9475 minus the permissions of “group”).

## 9476 FUTURE DIRECTIONS

9477 None.

## 9478 SEE ALSO

9479 *ls*, *umask*, the System Interfaces volume of IEEE Std 1003.1-2001, *chmod()*

## 9480 CHANGE HISTORY

9481 First released in Issue 2.

## 9482 Issue 6

9483 The following new requirements on POSIX implementations derive from alignment with the  
9484 Single UNIX Specification:

- 9485 • Octal modes have been kept and made mandatory despite being marked obsolescent in the  
9486 ISO POSIX-2:1993 standard.

9487 IEEE PASC Interpretation 1003.2 #172 is applied, changing the CONSEQUENCES OF ERRORS  
9488 section to “Default.”.

9489 The Open Group Base Resolution bwg2001-010 is applied, adding the description of the  
9490 S\_ISVTX bit and the **t perm** symbol as an XSI extension.

9491 IEEE Std 1003.1-2001/Cor 1-2002, item XCU/TC1/D6/16 is applied, changing the XSI shaded 1  
9492 text in the EXTENDED DESCRIPTION from: 1

9493     “The **perm** symbol **t** shall specify the S\_ISVTX bit and shall apply to directories only. The 1  
9494 effect when using it with any other file type is unspecified. It can be used with the **who** 1  
9495 symbols **o**, **a**, or with no **who** symbol. It shall not be an error to specify a **who** symbol of **u** or 1  
9496 **g** in conjunction with the **perm** symbol **t**; it shall be ignored for **u** and **g**.” 1

9497 to: 1

|      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |   |
|------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---|
| 9498 | “The <b>perm</b> symbol <b>t</b> shall specify the S_ISVTX bit. When used with a file of type directory, it can be used with the <b>who</b> symbol <b>a</b> , or with no <b>who</b> symbol. It shall not be an error to specify a <b>who</b> symbol of <b>u</b> , <b>g</b> , or <b>o</b> in conjunction with the <b>perm</b> symbol <b>t</b> , but the meaning of these combinations is unspecified. The effect when using the <b>perm</b> symbol <b>t</b> with any file type other than directory is unspecified.” | 1 |
| 9499 |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     | 1 |
| 9500 |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     | 1 |
| 9501 |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     | 1 |
| 9502 |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     | 1 |
| 9503 | This change is to permit historical behavior.                                                                                                                                                                                                                                                                                                                                                                                                                                                                       | 1 |

9504 **NAME**9505       *chown* — change the file ownership9506 **SYNOPSIS**9507       *chown* [-hR] *owner[:group]* *file* ...

1

9508       *chown* -R [-H | -L | -P] *owner[:group]* *file* ...

1

9509 **DESCRIPTION**9510       The *chown* utility shall set the user ID of the file named by each *file* operand to the user ID  
9511       specified by the *owner* operand.9512       For each *file* operand, or, if the **-R** option is used, each file encountered while walking the  
9513       directory trees specified by the *file* operands, the *chown* utility shall perform actions equivalent to  
9514       the *chown()* function defined in the System Interfaces volume of IEEE Std 1003.1-2001, called  
9515       with the following arguments:

- 9516       1. The *file* operand shall be used as the *path* argument.
- 9517       2. The user ID indicated by the *owner* portion of the first operand shall be used as the *owner*  
9518       argument.
- 9519       3. If the *group* portion of the first operand is given, the group ID indicated by it shall be used  
9520       as the *group* argument; otherwise, the group ownership shall not be changed.

9521       Unless *chown* is invoked by a process with appropriate privileges, the set-user-ID and set-  
9522       group-ID bits of a regular file shall be cleared upon successful completion; the set-user-ID and  
9523       set-group-ID bits of other file types may be cleared.9524 **OPTIONS**9525       The *chown* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section  
9526       12.2, Utility Syntax Guidelines.

9527       The following options shall be supported by the implementation:

- |                |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
|----------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 9528 <b>-h</b> | If the system supports user IDs for symbolic links, for each <i>file</i> operand that names<br>9529       a file of type symbolic link, <i>chown</i> shall attempt to set the user ID of the symbolic<br>9530       link. If the system supports group IDs for symbolic links, and a group ID was<br>9531       specified, for each <i>file</i> operand that names a file of type symbolic link, <i>chown</i> shall<br>9532       attempt to set the group ID of the symbolic link. If the system does not support<br>9533       user or group IDs for symbolic links, for each <i>file</i> operand that names a file of type<br>9534       symbolic link, <i>chown</i> shall do nothing more with the current file and shall go on to<br>9535       any remaining files. |
| 9536 <b>-H</b> | If the <b>-R</b> option is specified and a symbolic link referencing a file of type directory<br>9537       is specified on the command line, <i>chown</i> shall change the user ID (and group ID, if<br>9538       specified) of the directory referenced by the symbolic link and all files in the file<br>9539       hierarchy below it.                                                                                                                                                                                                                                                                                                                                                                                                                               |
| 9540 <b>-L</b> | If the <b>-R</b> option is specified and a symbolic link referencing a file of type directory<br>9541       is specified on the command line or encountered during the traversal of a file<br>9542       hierarchy, <i>chown</i> shall change the user ID (and group ID, if specified) of the<br>9543       directory referenced by the symbolic link and all files in the file hierarchy below it.                                                                                                                                                                                                                                                                                                                                                                       |
| 9544 <b>-P</b> | If the <b>-R</b> option is specified and a symbolic link is specified on the command line<br>9545       or encountered during the traversal of a file hierarchy, <i>chown</i> shall change the<br>9546       owner ID (and group ID, if specified) of the symbolic link if the system supports<br>9547       this operation. The <i>chown</i> utility shall not follow the symbolic link to any other<br>9548       part of the file hierarchy.                                                                                                                                                                                                                                                                                                                           |

9549           **-R**           Recursively change file user and group IDs. For each *file* operand that names a  
 9550            directory, *chown* shall change the user ID (and group ID, if specified) of the  
 9551            directory and all files in the file hierarchy below it. Unless a **-H**, **-L**, or **-P** option is  
 9552            specified, it is unspecified which of these options will be used as the default.

9553           Specifying more than one of the mutually-exclusive options **-H**, **-L**, and **-P** shall not be  
 9554           considered an error. The last option specified shall determine the behavior of the utility.

## 9555 OPERANDS

9556           The following operands shall be supported:

9557           *owner[:group]* A user ID and optional group ID to be assigned to *file*. The *owner* portion of this  
 9558            operand shall be a user name from the user database or a numeric user ID. Either  
 9559            specifies a user ID which shall be given to each file named by one of the *file*  
 9560            operands. If a numeric *owner* operand exists in the user database as a user name,  
 9561            the user ID number associated with that user name shall be used as the user ID.  
 9562            Similarly, if the *group* portion of this operand is present, it shall be a group name  
 9563            from the group database or a numeric group ID. Either specifies a group ID which  
 9564            shall be given to each file. If a numeric group operand exists in the group database  
 9565            as a group name, the group ID number associated with that group name shall be  
 9566            used as the group ID.

9567           *file*           A pathname of a file whose user ID is to be modified.

## 9568 STDIN

9569           Not used.

## 9570 INPUT FILES

9571           None.

## 9572 ENVIRONMENT VARIABLES

9573           The following environment variables shall affect the execution of *chown*:

9574           *LANG*           Provide a default value for the internationalization variables that are unset or null.  
 9575           (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2,  
 9576           Internationalization Variables for the precedence of internationalization variables  
 9577           used to determine the values of locale categories.)

9578           *LC\_ALL*       If set to a non-empty string value, override the values of all the other  
 9579           internationalization variables.

9580           *LC\_CTYPE*      Determine the locale for the interpretation of sequences of bytes of text data as  
 9581           characters (for example, single-byte as opposed to multi-byte characters in  
 9582           arguments).

## 9583 *LC\_MESSAGES*

9584           Determine the locale that should be used to affect the format and contents of  
 9585           diagnostic messages written to standard error.

9586           XSI           *NLSPATH*      Determine the location of message catalogs for the processing of *LC\_MESSAGES*.

## 9587 ASYNCHRONOUS EVENTS

9588           Default.

## 9589 STDOUT

9590           Not used.

**9591 STDRERR**

9592     The standard error shall be used only for diagnostic messages.

**9593 OUTPUT FILES**

9594     None.

**9595 EXTENDED DESCRIPTION**

9596     None.

**9597 EXIT STATUS**

9598     The following exit values shall be returned:

9599        0    The utility executed successfully and all requested changes were made.

9600        >0   An error occurred.

**9601 CONSEQUENCES OF ERRORS**

9602     Default.

**9603 APPLICATION USAGE**

9604     Only the owner of a file or the user with appropriate privileges may change the owner or group  
9605     of a file.

9606     Some implementations restrict the use of *chown* to a user with appropriate privileges.

**9607 EXAMPLES**

9608     None.

**9609 RATIONALE**

9610     The System V and BSD versions use different exit status codes. Some implementations used the  
9611     exit status as a count of the number of errors that occurred; this practice is unworkable since it  
9612     can overflow the range of valid exit status values. These are masked by specifying only 0 and >0  
9613     as exit values.

9614     The functionality of *chown* is described substantially through references to functions in the  
9615     System Interfaces volume of IEEE Std 1003.1-2001. In this way, there is no duplication of effort  
9616     required for describing the interactions of permissions, multiple groups, and so on.

9617     The 4.3 BSD method of specifying both owner and group was included in this volume of  
9618     IEEE Std 1003.1-2001 because:

- 9619       • There are cases where the desired end condition could not be achieved using the *chgrp* and  
9620           *chown* (that only changed the user ID) utilities. (If the current owner is not a member of the  
9621           desired group and the desired owner is not a member of the current group, the *chown()*  
9622           function could fail unless both owner and group are changed at the same time.)
- 9623       • Even if they could be changed independently, in cases where both are being changed, there is  
9624           a 100% performance penalty caused by being forced to invoke both utilities.

9625     The BSD syntax *user[group]* was changed to *user[:group]* in this volume of IEEE Std 1003.1-2001  
9626     because the period is a valid character in login names (as specified by the Base Definitions  
9627     volume of IEEE Std 1003.1-2001, login names consist of characters in the portable filename  
9628     character set). The colon character was chosen as the replacement for the period character  
9629     because it would never be allowed as a character in a user name or group name on historical  
9630     implementations.

9631     The **-R** option is considered by some observers as an undesirable departure from the historical  
9632     UNIX system tools approach; since a tool, *find*, already exists to recurse over directories, there  
9633     seemed to be no good reason to require other tools to have to duplicate that functionality.  
9634     However, the **-R** option was deemed an important user convenience, is far more efficient than

9635 forking a separate process for each element of the directory hierarchy, and is in widespread  
9636 historical use.

9637 **FUTURE DIRECTIONS**

9638 None.

9639 **SEE ALSO**

9640 *chmod*, *chgrp*, the System Interfaces volume of IEEE Std 1003.1-2001, *chown()*

9641 **CHANGE HISTORY**

9642 First released in Issue 2.

9643 **Issue 6**

9644 New options **-h**, **-H**, **-L**, and **-P** are added to align with the IEEE P1003.2b draft standard. These  
9645 options affect the processing of symbolic links.

9646 The normative text is reworded to avoid use of the term “must” for application requirements.

9647 IEEE PASC Interpretation 1003.2 #172 is applied, changing the CONSEQUENCES OF ERRORS  
9648 section to “Default.”.

9649 The “otherwise, …” text in item 3. of the DESCRIPTION is changed to “otherwise, the group  
9650 ownership shall not be changed”.

9651 IEEE Std 1003.1-2001/Cor 1-2002, item XCU/TC1/D6/17 is applied, changing the SYNOPSIS to 1  
9652 make it clear that **-h** and **-R** are optional. 1

9653 **NAME**

9654        cksum — write file checksums and sizes

9655 **SYNOPSIS**9656        cksum [*file* ...]9657 **DESCRIPTION**

9658        The *cksum* utility shall calculate and write to standard output a cyclic redundancy check (CRC)  
9659        for each input file, and also write to standard output the number of octets in each file. The CRC  
9660        used is based on the polynomial used for CRC error checking in the ISO/IEC 8802-3:1996  
9661        standard (Ethernet).

9662        The encoding for the CRC checksum is defined by the generating polynomial:

$$G(x) = x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^8 + x^7 + x^5 + x^4 + x^2 + x + 1$$

9664        Mathematically, the CRC value corresponding to a given file shall be defined by the following  
9665        procedure:

- 9666        1. The *n* bits to be evaluated are considered to be the coefficients of a mod 2 polynomial *M(x)*  
9667        of degree *n*-1. These *n* bits are the bits from the file, with the most significant bit being the  
9668        most significant bit of the first octet of the file and the last bit being the least significant bit  
9669        of the last octet, padded with zero bits (if necessary) to achieve an integral number of  
9670        octets, followed by one or more octets representing the length of the file as a binary value,  
9671        least significant octet first. The smallest number of octets capable of representing this  
9672        integer shall be used.
- 9673        2. *M(x)* is multiplied by  $x^{32}$  (that is, shifted left 32 bits) and divided by *G(x)* using mod 2  
9674        division, producing a remainder *R(x)* of degree  $\leq 31$ .
- 9675        3. The coefficients of *R(x)* are considered to be a 32-bit sequence.
- 9676        4. The bit sequence is complemented and the result is the CRC.

9677 **OPTIONS**

9678        None.

9679 **OPERANDS**

9680        The following operand shall be supported:

9681        *file*        A pathname of a file to be checked. If no *file* operands are specified, the standard  
9682        input shall be used.

9683 **STDIN**

9684        The standard input shall be used only if no *file* operands are specified. See the INPUT FILES  
9685        section.

9686 **INPUT FILES**

9687        The input files can be any file type.

9688 **ENVIRONMENT VARIABLES**9689        The following environment variables shall affect the execution of *cksum*:

9690        *LANG*        Provide a default value for the internationalization variables that are unset or null.  
9691        (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2,  
9692        Internationalization Variables for the precedence of internationalization variables  
9693        used to determine the values of locale categories.)

9694        *LC\_ALL*      If set to a non-empty string value, override the values of all the other  
9695        internationalization variables.

|      |                               |                                                                                                                                                                                                                                                                                                                                                                                                                       |
|------|-------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 9696 | <i>LC_CTYPE</i>               | Determine the locale for the interpretation of sequences of bytes of text data as characters (for example, single-byte as opposed to multi-byte characters in arguments).                                                                                                                                                                                                                                             |
| 9699 | <i>LC_MESSAGES</i>            | Determine the locale that should be used to affect the format and contents of diagnostic messages written to standard error.                                                                                                                                                                                                                                                                                          |
| 9702 | XSI <i>NLSPATH</i>            | Determine the location of message catalogs for the processing of <i>LC_MESSAGES</i> .                                                                                                                                                                                                                                                                                                                                 |
| 9703 | <b>ASYNCHRONOUS EVENTS</b>    |                                                                                                                                                                                                                                                                                                                                                                                                                       |
| 9704 |                               | Default.                                                                                                                                                                                                                                                                                                                                                                                                              |
| 9705 | <b>STDOUT</b>                 |                                                                                                                                                                                                                                                                                                                                                                                                                       |
| 9706 |                               | For each file processed successfully, the <i>cksum</i> utility shall write in the following format:                                                                                                                                                                                                                                                                                                                   |
| 9707 |                               | "%u %d %s\n", <checksum>, <# of octets>, <pathname>                                                                                                                                                                                                                                                                                                                                                                   |
| 9708 |                               | If no <i>file</i> operand was specified, the pathname and its leading <space> shall be omitted.                                                                                                                                                                                                                                                                                                                       |
| 9709 | <b>STDERR</b>                 |                                                                                                                                                                                                                                                                                                                                                                                                                       |
| 9710 |                               | The standard error shall be used only for diagnostic messages.                                                                                                                                                                                                                                                                                                                                                        |
| 9711 | <b>OUTPUT FILES</b>           |                                                                                                                                                                                                                                                                                                                                                                                                                       |
| 9712 |                               | None.                                                                                                                                                                                                                                                                                                                                                                                                                 |
| 9713 | <b>EXTENDED DESCRIPTION</b>   |                                                                                                                                                                                                                                                                                                                                                                                                                       |
| 9714 |                               | None.                                                                                                                                                                                                                                                                                                                                                                                                                 |
| 9715 | <b>EXIT STATUS</b>            |                                                                                                                                                                                                                                                                                                                                                                                                                       |
| 9716 |                               | The following exit values shall be returned:                                                                                                                                                                                                                                                                                                                                                                          |
| 9717 |                               | 0 All files were processed successfully.                                                                                                                                                                                                                                                                                                                                                                              |
| 9718 |                               | >0 An error occurred.                                                                                                                                                                                                                                                                                                                                                                                                 |
| 9719 | <b>CONSEQUENCES OF ERRORS</b> |                                                                                                                                                                                                                                                                                                                                                                                                                       |
| 9720 |                               | Default.                                                                                                                                                                                                                                                                                                                                                                                                              |
| 9721 | <b>APPLICATION USAGE</b>      |                                                                                                                                                                                                                                                                                                                                                                                                                       |
| 9722 |                               | The <i>cksum</i> utility is typically used to quickly compare a suspect file against a trusted version of the same, such as to ensure that files transmitted over noisy media arrive intact. However, this comparison cannot be considered cryptographically secure. The chances of a damaged file producing the same CRC as the original are small; deliberate deception is difficult, but probably not impossible.  |
| 9727 |                               | Although input files to <i>cksum</i> can be any type, the results need not be what would be expected on character special device files or on file types not described by the System Interfaces volume of IEEE Std 1003.1-2001. Since this volume of IEEE Std 1003.1-2001 does not specify the block size used when doing input, checksums of character special files need not process all of the data in those files. |
| 9732 |                               | The algorithm is expressed in terms of a bitstream divided into octets. If a file is transmitted between two systems and undergoes any data transformation (such as changing little-endian byte ordering to big-endian), identical CRC values cannot be expected. Implementations performing such transformations may extend <i>cksum</i> to handle such situations.                                                  |

9736 **EXAMPLES**

9737 None.

9738 **RATIONALE**

9739 The following C-language program can be used as a model to describe the algorithm. It assumes  
9740 that a **char** is one octet. It also assumes that the entire file is available for one pass through the  
9741 function. This was done for simplicity in demonstrating the algorithm, rather than as an  
9742 implementation model.

```
9743 static unsigned long crctab[] = {
9744 0x00000000,
9745 0x04c11db7, 0x09823b6e, 0x0d4326d9, 0x130476dc, 0x17c56b6b,
9746 0x1a864db2, 0x1e475005, 0x2608edb8, 0x22c9f00f, 0x2f8ad6d6,
9747 0x2b4bcb61, 0x350c9b64, 0x31cd86d3, 0x3c8ea00a, 0x384fbdbd,
9748 0x4c11db70, 0x48d0c6c7, 0x4593e01e, 0x4152fda9, 0x5f15adac,
9749 0x5bd4b01b, 0x569796c2, 0x52568b75, 0x6a1936c8, 0x6ed82b7f,
9750 0x639b0da6, 0x675a1011, 0x791d4014, 0x7ddc5da3, 0x709f7b7a,
9751 0x745e66cd, 0x9823b6e0, 0x9ce2ab57, 0x91a18d8e, 0x95609039,
9752 0x8b27c03c, 0x8fe6dd8b, 0x82a5fb52, 0x8664e6e5, 0xbe2b5b58,
9753 0xbaea46ef, 0xb7a96036, 0xb3687d81, 0xad2f2d84, 0xa9ee3033,
9754 0xa4ad16ea, 0xa06c0b5d, 0xd4326d90, 0xd0f37027, 0xddb056fe,
9755 0xd9714b49, 0xc7361b4c, 0xc3f706fb, 0xceb42022, 0xca753d95,
9756 0xf23a8028, 0xf6fb9d9f, 0xfbb8bb46, 0xff79a6f1, 0xe13ef6f4,
9757 0xe5ffeb43, 0xe8bccd9a, 0xec7ddd02d, 0x34867077, 0x30476dc0,
9758 0x3d044b19, 0x39c556ae, 0x278206ab, 0x23431b1c, 0x2e003dc5,
9759 0x2ac12072, 0x128e9dcf, 0x164f8078, 0x1b0ca6a1, 0x1fcdbb16,
9760 0x018aeb13, 0x054bf6a4, 0x0808d07d, 0x0cc9cdca, 0x7897ab07,
9761 0x7c56b6b0, 0x71159069, 0x75d48dde, 0x6b93dddb, 0x6f52c06c,
9762 0x6211e6b5, 0x66d0fb02, 0x5e9f46bf, 0x5a5e5b08, 0x571d7dd1,
9763 0x53dc6066, 0x4d9b3063, 0x495a2dd4, 0x44190b0d, 0x40d816ba,
9764 0xacaca5c697, 0xa864db20, 0xa527fdf9, 0xa1e6e04e, 0xbfa1b04b,
9765 0xbb60adfc, 0xb6238b25, 0xb2e29692, 0x8aad2b2f, 0x8e6c3698,
9766 0x832f1041, 0x87ee0df6, 0x99a95df3, 0x9d684044, 0x902b669d,
9767 0x94ea7b2a, 0xe0b41de7, 0xe4750050, 0xe9362689, 0xedf73b3e,
9768 0xf3b06b3b, 0xf771768c, 0xfa325055, 0xefef34de2, 0xc6bcf05f,
9769 0xc27dede8, 0xcf3ecb31, 0xcbffd686, 0xd5b88683, 0xd1799b34,
9770 0xdc3abded, 0xd8fba05a, 0x690ce0ee, 0x6dcdf59, 0x608edb80,
9771 0x644fc637, 0x7a089632, 0x7ec98b85, 0x738aad5c, 0x774bb0eb,
9772 0x4f040d56, 0x4bc510e1, 0x46863638, 0x42472b8f, 0x5c007b8a,
9773 0x58c1663d, 0x558240e4, 0x51435d53, 0x251d3b9e, 0x21dc2629,
9774 0x2c9f00f0, 0x285e1d47, 0x36194d42, 0x32d850f5, 0x3f9b762c,
9775 0x3b5a6b9b, 0x0315d626, 0x07d4cb91, 0xa97ed48, 0xe56f0ff,
9776 0x1011a0fa, 0x14d0bd4d, 0x19939b94, 0x1d528623, 0xf12f560e,
9777 0xf5ee4bb9, 0xf8ad6d60, 0xfc6c70d7, 0xe22b20d2, 0xe6ea3d65,
9778 0xeba91bbc, 0xef68060b, 0xd727bbb6, 0xd3e6a601, 0xdeaa580d8,
9779 0xda649d6f, 0xc423cd6a, 0xc0e2d0dd, 0xcdaf604, 0xc960ebb3,
9780 0xbd3e8d7e, 0xb9ff90c9, 0xb4bcb610, 0xb07daba7, 0xae3afba2,
9781 0xaafbe615, 0xa7b8c0cc, 0xa379dd7b, 0x9b3660c6, 0x9ff77d71,
9782 0x92b45ba8, 0x9675461f, 0x8832161a, 0x8cf30bad, 0x81b02d74,
9783 0x857130c3, 0x5d8a9099, 0x594b8d2e, 0x5408abf7, 0x50c9b640,
9784 0x4e8ee645, 0x4a4ffbf2, 0x470cdd2b, 0x43cdc09c, 0x7b827d21,
9785 0x7f436096, 0x7200464f, 0x76c15bf8, 0x68860bfd, 0x6c47164a,
9786 0x61043093, 0x65c52d24, 0x119b4be9, 0x155a565e, 0x18197087,
```

```

9787 0x1cd86d30, 0x029f3d35, 0x065e2082, 0x0b1d065b, 0x0fdc1bec,
9788 0x3793a651, 0x3352bbe6, 0x3e119d3f, 0x3ad08088, 0x2497d08d,
9789 0x2056cd3a, 0x2d15ebe3, 0x29d4f654, 0xc5a92679, 0xc1683bce,
9790 0xcc2b1d17, 0xc8ea00a0, 0xd6ad50a5, 0xd26c4d12, 0xdf2f6bcb,
9791 0xdbbe767c, 0xe3a1cbc1, 0xe760d676, 0xea23f0af, 0xeee2ed18,
9792 0xf0a5bd1d, 0xf464a0aa, 0xf9278673, 0xfde69bc4, 0x89b8fd09,
9793 0x8d79e0be, 0x803ac667, 0x84fbdbd0, 0x9abc8bd5, 0x9e7d9662,
9794 0x933eb0bb, 0x97ffad0c, 0xafb010b1, 0xab710d06, 0xa6322bdf,
9795 0xa2f33668, 0xbcb4666d, 0xb8757bda, 0xb5365d03, 0xb1f740b4
9796 };

9797 unsigned long memcrc(const unsigned char *b, size_t n)
9798 {
9799 /* Input arguments:
9800 * const char* b == byte sequence to checksum
9801 * size_t n == length of sequence
9802 */
9803
9804 register unsigned i, c, s = 0;
9805
9806 for (i = n; i > 0; --i) {
9807 c = (unsigned)(*b++);
9808 s = (s << 8) ^ crctab[(s >> 24) ^ c];
9809 }
9810
9811 /* Extend with the length of the string. */
9812 while (n != 0) {
9813 c = n & 0377;
9814 n >>= 8;
9815 s = (s << 8) ^ crctab[(s >> 24) ^ c];
9816 }
9817
9818 return ~s;
9819 }

```

The historical practice of writing the number of “blocks” has been changed to writing the number of octets, since the latter is not only more useful, but also since historical implementations have not been consistent in defining what a “block” meant. Octets are used instead of bytes because bytes can differ in size between systems.

The algorithm used was selected to increase the operational robustness of *cksum*. Neither the System V nor BSD *sum* algorithm was selected. Since each of these was different and each was the default behavior on those systems, no realistic compromise was available if either were selected—some set of historical applications would break. Therefore, the name was changed to *cksum*. Although the historical *sum* commands will probably continue to be provided for many years, programs designed for portability across systems should use the new name.

The algorithm selected is based on that used by the ISO/IEC 8802-3:1996 standard (Ethernet) for the frame check sequence field. The algorithm used does not match the technical definition of a *checksum*; the term is used for historical reasons. The length of the file is included in the CRC calculation because this parallels inclusion of a length field by Ethernet in its CRC, but also because it guards against inadvertent collisions between files that begin with different series of zero octets. The chance that two different files produce identical CRCs is much greater when their lengths are not considered. Keeping the length and the checksum of the file itself separate would yield a slightly more robust algorithm, but historical usage has always been that a single number (the checksum as printed) represents the signature of the file. It was decided that

9835 historical usage was the more important consideration.  
9836 Early proposals contained modifications to the Ethernet algorithm that involved extracting table  
9837 values whenever an intermediate result became zero. This was demonstrated to be less robust  
9838 than the current method and mathematically difficult to describe or justify.

9839 The calculation used is identical to that given in pseudo-code in the referenced Sarwate article.  
9840 The pseudo-code rendition is:

```
9841 X <- 0; Y <- 0;
9842 for i <- m -1 step -1 until 0 do
9843 begin
9844 T <- X(1) ^ A[i];
9845 X(1) <- X(0); X(0) <- Y(1); Y(1) <- Y(0); Y(0) <- 0;
9846 comment: f[T] and f'[T] denote the T-th words in the
9847 table f and f' ;
9848 X <- X ^ f[T]; Y <- Y ^ f'[T];
9849 end
```

9850 The pseudo-code is reproduced exactly as given; however, note that in the case of **cksum**, **A[i]**  
9851 represents a byte of the file, the words **X** and **Y** are treated as a single 32-bit value, and the tables  
9852 **f** and **f'** are a single table containing 32-bit values.

9853 The referenced Sarwate article also discusses generating the table.

#### 9854 FUTURE DIRECTIONS

9855 None.

#### 9856 SEE ALSO

9857 None.

#### 9858 CHANGE HISTORY

9859 First released in Issue 4.

**9860 NAME**

9861        cmp — compare two files

**9862 SYNOPSIS**

9863        cmp [ -l | -s ] file1 file2

**9864 DESCRIPTION**

9865        The *cmp* utility shall compare two files. The *cmp* utility shall write no output if the files are the  
9866        same. Under default options, if they differ, it shall write to standard output the byte and line  
9867        number at which the first difference occurred. Bytes and lines shall be numbered beginning with  
9868        1.

**9869 OPTIONS**

9870        The *cmp* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section  
9871        12.2, Utility Syntax Guidelines.

9872        The following options shall be supported:

9873        **-l**        (Lowercase ell.) Write the byte number (decimal) and the differing bytes (octal) for  
9874        each difference.

9875        **-s**        Write nothing for differing files; return exit status only.

**9876 OPERANDS**

9877        The following operands shall be supported:

9878        *file1*      A pathname of the first file to be compared. If *file1* is ‘-’, the standard input shall  
9879        be used.

9880        *file2*      A pathname of the second file to be compared. If *file2* is ‘-’, the standard input  
9881        shall be used.

9882        If both *file1* and *file2* refer to standard input or refer to the same FIFO special, block special, or  
9883        character special file, the results are undefined.

**9884 STDIN**

9885        The standard input shall be used only if the *file1* or *file2* operand refers to standard input. See the  
9886        INPUT FILES section.

**9887 INPUT FILES**

9888        The input files can be any file type.

**9889 ENVIRONMENT VARIABLES**

9890        The following environment variables shall affect the execution of *cmp*:

9891        **LANG**      Provide a default value for the internationalization variables that are unset or null.  
9892                  (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2,  
9893                  Internationalization Variables for the precedence of internationalization variables  
9894                  used to determine the values of locale categories.)

9895        **LC\_ALL**     If set to a non-empty string value, override the values of all the other  
9896        internationalization variables.

9897        **LC\_CTYPE**   Determine the locale for the interpretation of sequences of bytes of text data as  
9898                  characters (for example, single-byte as opposed to multi-byte characters in  
9899                  arguments).

**9900 *LC\_MESSAGES***

9901        Determine the locale that should be used to affect the format and contents of  
9902                  diagnostic messages written to standard error and informative messages written to  
9903                  standard output.

9904 XSI **NLSPATH** Determine the location of message catalogs for the processing of *LC\_MESSAGES*.

9905 **ASYNCHRONOUS EVENTS**

9906 Default.

9907 **STDOUT**

9908 In the POSIX locale, results of the comparison shall be written to standard output. When no  
9909 options are used, the format shall be:

9910 "%s %s differ: char %d, line %d\n", *file1*, *file2*,  
9911 <byte number>, <line number>

9912 When the **-l** option is used, the format shall be:

9913 "%d %o %o\n", <byte number>, <differing byte>,  
9914 <differing byte>

9915 for each byte that differs. The first <differing byte> number is from *file1* while the second is from  
9916 *file2*. In both cases, <byte number> shall be relative to the beginning of the file, beginning with 1.

9917 No output shall be written to standard output when the **-s** option is used.

9918 **STDERR**

9919 The standard error shall be used only for diagnostic messages. If *file1* and *file2* are identical for  
9920 the entire length of the shorter file, in the POSIX locale the following diagnostic message shall be  
9921 written, unless the **-s** option is specified:

9922 "cmp: EOF on %s%s\n", <name of shorter file>, <additional info>

9923 The <additional info> field shall either be null or a string that starts with a <blank> and contains  
9924 no <newline>s. Some implementations report on the number of lines in this case.

9925 **OUTPUT FILES**

9926 None.

9927 **EXTENDED DESCRIPTION**

9928 None.

9929 **EXIT STATUS**

9930 The following exit values shall be returned:

9931 0 The files are identical.

9932 1 The files are different; this includes the case where one file is identical to the first part of the  
9933 other.

9934 >1 An error occurred.

9935 **CONSEQUENCES OF ERRORS**

9936 Default.

9937 **APPLICATION USAGE**

9938 Although input files to *cmp* can be any type, the results might not be what would be expected on  
9939 character special device files or on file types not described by the System Interfaces volume of  
9940 IEEE Std 1003.1-2001. Since this volume of IEEE Std 1003.1-2001 does not specify the block size  
9941 used when doing input, comparisons of character special files need not compare all of the data  
9942 in those files.

9943 For files which are not text files, line numbers simply reflect the presence of a <newline>,  
9944 without any implication that the file is organized into lines.

9945 **EXAMPLES**

9946 None.

9947 **RATIONALE**

9948 The global language in Section 1.11 (on page 20) indicates that using two mutually-exclusive  
9949 options together produces unspecified results. Some System V implementations consider the  
9950 option usage:

9951 `cmp -l -s ...`

9952 to be an error. They also treat:

9953 `cmp -s -l ...`

9954 as if no options were specified. Both of these behaviors are considered bugs, but are allowed.

9955 The word **char** in the standard output format comes from historical usage, even though it is  
9956 actually a byte number. When *cmp* is supported in other locales, implementations are  
9957 encouraged to use the word *byte* or its equivalent in another language. Users should not  
9958 interpret this difference to indicate that the functionality of the utility changed between locales.

9959 Some implementations report on the number of lines in the identical-but-shorter file case. This is  
9960 allowed by the inclusion of the <additional info> fields in the output format. The restriction on  
9961 having a leading <blank> and no <newline>s is to make parsing for the filename easier. It is  
9962 recognized that some filenames containing white-space characters make parsing difficult  
9963 anyway, but the restriction does aid programs used on systems where the names are  
9964 predominantly well behaved.

9965 **FUTURE DIRECTIONS**

9966 None.

9967 **SEE ALSO**9968 *comm, diff*9969 **CHANGE HISTORY**

9970 First released in Issue 2.

**9971 NAME**

9972        comm — select or reject lines common to two files

**9973 SYNOPSIS**

9974        comm [-123] *file1 file2*

**9975 DESCRIPTION**

9976        The *comm* utility shall read *file1* and *file2*, which should be ordered in the current collating sequence, and produce three text columns as output: lines only in *file1*, lines only in *file2*, and lines in both files.

9979        If the lines in both files are not ordered according to the collating sequence of the current locale, the results are unspecified.

**9981 OPTIONS**

9982        The *comm* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section 12.2, Utility Syntax Guidelines.

9984        The following options shall be supported:

9985        **-1**        Suppress the output column of lines unique to *file1*.

9986        **-2**        Suppress the output column of lines unique to *file2*.

9987        **-3**        Suppress the output column of lines duplicated in *file1* and *file2*.

**9988 OPERANDS**

9989        The following operands shall be supported:

9990        *file1*      A pathname of the first file to be compared. If *file1* is ‘-’, the standard input shall be used.

9992        *file2*      A pathname of the second file to be compared. If *file2* is ‘-’, the standard input shall be used.

9994        If both *file1* and *file2* refer to standard input or to the same FIFO special, block special, or character special file, the results are undefined.

**9996 STDIN**

9997        The standard input shall be used only if one of the *file1* or *file2* operands refers to standard input.  
9998        See the INPUT FILES section.

**9999 INPUT FILES**

10000        The input files shall be text files.

**10001 ENVIRONMENT VARIABLES**

10002        The following environment variables shall affect the execution of *comm*:

10003        *LANG*      Provide a default value for the internationalization variables that are unset or null.  
10004                  (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2,  
10005                  Internationalization Variables for the precedence of internationalization variables  
10006                  used to determine the values of locale categories.)

10007        *LC\_ALL*     If set to a non-empty string value, override the values of all the other  
10008                  internationalization variables.

**10009 *LC\_COLLATE***

10010        Determine the locale for the collating sequence *comm* expects to have been used  
10011                  when the input files were sorted.

10012        *LC\_CTYPE*   Determine the locale for the interpretation of sequences of bytes of text data as  
10013                  characters (for example, single-byte as opposed to multi-byte characters in

10014 arguments and input files).

10015 **LC\_MESSAGES**

10016 Determine the locale that should be used to affect the format and contents of

10017 diagnostic messages written to standard error.

10018 XSI **NLSPATH** Determine the location of message catalogs for the processing of *LC\_MESSAGES*.

10019 **ASYNCHRONOUS EVENTS**

10020 Default.

10021 **STDOUT**

10022 The *comm* utility shall produce output depending on the options selected. If the **-1**, **-2**, and **-3** options are all selected, *comm* shall write nothing to standard output.

10023

10024 If the **-1** option is not selected, lines contained only in *file1* shall be written using the format:

10025 "%s\n", <line in file1>

10026 If the **-2** option is not selected, lines contained only in *file2* are written using the format:

10027 "%s%s\n", <lead>, <line in file2>

10028 where the string <lead> is as follows:

10029 <tab> The **-1** option is not selected.

10030 null string The **-1** option is selected.

10031 If the **-3** option is not selected, lines contained in both files shall be written using the format:

10032 "%s%s\n", <lead>, <line in both>

10033 where the string <lead> is as follows:

10034 <tab><tab> Neither the **-1** nor the **-2** option is selected.

10035 <tab> Exactly one of the **-1** and **-2** options is selected.

10036 null string Both the **-1** and **-2** options are selected.

10037 If the input files were ordered according to the collating sequence of the current locale, the lines

10038 written shall be in the collating sequence of the original lines.

10039 **STDERR**

10040 The standard error shall be used only for diagnostic messages.

10041 **OUTPUT FILES**

10042 None.

10043 **EXTENDED DESCRIPTION**

10044 None.

10045 **EXIT STATUS**

10046 The following exit values shall be returned:

10047 0 All input files were successfully output as specified.

10048 >0 An error occurred.

10049 **CONSEQUENCES OF ERRORS**

10050 Default.

**10051 APPLICATION USAGE**

10052 If the input files are not properly presorted, the output of *comm* might not be useful.

**10053 EXAMPLES**

10054 If a file named **xcu** contains a sorted list of the utilities in this volume of IEEE Std 1003.1-2001, a  
10055 file named **xpg3** contains a sorted list of the utilities specified in the X/Open Portability Guide,  
10056 Issue 3, and a file named **svid89** contains a sorted list of the utilities in the System V Interface  
10057 Definition Third Edition:

10058 `comm -23 xcu xpg3 | comm -23 - svid89`

10059 would print a list of utilities in this volume of IEEE Std 1003.1-2001 not specified by either of the  
10060 other documents:

10061 `comm -12 xcu xpg3 | comm -12 - svid89`

10062 would print a list of utilities specified by all three documents, and:

10063 `comm -12 xpg3 svid89 | comm -23 - xcu`

10064 would print a list of utilities specified by both XPG3 and the SVID, but not specified in this  
10065 volume of IEEE Std 1003.1-2001.

**10066 RATIONALE**

10067 None.

**10068 FUTURE DIRECTIONS**

10069 None.

**10070 SEE ALSO**

10071 *cmp, diff, sort, uniq*

**10072 CHANGE HISTORY**

10073 First released in Issue 2.

**10074 Issue 6**

10075 The normative text is reworded to avoid use of the term “must” for application requirements.

## 10076 NAME

10077 command — execute a simple command

## 10078 SYNOPSIS

10079 command [-p] *command\_name* [*argument ...*]10080 UP command [-v | -V] *command\_name*

10081

## 10082 DESCRIPTION

10083 The *command* utility shall cause the shell to treat the arguments as a simple command,  
10084 suppressing the shell function lookup that is described in Section 2.9.1.1 (on page 48), item 1b.

10085 If the *command\_name* is the same as the name of one of the special built-in utilities, the special  
10086 properties in the enumerated list at the beginning of Section 2.14 (on page 64) shall not occur. In  
10087 every other respect, if *command\_name* is not the name of a function, the effect of *command* (with  
10088 no options) shall be the same as omitting *command*.

10089 On systems supporting the User Portability Utilities option, the *command* utility also shall  
10090 provide information concerning how a command name is interpreted by the shell; see **-v** and  
10091 **-V**.

## 10092 OPTIONS

10093 The *command* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001,  
10094 Section 12.2, Utility Syntax Guidelines.

10095 The following options shall be supported:

10096 **-p** Perform the command search using a default value for *PATH* that is guaranteed to  
10097 find all of the standard utilities.

10098 **-v** (On systems supporting the User Portability Utilities option.) Write a string to  
10099 standard output that indicates the pathname or command that will be used by the  
10100 shell, in the current shell execution environment (see Section 2.12 (on page 61)), to  
10101 invoke *command\_name*, but do not invoke *command\_name*.

- 10102 • Utilities, regular built-in utilities, *command\_names* including a slash character,  
10103 and any implementation-defined functions that are found using the *PATH*  
10104 variable (as described in Section 2.9.1.1 (on page 48)), shall be written as  
10105 absolute pathnames.
- 10106 • Shell functions, special built-in utilities, regular built-in utilities not associated  
10107 with a *PATH* search, and shell reserved words shall be written as just their  
10108 names.
- 10109 • An alias shall be written as a command line that represents its alias definition.
- 10110 • Otherwise, no output shall be written and the exit status shall reflect that the  
10111 name was not found.

10112 **-V** (On systems supporting the User Portability Utilities option.) Write a string to  
10113 standard output that indicates how the name given in the *command\_name* operand  
10114 will be interpreted by the shell, in the current shell execution environment (see  
10115 Section 2.12 (on page 61)), but do not invoke *command\_name*. Although the format  
10116 of this string is unspecified, it shall indicate in which of the following categories  
10117 *command\_name* falls and shall include the information stated:

- 10118 • Utilities, regular built-in utilities, and any implementation-defined functions  
10119 that are found using the *PATH* variable (as described in Section 2.9.1.1 (on page  
10120 48)), shall be identified as such and include the absolute pathname in the string.

- 10121           • Other shell functions shall be identified as functions.  
10122           • Aliases shall be identified as aliases and their definitions included in the string.  
10123           • Special built-in utilities shall be identified as special built-in utilities.  
10124           • Regular built-in utilities not associated with a *PATH* search shall be identified  
10125            as regular built-in utilities. (The term “regular” need not be used.)  
10126           • Shell reserved words shall be identified as reserved words.

## 10127 OPERANDS

- 10128       The following operands shall be supported:  
10129        *argument*   One of the strings treated as an argument to *command\_name*.  
10130        *command\_name*  
10131            The name of a utility or a special built-in utility.

## 10132 STDIN

- 10133       Not used.

## 10134 INPUT FILES

- 10135       None.

## 10136 ENVIRONMENT VARIABLES

- 10137       The following environment variables shall affect the execution of *command*:

- 10138       *LANG*      Provide a default value for the internationalization variables that are unset or null.  
10139           (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2,  
10140           Internationalization Variables for the precedence of internationalization variables  
10141           used to determine the values of locale categories.)  
10142       *LC\_ALL*     If set to a non-empty string value, override the values of all the other  
10143           internationalization variables.  
10144       *LC\_CTYPE*   Determine the locale for the interpretation of sequences of bytes of text data as  
10145           characters (for example, single-byte as opposed to multi-byte characters in  
10146           arguments).  
10147       *LC\_MESSAGES*  
10148           Determine the locale that should be used to affect the format and contents of  
10149           diagnostic messages written to standard error and informative messages written to  
10150           standard output.  
10151 XSI      *NLSPATH*   Determine the location of message catalogs for the processing of *LC\_MESSAGES*.  
10152       *PATH*       Determine the search path used during the command search described in Section  
10153           2.9.1.1 (on page 48), except as described under the **-p** option.

## 10154 ASYNCHRONOUS EVENTS

- 10155       Default.

## 10156 STDOUT

- 10157       When the **-v** option is specified, standard output shall be formatted as:

10158       "%s\n", <pathname or command>

- 10159       When the **-V** option is specified, standard output shall be formatted as:

10160       "%s\n", <unspecified>

**10161 STDERR**

10162 The standard error shall be used only for diagnostic messages.

**10163 OUTPUT FILES**

10164 None.

**10165 EXTENDED DESCRIPTION**

10166 None.

**10167 EXIT STATUS**

10168 When the **-v** or **-V** options are specified, the following exit values shall be returned:

10169 0 Successful completion.

10170 >0 The *command\_name* could not be found or an error occurred.

10171 Otherwise, the following exit values shall be returned:

10172 126 The utility specified by *command\_name* was found but could not be invoked.

10173 127 An error occurred in the *command* utility or the utility specified by *command\_name* could not be found.

10175 Otherwise, the exit status of *command* shall be that of the simple command specified by the arguments to *command*.

**10177 CONSEQUENCES OF ERRORS**

10178 Default.

**10179 APPLICATION USAGE**

10180 The order for command search allows functions to override regular built-ins and path searches.  
10181 This utility is necessary to allow functions that have the same name as a utility to call the utility  
10182 (instead of a recursive call to the function).

10183 The system default path is available using *getconf*, however, since *getconf* may need to have the  
10184 *PATH* set up before it can be called itself, the following can be used:

10185 `command -p getconf _CS_PATH`

10186 There are some advantages to suppressing the special characteristics of special built-ins on  
10187 occasion. For example:

10188 `command exec > unwritable-file`

10189 does not cause a non-interactive script to abort, so that the output status can be checked by the  
10190 script.

10191 The *command*, *env*, *nohup*, *time*, and *xargs* utilities have been specified to use exit code 127 if an  
10192 error occurs so that applications can distinguish “failure to find a utility” from “invoked utility  
10193 exited with an error indication”. The value 127 was chosen because it is not commonly used for  
10194 other meanings; most utilities use small values for “normal error conditions” and the values  
10195 above 128 can be confused with termination due to receipt of a signal. The value 126 was chosen  
10196 in a similar manner to indicate that the utility could be found, but not invoked. Some scripts  
10197 produce meaningful error messages differentiating the 126 and 127 cases. The distinction  
10198 between exit codes 126 and 127 is based on KornShell practice that uses 127 when all attempts to  
10199 *exec* the utility fail with [ENOENT], and uses 126 when any attempt to *exec* the utility fails for  
10200 any other reason.

10201 Since the **-v** and **-V** options of *command* produce output in relation to the current shell execution  
10202 environment, *command* is generally provided as a shell regular built-in. If it is called in a subshell  
10203 or separate utility execution environment, such as one of the following:

10204 (PATH=foo command -v)  
 10205 nohup command -v

10206 it does not necessarily produce correct results. For example, when called with *nohup* or an *exec*  
 10207 function, in a separate utility execution environment, most implementations are not able to  
 10208 identify aliases, functions, or special built-ins.

10209 Two types of regular built-ins could be encountered on a system and these are described  
 10210 separately by *command*. The description of command search in Section 2.9.1.1 (on page 48)  
 10211 allows for a standard utility to be implemented as a regular built-in as long as it is found in the  
 10212 appropriate place in a *PATH* search. So, for example, *command -v true* might yield */bin/true* or  
 10213 some similar pathname. Other implementation-defined utilities that are not defined by this  
 10214 volume of IEEE Std 1003.1-2001 might exist only as built-ins and have no pathname associated  
 10215 with them. These produce output identified as (regular) built-ins. Applications encountering  
 10216 these are not able to count on execing them, using them with *nohup*, overriding them with a  
 10217 different *PATH*, and so on.

## 10218 EXAMPLES

1. Make a version of *cd* that always prints out the new working directory exactly once:

10220 cd() {  
 10221 command cd "\$@" >/dev/null  
 10222 pwd  
 10223 }

2. Start off a “secure shell script” in which the script avoids being spoofed by its parent:

10225 IFS='  
 10226 '  
 10227 # The preceding value should be <space><tab><newline>.  
 10228 # Set IFS to its default value.  
 10229 \unalias -a  
 10230 # Unset all possible aliases.  
 10231 # Note that unalias is escaped to prevent an alias  
 10232 # being used for unalias.  
 10233 unset -f command  
 10234 # Ensure command is not a user function.  
 10235 PATH=\$(command -p getconf \_CS\_PATH):\$PATH  
 10236 # Put on a reliable PATH prefix.  
 10237 # ...

10238 At this point, given correct permissions on the directories called by *PATH*, the script has  
 10239 the ability to ensure that any utility it calls is the intended one. It is being very cautious  
 10240 because it assumes that implementation extensions may be present that would allow user  
 10241 functions to exist when it is invoked; this capability is not specified by this volume of  
 10242 IEEE Std 1003.1-2001, but it is not prohibited as an extension. For example, the *ENV*  
 10243 variable precedes the invocation of the script with a user start-up script. Such a script  
 10244 could define functions to spoof the application.

## 10245 RATIONALE

10246 Since *command* is a regular built-in utility it is always found prior to the *PATH* search.

10247 There is nothing in the description of *command* that implies the command line is parsed any  
 10248 differently from that of any other simple command. For example:

10249 command a | b ; c

10250 is not parsed in any special way that causes ‘|’ or ‘;’ to be treated other than a pipe operator  
10251 or semicolon or that prevents function lookup on **b** or **c**.

10252 The *command* utility is somewhat similar to the Eighth Edition shell *builtin* command, but since  
10253 *command* also goes to the file system to search for utilities, the name *builtin* would not be  
10254 intuitive.

10255 The *command* utility is most likely to be provided as a regular built-in. It is not listed as a special  
10256 built-in for the following reasons:

- The removal of exportable functions made the special precedence of a special built-in unnecessary.

- A special built-in has special properties (see Section 2.14 (on page 64)) that were inappropriate for invoking other utilities. For example, two commands such as:

10261 date > *unwritable-file*

10262 command date > *unwritable-file*

10263 would have entirely different results; in a non-interactive script, the former would continue  
10264 to execute the next command, the latter would abort. Introducing this semantic difference  
10265 along with suppressing functions was seen to be non-intuitive.

10266 The –p option is present because it is useful to be able to ensure a safe path search that finds all  
10267 the standard utilities. This search might not be identical to the one that occurs through one of the  
10268 *exec* functions (as defined in the System Interfaces volume of IEEE Std 1003.1-2001) when *PATH*  
10269 is unset. At the very least, this feature is required to allow the script to access the correct version  
10270 of *getconf* so that the value of the default path can be accurately retrieved.

10271 The *command* –v and –V options were added to satisfy requirements from users that are  
10272 currently accomplished by three different historical utilities: *type* in the System V shell, *whence* in  
10273 the KornShell, and *which* in the C shell. Since there is no historical agreement on how and what  
10274 to accomplish here, the POSIX *command* utility was enhanced and the historical utilities were left  
10275 unmodified. The C shell *which* merely conducts a path search. The KornShell *whence* is more  
10276 elaborate—in addition to the categories required by POSIX, it also reports on tracked aliases,  
10277 exported aliases, and undefined functions.

10278 The output format of –V was left mostly unspecified because human users are its only audience.  
10279 Applications should not be written to care about this information; they can use the output of –v  
10280 to differentiate between various types of commands, but the additional information that may be  
10281 emitted by the more verbose –V is not needed and should not be arbitrarily constrained in its  
10282 verbosity or localization for application parsing reasons.

## 10283 FUTURE DIRECTIONS

10284 None.

## 10285 SEE ALSO

10286 Section 2.9.1.1 (on page 48), Section 2.12 (on page 61), Section 2.14 (on page 64), *sh*, *type*, the  
10287 System Interfaces volume of IEEE Std 1003.1-2001, *exec*

## 10288 CHANGE HISTORY

10289 First released in Issue 4.

## 10290 NAME

10291 compress — compress data

## 10292 SYNOPSIS

10293 XSI compress [-fv][-b bits][file ...]

10294 compress [-cfv][-b bits][file]

10295

## 10296 DESCRIPTION

10297 The *compress* utility shall attempt to reduce the size of the named files by using adaptive  
10298 Lempel-Ziv coding algorithm.

10299 Note: Lempel-Ziv is US Patent 4464650, issued to William Eastman, Abraham Lempel, Jacob Ziv,  
10300 Martin Cohn on August 7th, 1984, and assigned to Sperry Corporation.

10301 Lempel-Ziv-Welch compression is covered by US Patent 4558302, issued to Terry A. Welch on  
10302 December 10th, 1985, and assigned to Sperry Corporation.

10303 On systems not supporting adaptive Lempel-Ziv coding algorithm, the input files shall not be  
10304 changed and an error value greater than two shall be returned. Except when the output is to the  
10305 standard output, each file shall be replaced by one with the extension .Z. If the invoking process  
10306 has appropriate privileges, the ownership, modes, access time, and modification time of the  
10307 original file are preserved. If appending the .Z to the filename would make the name exceed  
10308 {NAME\_MAX} bytes, the command shall fail. If no files are specified, the standard input shall be  
10309 compressed to the standard output.

## 10310 OPTIONS

10311 The *compress* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001,  
10312 Section 12.2, Utility Syntax Guidelines.

10313 The following options shall be supported:

10314 **-b bits** Specify the maximum number of bits to use in a code. For a conforming  
10315 application, the *bits* argument shall be:

10316 9 <= *bits* <= 14

10317 The implementation may allow *bits* values of greater than 14. The default is 14, 15,  
10318 or 16.

10319 **-c** Cause *compress* to write to the standard output; the input file is not changed, and  
10320 no .Z files are created.

10321 **-f** Force compression of *file*, even if it does not actually reduce the size of the file, or if  
10322 the corresponding *file.Z* file already exists. If the **-f** option is not given, and the  
10323 process is not running in the background, the user is prompted as to whether an  
10324 existing *file.Z* file should be overwritten.

10325 **-v** Write the percentage reduction of each file to standard error.

## 10326 OPERANDS

10327 The following operand shall be supported:

10328 *file* A pathname of a file to be compressed.

## 10329 STDIN

10330 The standard input shall be used only if no *file* operands are specified, or if a *file* operand is '-'.

**10331 INPUT FILES**

10332 If *file* operands are specified, the input files contain the data to be compressed.

**10333 ENVIRONMENT VARIABLES**

10334 The following environment variables shall affect the execution of *compress*:

10335 *LANG* Provide a default value for the internationalization variables that are unset or null.  
(See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2, Internationalization Variables for the precedence of internationalization variables used to determine the values of locale categories.)

10339 *LC\_ALL* If set to a non-empty string value, override the values of all the other  
10340 internationalization variables.

10341 *LC\_CTYPE* Determine the locale for the interpretation of sequences of bytes of text data as  
10342 characters (for example, single-byte as opposed to multi-byte characters in  
10343 arguments).

**10344 *LC\_MESSAGES***

10345 Determine the locale that should be used to affect the format and contents of  
10346 diagnostic messages written to standard error.

10347 *NLSPATH* Determine the location of message catalogs for the processing of *LC\_MESSAGES*.

**10348 ASYNCHRONOUS EVENTS**

10349 Default.

**10350 STDOUT**

10351 If no *file* operands are specified, or if a *file* operand is ‘–’, or if the *-c* option is specified, the  
10352 standard output contains the compressed output.

**10353 STDERR**

10354 The standard error shall be used only for diagnostic and prompt messages and the output from  
10355 **–v**.

**10356 OUTPUT FILES**

10357 The output files shall contain the compressed output. The format of compressed files is  
10358 unspecified and interchange of such files between implementations (including access via  
10359 unspecified file sharing mechanisms) is not required by IEEE Std 1003.1-2001.

**10360 EXTENDED DESCRIPTION**

10361 None.

**10362 EXIT STATUS**

10363 The following exit values shall be returned:

10364 0 Successful completion.

10365 1 An error occurred.

10366 2 One or more files were not compressed because they would have increased in size (and the  
10367 *-f* option was not specified).

10368 >2 An error occurred.

**10369 CONSEQUENCES OF ERRORS**

10370 The input file shall remain unmodified.

10371 **APPLICATION USAGE**

10372     The amount of compression obtained depends on the size of the input, the number of *bits* per  
10373     code, and the distribution of common substrings. Typically, text such as source code or English  
10374     is reduced by 50-60%. Compression is generally much better than that achieved by Huffman  
10375     coding or adaptive Huffman coding (*compact*), and takes less time to compute.

10376     Although *compress* strictly follows the default actions upon receipt of a signal or when an error  
10377     occurs, some unexpected results may occur. In some implementations it is likely that a partially  
10378     compressed file is left in place, alongside its uncompressed input file. Since the general  
10379     operation of *compress* is to delete the uncompressed file only after the .Z file has been  
10380     successfully filled, an application should always carefully check the exit status of *compress* before  
10381     arbitrarily deleting files that have like-named neighbors with .Z suffixes.

10382     The limit of 14 on the *bits* option-argument is to achieve portability to all systems (within the  
10383     restrictions imposed by the lack of an explicit published file format). Some implementations  
10384     based on 16-bit architectures cannot support 15 or 16-bit uncompression.

10385 **EXAMPLES**

10386         None.

10387 **RATIONALE**

10388         None.

10389 **FUTURE DIRECTIONS**

10390         None.

10391 **SEE ALSO**

10392         *uncompress*, *zcat*

10393 **CHANGE HISTORY**

10394         First released in Issue 4.

10395 **Issue 6**

10396         The normative text is reworded to avoid use of the term “must” for application requirements.

10397         An error case is added for systems not supporting adaptive Lempel-Ziv coding.

## 10398 NAME

10399 cp — copy files

## 10400 SYNOPSIS

```
10401 cp [-fip] source_file target_file
10402 cp [-fip] source_file ... target
10403 cp -R [-H | -L | -P][-fip] source_file ... target
10404 OB cp -r [-H | -L | -P][-fip] source_file ... target
```

## 10405 DESCRIPTION

10406 The first synopsis form is denoted by two operands, neither of which are existing files of type  
10407 directory. The *cp* utility shall copy the contents of *source\_file* (or, if *source\_file* is a file of type  
10408 symbolic link, the contents of the file referenced by *source\_file*) to the destination path named by  
10409 *target\_file*.

10410 The second synopsis form is denoted by two or more operands where the **-R** or **-r** options are  
10411 not specified and the first synopsis form is not applicable. It shall be an error if any *source\_file* is a  
10412 file of type directory, if *target* does not exist, or if *target* is a file of a type defined by the System  
10413 Interfaces volume of IEEE Std 1003.1-2001, but is not a file of type directory. The *cp* utility shall  
10414 copy the contents of each *source\_file* (or, if *source\_file* is a file of type symbolic link, the contents  
10415 of the file referenced by *source\_file*) to the destination path named by the concatenation of *target*,  
10416 a slash character, and the last component of *source\_file*.

10417 The third and fourth synopsis forms are denoted by two or more operands where the **-R** or **-r**  
10418 options are specified. The *cp* utility shall copy each file in the file hierarchy rooted in each  
10419 *source\_file* to a destination path named as follows:

- 10420 • If *target* exists and is a file of type directory, the name of the corresponding destination path  
10421 for each file in the file hierarchy shall be the concatenation of *target*, a slash character, and the  
10422 pathname of the file relative to the directory containing *source\_file*.
- 10423 • If *target* does not exist and two operands are specified, the name of the corresponding destination path  
10424 for *source\_file* shall be *target*; the name of the corresponding destination path  
10425 for all other files in the file hierarchy shall be the concatenation of *target*, a slash character,  
10426 and the pathname of the file relative to *source\_file*.

10427 It shall be an error if *target* does not exist and more than two operands are specified, or if *target*  
10428 exists and is a file of a type defined by the System Interfaces volume of IEEE Std 1003.1-2001, but  
10429 is not a file of type directory.

10430 In the following description, the term *dest\_file* refers to the file named by the destination path.  
10431 The term *source\_file* refers to the file that is being copied, whether specified as an operand or a  
10432 file in a file hierarchy rooted in a *source\_file* operand. If *source\_file* is a file of type symbolic link:

- 10433 • If neither the **-R** nor **-r** options were specified, *cp* shall take actions based on the type and  
10434 contents of the file referenced by the symbolic link, and not by the symbolic link itself.
- 10435 • If the **-R** option was specified:
  - 10436 — If none of the options **-H**, **-L**, nor **-P** were specified, it is unspecified which of **-H**, **-L**, or  
10437 **-P** will be used as a default.
  - 10438 — If the **-H** option was specified, *cp* shall take actions based on the type and contents of the  
10439 file referenced by any symbolic link specified as a *source\_file* operand.
  - 10440 — If the **-L** option was specified, *cp* shall take actions based on the type and contents of the  
10441 file referenced by any symbolic link specified as a *source\_file* operand or any symbolic

- links encountered during traversal of a file hierarchy.
- If the **-P** option was specified, *cp* shall copy any symbolic link specified as a *source\_file* operand and any symbolic links encountered during traversal of a file hierarchy, and shall not follow any symbolic links.
  - If the **-r** option was specified, the behavior is implementation-defined.
- For each *source\_file*, the following steps shall be taken:
1. If *source\_file* references the same file as *dest\_file*, *cp* may write a diagnostic message to standard error; it shall do nothing more with *source\_file* and shall go on to any remaining files.
  2. If *source\_file* is of type directory, the following steps shall be taken:
    - a. If neither the **-R** or **-r** options were specified, *cp* shall write a diagnostic message to standard error, do nothing more with *source\_file*, and go on to any remaining files.
    - b. If *source\_file* was not specified as an operand and *source\_file* is dot or dot-dot, *cp* shall do nothing more with *source\_file* and go on to any remaining files.
    - c. If *dest\_file* exists and it is a file type not specified by the System Interfaces volume of IEEE Std 1003.1-2001, the behavior is implementation-defined.
    - d. If *dest\_file* exists and it is not of type directory, *cp* shall write a diagnostic message to standard error, do nothing more with *source\_file* or any files below *source\_file* in the file hierarchy, and go on to any remaining files.
    - e. If the directory *dest\_file* does not exist, it shall be created with file permission bits set to the same value as those of *source\_file*, modified by the file creation mask of the user if the **-p** option was not specified, and then bitwise-inclusively OR'ed with S\_IRWXU. If *dest\_file* cannot be created, *cp* shall write a diagnostic message to standard error, do nothing more with *source\_file*, and go on to any remaining files. It is unspecified if *cp* attempts to copy files in the file hierarchy rooted in *source\_file*.
    - f. The files in the directory *source\_file* shall be copied to the directory *dest\_file*, taking the four steps (1 to 4) listed here with the files as *source\_files*.
    - g. If *dest\_file* was created, its file permission bits shall be changed (if necessary) to be the same as those of *source\_file*, modified by the file creation mask of the user if the **-p** option was not specified.
    - h. The *cp* utility shall do nothing more with *source\_file* and go on to any remaining files.
  3. If *source\_file* is of type regular file, the following steps shall be taken:
    - a. If *dest\_file* exists, the following steps shall be taken:
      - i. If the **-i** option is in effect, the *cp* utility shall write a prompt to the standard error and read a line from the standard input. If the response is not affirmative, *cp* shall do nothing more with *source\_file* and go on to any remaining files.
      - ii. A file descriptor for *dest\_file* shall be obtained by performing actions equivalent to the *open()* function defined in the System Interfaces volume of IEEE Std 1003.1-2001 called using *dest\_file* as the *path* argument, and the bitwise-inclusive OR of O\_WRONLY and O\_TRUNC as the *oflag* argument.
      - iii. If the attempt to obtain a file descriptor fails and the **-f** option is in effect, *cp* shall attempt to remove the file by performing actions equivalent to the *unlink()* function defined in the System Interfaces volume of IEEE Std 1003.1-2001 called using *dest\_file* as the *path* argument.

- 10485 IEEE Std 1003.1-2001 called using *dest\_file* as the *path* argument. If this attempt  
10486 succeeds, *cp* shall continue with step 3b.
- 10487 b. If *dest\_file* does not exist, a file descriptor shall be obtained by performing actions  
10488 equivalent to the *open()* function defined in the System Interfaces volume of  
10489 IEEE Std 1003.1-2001 called using *dest\_file* as the *path* argument, and the bitwise-  
10490 inclusive OR of O\_WRONLY and O\_CREAT as the *oflag* argument. The file  
10491 permission bits of *source\_file* shall be the *mode* argument.
- 10492 c. If the attempt to obtain a file descriptor fails, *cp* shall write a diagnostic message to  
10493 standard error, do nothing more with *source\_file*, and go on to any remaining files.
- 10494 d. The contents of *source\_file* shall be written to the file descriptor. Any write errors  
10495 shall cause *cp* to write a diagnostic message to standard error and continue to step 3e.
- 10496 e. The file descriptor shall be closed.
- 10497 f. The *cp* utility shall do nothing more with *source\_file*. If a write error occurred in step  
10498 3d, it is unspecified if *cp* continues with any remaining files. If no write error  
10499 occurred in step 3d, *cp* shall go on to any remaining files.
- 10500 4. Otherwise, the following steps shall be taken:
- 10501 a. If the **-r** option was specified, the behavior is implementation-defined.
- 10502 b. If the **-R** option was specified, the following steps shall be taken:
- 10503 i. The *dest\_file* shall be created with the same file type as *source\_file*.
- 10504 ii. If *source\_file* is a file of type FIFO, the file permission bits shall be the same as  
10505 those of *source\_file*, modified by the file creation mask of the user if the **-p**  
10506 option was not specified. Otherwise, the permissions, owner ID, and group ID  
10507 of *dest\_file* are implementation-defined.
- 10508 If this creation fails for any reason, *cp* shall write a diagnostic message to  
10509 standard error, do nothing more with *source\_file*, and go on to any remaining  
10510 files.
- 10511 iii. If *source\_file* is a file of type symbolic link, the pathname contained in *dest\_file*  
10512 shall be the same as the pathname contained in *source\_file*.
- 10513 If this fails for any reason, *cp* shall write a diagnostic message to standard error,  
10514 do nothing more with *source\_file*, and go on to any remaining files.

10515 If the implementation provides additional or alternate access control mechanisms (see the Base  
10516 Definitions volume of IEEE Std 1003.1-2001, Section 4.4, File Access Permissions), their effect on  
10517 copies of files is implementation-defined.

## 10518 OPTIONS

10519 The *cp* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section 12.2,  
10520 Utility Syntax Guidelines.

10521 The following options shall be supported:

- 10522 **-f** If a file descriptor for a destination file cannot be obtained, as described in step  
10523 3.a.ii., attempt to unlink the destination file and proceed.
- 10524 **-H** Take actions based on the type and contents of the file referenced by any symbolic  
10525 link specified as a *source\_file* operand.
- 10526 **-i** Write a prompt to standard error before copying to any existing destination file. If  
10527 the response from the standard input is affirmative, the copy shall be attempted;

10528 otherwise, it shall not.

10529 **-L** Take actions based on the type and contents of the file referenced by any symbolic link specified as a *source\_file* operand or any symbolic links encountered during traversal of a file hierarchy.

10530

10531

10532 **-P** Take actions on any symbolic link specified as a *source\_file* operand or any symbolic link encountered during traversal of a file hierarchy.

10533

10534 **-p** Duplicate the following characteristics of each source file in the corresponding destination file:

10535

10536 1. The time of last data modification and time of last access. If this duplication fails for any reason, *cp* shall write a diagnostic message to standard error.

10537

10538 2. The user ID and group ID. If this duplication fails for any reason, it is unspecified whether *cp* writes a diagnostic message to standard error.

10539

10540 3. The file permission bits and the S\_ISUID and S\_ISGID bits. Other, implementation-defined, bits may be duplicated as well. If this duplication fails for any reason, *cp* shall write a diagnostic message to standard error.

10541

10542

10543 If the user ID or the group ID cannot be duplicated, the file permission bits S\_ISUID and S\_ISGID shall be cleared. If these bits are present in the source file but are not duplicated in the destination file, it is unspecified whether *cp* writes a diagnostic message to standard error.

10544

10545

10546

10547 The order in which the preceding characteristics are duplicated is unspecified. The *dest\_file* shall not be deleted if these characteristics cannot be preserved.

10548

10549 **-R** Copy file hierarchies.

10550 OB **-r** Copy file hierarchies. The treatment of special files is implementation-defined.

10551 Specifying more than one of the mutually-exclusive options **-H**, **-L**, and **-P** shall not be considered an error. The last option specified shall determine the behavior of the utility.

10552

## 10553 OPERANDS

10554 The following operands shall be supported:

10555 *source\_file* A pathname of a file to be copied.

10556 *target\_file* A pathname of an existing or nonexistent file, used for the output when a single file is copied.

10557

10558 *target* A pathname of a directory to contain the copied files.

## 10559 STDIN

10560 The standard input shall be used to read an input line in response to each prompt specified in the STDERR section. Otherwise, the standard input shall not be used.

10561

## 10562 INPUT FILES

10563 The input files specified as operands may be of any file type.

## 10564 ENVIRONMENT VARIABLES

10565 The following environment variables shall affect the execution of *cp*:

10566 *LANG* Provide a default value for the internationalization variables that are unset or null. (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2, Internationalization Variables for the precedence of internationalization variables used to determine the values of locale categories.)

10567

10568

10569

|           |                               |                                                                                                                                                                                                                                                                                                                                                          |
|-----------|-------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 10570     | <i>LC_ALL</i>                 | If set to a non-empty string value, override the values of all the other internationalization variables.                                                                                                                                                                                                                                                 |
| 10572     | <i>LC_COLLATE</i>             | Determine the locale for the behavior of ranges, equivalence classes, and multi-character collating elements used in the extended regular expression defined for the <b>yesexpr</b> locale keyword in the <i>LC_MESSAGES</i> category.                                                                                                                   |
| 10576     | <i>LC_CTYPE</i>               | Determine the locale for the interpretation of sequences of bytes of text data as characters (for example, single-byte as opposed to multi-byte characters in arguments and input files) and the behavior of character classes used in the extended regular expression defined for the <b>yesexpr</b> locale keyword in the <i>LC_MESSAGES</i> category. |
| 10581     | <i>LC_MESSAGES</i>            | Determine the locale for the processing of affirmative responses that should be used to affect the format and contents of diagnostic messages written to standard error.                                                                                                                                                                                 |
| 10585 XSI | <i>NLSPATH</i>                | Determine the location of message catalogs for the processing of <i>LC_MESSAGES</i> .                                                                                                                                                                                                                                                                    |
| 10586     | <b>ASYNCHRONOUS EVENTS</b>    |                                                                                                                                                                                                                                                                                                                                                          |
| 10587     |                               | Default.                                                                                                                                                                                                                                                                                                                                                 |
| 10588     | <b>STDOUT</b>                 |                                                                                                                                                                                                                                                                                                                                                          |
| 10589     |                               | Not used.                                                                                                                                                                                                                                                                                                                                                |
| 10590     | <b>STDERR</b>                 |                                                                                                                                                                                                                                                                                                                                                          |
| 10591     |                               | A prompt shall be written to standard error under the conditions specified in the DESCRIPTION section. The prompt shall contain the destination pathname, but its format is otherwise unspecified. Otherwise, the standard error shall be used only for diagnostic messages.                                                                             |
| 10594     | <b>OUTPUT FILES</b>           |                                                                                                                                                                                                                                                                                                                                                          |
| 10595     |                               | The output files may be of any type.                                                                                                                                                                                                                                                                                                                     |
| 10596     | <b>EXTENDED DESCRIPTION</b>   |                                                                                                                                                                                                                                                                                                                                                          |
| 10597     |                               | None.                                                                                                                                                                                                                                                                                                                                                    |
| 10598     | <b>EXIT STATUS</b>            |                                                                                                                                                                                                                                                                                                                                                          |
| 10599     |                               | The following exit values shall be returned:                                                                                                                                                                                                                                                                                                             |
| 10600     |                               | 0 All files were copied successfully.                                                                                                                                                                                                                                                                                                                    |
| 10601     |                               | >0 An error occurred.                                                                                                                                                                                                                                                                                                                                    |
| 10602     | <b>CONSEQUENCES OF ERRORS</b> |                                                                                                                                                                                                                                                                                                                                                          |
| 10603     |                               | If <i>cp</i> is prematurely terminated by a signal or error, files or file hierarchies may be only partially copied and files and directories may have incorrect permissions or access and modification times.                                                                                                                                           |

## 10606 APPLICATION USAGE

10607 The difference between **-R** and **-r** is in the treatment by *cp* of file types other than regular and  
10608 directory. The original **-r** flag, for historic reasons, does not handle special files any differently  
10609 from regular files, but always reads the file and copies its contents. This has obvious problems in  
10610 the presence of special file types; for example, character devices, FIFOs, and sockets. The **-R**  
10611 option is intended to recreate the file hierarchy and the **-r** option supports historical practice. It  
10612 was anticipated that a future version of this volume of IEEE Std 1003.1-2001 would deprecate the  
10613 **-r** option, and for that reason, there has been no attempt to fix its behavior with respect to FIFOs  
10614 or other file types where copying the file is clearly wrong. However, some implementations  
10615 support **-r** with the same abilities as the **-R** defined in this volume of IEEE Std 1003.1-2001. To  
10616 accommodate them as well as systems that do not, the differences between **-r** and **-R** are  
10617 implementation-defined. Implementations may make them identical. The **-r** option is marked  
10618 obsolescent.

10619 The set-user-ID and set-group-ID bits are explicitly cleared when files are created. This is to  
10620 prevent users from creating programs that are set-user-ID or set-group-ID to them when  
10621 copying files or to make set-user-ID or set-group-ID files accessible to new groups of users. For  
10622 example, if a file is set-user-ID and the copy has a different group ID than the source, a new  
10623 group of users has execute permission to a set-user-ID program than did previously. In  
10624 particular, this is a problem for superusers copying users' trees.

## 10625 EXAMPLES

10626 None.

## 10627 RATIONALE

10628 The **-i** option exists on BSD systems, giving applications and users a way to avoid accidentally  
10629 removing files when copying. Although the 4.3 BSD version does not prompt if the standard  
10630 input is not a terminal, the standard developers decided that use of **-i** is a request for interaction,  
10631 so when the destination path exists, the utility takes instructions from whatever responds on  
10632 standard input.

10633 The exact format of the interactive prompts is unspecified. Only the general nature of the  
10634 contents of prompts are specified because implementations may desire more descriptive  
10635 prompts than those used on historical implementations. Therefore, an application using the **-i**  
10636 option relies on the system to provide the most suitable dialog directly with the user, based on  
10637 the behavior specified.

10638 The **-p** option is historical practice on BSD systems, duplicating the time of last data  
10639 modification and time of last access. This volume of IEEE Std 1003.1-2001 extends it to preserve  
10640 the user and group IDs, as well as the file permissions. This requirement has obvious problems  
10641 in that the directories are almost certainly modified after being copied. This volume of  
10642 IEEE Std 1003.1-2001 requires that the modification times be preserved. The statement that the  
10643 order in which the characteristics are duplicated is unspecified is to permit implementations to  
10644 provide the maximum amount of security for the user. Implementations should take into  
10645 account the obvious security issues involved in setting the owner, group, and mode in the  
10646 wrong order or creating files with an owner, group, or mode different from the final value.

10647 It is unspecified whether *cp* writes diagnostic messages when the user and group IDs cannot be  
10648 set due to the widespread practice of users using **-p** to duplicate some portion of the file  
10649 characteristics, indifferent to the duplication of others. Historic implementations only write  
10650 diagnostic messages on errors other than [EPERM].

10651 The **-r** option is historical practice on BSD and BSD-derived systems, copying file hierarchies as  
10652 opposed to single files. This functionality is used heavily in historical applications, and its loss  
10653 would significantly decrease consensus. The **-R** option was added as a close synonym to the **-r**  
10654 option, selected for consistency with all other options in this volume of IEEE Std 1003.1-2001 that

- 10655 do recursive directory descent.
- 10656 When a failure occurs during the copying of a file hierarchy, *cp* is required to attempt to copy  
10657 files that are on the same level in the hierarchy or above the file where the failure occurred. It is  
10658 unspecified if *cp* shall attempt to copy files below the file where the failure occurred (which  
10659 cannot succeed in any case).
- 10660 Permissions, owners, and groups of created special file types have been deliberately left as  
10661 implementation-defined. This is to allow systems to satisfy special requirements (for example,  
10662 allowing users to create character special devices, but requiring them to be owned by a certain  
10663 group). In general, it is strongly suggested that the permissions, owner, and group be the same  
10664 as if the user had run the historical *mknod*, *In*, or other utility to create the file. It is also probable  
10665 that additional privileges are required to create block, character, or other implementation-  
10666 defined special file types.
- 10667 Additionally, the **-p** option explicitly requires that all set-user-ID and set-group-ID permissions  
10668 be discarded if any of the owner or group IDs cannot be set. This is to keep users from  
10669 unintentionally giving away special privilege when copying programs.
- 10670 When creating regular files, historical versions of *cp* use the mode of the source file as modified  
10671 by the file mode creation mask. Other choices would have been to use the mode of the source file  
10672 unmodified by the creation mask or to use the same mode as would be given to a new file  
10673 created by the user (plus the execution bits of the source file) and then modify it by the file mode  
10674 creation mask. In the absence of any strong reason to change historic practice, it was in large part  
10675 retained.
- 10676 When creating directories, historical versions of *cp* use the mode of the source directory, plus  
10677 read, write, and search bits for the owner, as modified by the file mode creation mask. This is  
10678 done so that *cp* can copy trees where the user has read permission, but the owner does not. A  
10679 side effect is that if the file creation mask denies the owner permissions, *cp* fails. Also, once the  
10680 copy is done, historical versions of *cp* set the permissions on the created directory to be the same  
10681 as the source directory, unmodified by the file creation mask.
- 10682 This behavior has been modified so that *cp* is always able to create the contents of the directory,  
10683 regardless of the file creation mask. After the copy is done, the permissions are set to be the same  
10684 as the source directory, as modified by the file creation mask. This latter change from historical  
10685 behavior is to prevent users from accidentally creating directories with permissions beyond  
10686 those they would normally set and for consistency with the behavior of *cp* in creating files.
- 10687 It is not a requirement that *cp* detect attempts to copy a file to itself; however, implementations  
10688 are strongly encouraged to do so. Historical implementations have detected the attempt in most  
10689 cases.
- 10690 There are two methods of copying subtrees in this volume of IEEE Std 1003.1-2001. The other  
10691 method is described as part of the *pax* utility (see *pax*). Both methods are historical practice. The  
10692 *cp* utility provides a simpler, more intuitive interface, while *pax* offers a finer granularity of  
10693 control. Each provides additional functionality to the other; in particular, *pax* maintains the  
10694 hard-link structure of the hierarchy, while *cp* does not. It is the intention of the standard  
10695 developers that the results be similar (using appropriate option combinations in both utilities).  
10696 The results are not required to be identical; there seemed insufficient gain to applications to  
10697 balance the difficulty of implementations having to guarantee that the results would be exactly  
10698 identical.
- 10699 The wording allowing *cp* to copy a directory to implementation-defined file types not specified  
10700 by the System Interfaces volume of IEEE Std 1003.1-2001 is provided so that implementations  
10701 supporting symbolic links are not required to prohibit copying directories to symbolic links.  
10702 Other extensions to the System Interfaces volume of IEEE Std 1003.1-2001 file types may need to

10703 use this loophole as well.

10704 **FUTURE DIRECTIONS**

10705 The **-r** option may be removed; use **-R** instead.

10706 **SEE ALSO**

10707 *mv*, *find*, *In*, *pax*, the System Interfaces volume of IEEE Std 1003.1-2001, *open()*, *unlink()*

1

10708 **CHANGE HISTORY**

10709 First released in Issue 2.

10710 **Issue 6**

10711 The **-r** option is marked obsolescent.

10712 The new options **-H**, **-L**, and **-P** are added to align with the IEEE P1003.2b draft standard. These  
10713 options affect the processing of symbolic links.

10714 IEEE PASC Interpretation 1003.2 #194 is applied, adding a description of the **-P** option.

10715 IEEE Std 1003.1-2001/Cor 1-2002, item XCU/TC1/D6/18 is applied, correcting an error in the  
10716 SEE ALSO section.

1  
1

**10717 NAME**

10718        crontab — schedule periodic background work

**10719 SYNOPSIS**

10720 UP        **crontab [file]**

10721        **crontab [ -e | -l | -r ]**

10722

**10723 DESCRIPTION**

10724        The *crontab* utility shall create, replace, or edit a user's crontab entry; a crontab entry is a list of  
10725        commands and the times at which they shall be executed. The new crontab entry can be input by  
10726        specifying *file* or input from standard input if no *file* operand is specified, or by using an editor, if  
10727        **-e** is specified.

10728        Upon execution of a command from a crontab entry, the implementation shall supply a default  
10729        environment, defining at least the following environment variables:

10730        **HOME**        A pathname of the user's home directory.

10731        **LOGNAME** The user's login name.

10732        **PATH**        A string representing a search path guaranteed to find all of the standard utilities.

10733        **SHELL**        A pathname of the command interpreter. When *crontab* is invoked as specified by  
10734        this volume of IEEE Std 1003.1-2001, the value shall be a pathname for *sh*.

10735        The values of these variables when *crontab* is invoked as specified by this volume of  
10736        IEEE Std 1003.1-2001 shall not affect the default values provided when the scheduled command  
10737        is run.

10738        If standard output and standard error are not redirected by commands executed from the  
10739        crontab entry, any generated output or errors shall be mailed, via an implementation-defined  
10740        method, to the user.

10741 XSI        Users shall be permitted to use *crontab* if their names appear in the file **/usr/lib/cron/cron.allow**.  
10742        If that file does not exist, the file **/usr/lib/cron/cron.deny** shall be checked to determine whether  
10743        the user shall be denied access to *crontab*. If neither file exists, only a process with appropriate  
10744        privileges shall be allowed to submit a job. If only **cron.deny** exists and is empty, global usage  
10745        shall be permitted. The **cron.allow** and **cron.deny** files shall consist of one user name per line.

**10746 OPTIONS**

10747        The *crontab* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section  
10748        12.2, Utility Syntax Guidelines.

10749        The following options shall be supported:

10750        **-e**        Edit a copy of the invoking user's crontab entry, or create an empty entry to edit if  
10751        the crontab entry does not exist. When editing is complete, the entry shall be  
10752        installed as the user's crontab entry.

10753        **-l**        (The letter ell.) List the invoking user's crontab entry.

10754        **-r**        Remove the invoking user's crontab entry.

**10755 OPERANDS**

10756        The following operand shall be supported:

10757        **file**        The pathname of a file that contains specifications, in the format defined in the  
10758        INPUT FILES section, for crontab entries.

10759 **STDIN**

10760 See the INPUT FILES section.

10761 **INPUT FILES**

10762 In the POSIX locale, the user or application shall ensure that a crontab entry is a text file  
10763 consisting of lines of six fields each. The fields shall be separated by <blank>s. The first five  
10764 fields shall be integer patterns that specify the following:

- 10765 1. Minute [0,59]
- 10766 2. Hour [0,23]
- 10767 3. Day of the month [1,31]
- 10768 4. Month of the year [1,12]
- 10769 5. Day of the week ([0,6] with 0=Sunday)

10770 Each of these patterns can be either an asterisk (meaning all valid values), an element, or a list of  
10771 elements separated by commas. An element shall be either a number or two numbers separated  
10772 by a hyphen (meaning an inclusive range). The specification of days can be made by two fields  
10773 (day of the month and day of the week). If month, day of month, and day of week are all  
10774 asterisks, every day shall be matched. If either the month or day of month is specified as an  
10775 element or list, but the day of week is an asterisk, the month and day of month fields shall  
10776 specify the days that match. If both month and day of month are specified as an asterisk, but day  
10777 of week is an element or list, then only the specified days of the week match. Finally, if either the  
10778 month or day of month is specified as an element or list, and the day of week is also specified as  
10779 an element or list, then any day matching either the month and day of month, or the day of  
10780 week, shall be matched.

10781 The sixth field of a line in a crontab entry is a string that shall be executed by *sh* at the specified  
10782 times. A percent sign character in this field shall be translated to a <newline>. Any character  
10783 preceded by a backslash (including the '%'') shall cause that character to be treated literally.  
10784 Only the first line (up to a '%' or end-of-line) of the command field shall be executed by the  
10785 command interpreter. The other lines shall be made available to the command as standard input.

10786 Blank lines and those whose first non-<blank> is '#' shall be ignored.

10787 XSI The text files */usr/lib/cron/cron.allow* and */usr/lib/cron/cron.deny* shall contain zero or more  
10788 user names, one per line, of users who are, respectively, authorized or denied access to the  
10789 service underlying the *crontab* utility.

10790 **ENVIRONMENT VARIABLES**

10791 The following environment variables shall affect the execution of *crontab*:

- 10792 **EDITOR** Determine the editor to be invoked when the **-e** option is specified. The default  
10793 editor shall be *vi*.
- 10794 **LANG** Provide a default value for the internationalization variables that are unset or null.  
10795 (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2,  
10796 Internationalization Variables for the precedence of internationalization variables  
10797 used to determine the values of locale categories.)
- 10798 **LC\_ALL** If set to a non-empty string value, override the values of all the other  
10799 internationalization variables.
- 10800 **LC\_CTYPE** Determine the locale for the interpretation of sequences of bytes of text data as  
10801 characters (for example, single-byte as opposed to multi-byte characters in  
10802 arguments and input files).

- 10803       ***LC\_MESSAGES***  
10804              Determine the locale that should be used to affect the format and contents of  
10805              diagnostic messages written to standard error.
- 10806 XSI       ***NLSPATH***     Determine the location of message catalogs for the processing of ***LC\_MESSAGES***.
- 10807 **ASYNCHRONOUS EVENTS**
- 10808              Default.
- 10809 **STDOUT**
- 10810              If the **-l** option is specified, the crontab entry shall be written to the standard output.
- 10811 **STDERR**
- 10812              The standard error shall be used only for diagnostic messages.
- 10813 **OUTPUT FILES**
- 10814              None.
- 10815 **EXTENDED DESCRIPTION**
- 10816              None.
- 10817 **EXIT STATUS**
- 10818              The following exit values shall be returned:
- 10819              0   Successful completion.
- 10820              >0   An error occurred.
- 10821 **CONSEQUENCES OF ERRORS**
- 10822              The user's crontab entry is not submitted, removed, edited, or listed.
- 10823 **APPLICATION USAGE**
- 10824              The format of the crontab entry shown here is guaranteed only for the POSIX locale. Other  
10825              cultures may be supported with substantially different interfaces, although implementations are  
10826              encouraged to provide comparable levels of functionality.
- 10827              The default settings of the *HOME*, *LOGNAME*, *PATH*, and *SHELL* variables that are given to the  
10828              scheduled job are not affected by the settings of those variables when *crontab* is run; as stated,  
10829              they are defaults. The text about "invoked as specified by this volume of IEEE Std 1003.1-2001"  
10830              means that the implementation may provide extensions that allow these variables to be affected  
10831              at runtime, but that the user has to take explicit action in order to access the extension, such as  
10832              give a new option flag or modify the format of the crontab entry.
- 10833              A typical user error is to type only *crontab*; this causes the system to wait for the new crontab  
10834              entry on standard input. If end-of-file is typed (generally <control>-D), the crontab entry is  
10835              replaced by an empty file. In this case, the user should type the interrupt character, which  
10836              prevents the crontab entry from being replaced.
- 10837 **EXAMPLES**
- 10838              1. Clean up **core** files every weekday morning at 3:15 am:  
10839                      15 3 \* \* 1-5 find \$HOME -name core 2>/dev/null | xargs rm -f
- 10840              2. Mail a birthday greeting:  
10841                      0 12 14 2 \* mailx john%Happy Birthday!%Time for lunch.
- 10842              3. As an example of specifying the two types of days:  
10843                      0 0 1,15 \* 1

10844 would run a command on the first and fifteenth of each month, as well as on every  
10845 Monday. To specify days by only one field, the other field should be set to '\*' ; for  
10846 example:

10847 0 0 \* \* 1

10848 would run a command only on Mondays.

10849 **RATIONALE**

10850 All references to a *cron* daemon and to *cron files* have been omitted. Although historical  
10851 implementations have used this arrangement, there is no reason to limit future implementations.

10852 This description of *crontab* is designed to support only users with normal privileges. The format  
10853 of the input is based on the System V *crontab*; however, there is no requirement here that the  
10854 actual system database used by the *cron* daemon (or a similar mechanism) use this format  
10855 internally. For example, systems derived from BSD are likely to have an additional field  
10856 appended that indicates the user identity to be used when the job is submitted.

10857 The **-e** option was adopted from the SVID as a user convenience, although it does not exist in all  
10858 historical implementations.

10859 **FUTURE DIRECTIONS**

10860 None.

10861 **SEE ALSO**

10862 *at*

10863 **CHANGE HISTORY**

10864 First released in Issue 2.

10865 **Issue 6**

10866 This utility is marked as part of the User Portability Utilities option.

10867 The normative text is reworded to avoid use of the term "must" for application requirements.

## 10868 NAME

10869 csplit — split files based on context

## 10870 SYNOPSIS

10871 UP csplit [-ks][-f prefix][-n number] file arg1 ...argn

10872

## 10873 DESCRIPTION

10874 The *csplit* utility shall read the file named by the *file* operand, write all or part of that file into  
10875 other files as directed by the *arg* operands, and write the sizes of the files.

## 10876 OPTIONS

10877 The *csplit* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section  
10878 12.2, Utility Syntax Guidelines.

10879 The following options shall be supported:

- 10880 **-f prefix** Name the created files *prefix00*, *prefix01*, ..., *prefixn*. The default is **xx00** ... **xxn**. If  
10881 the *prefix* argument would create a filename exceeding {NAME\_MAX} bytes, an  
10882 error shall result, *csplit* shall exit with a diagnostic message, and no files shall be  
10883 created.
- 10884 **-k** Leave previously created files intact. By default, *csplit* shall remove created files if  
10885 an error occurs.
- 10886 **-n number** Use *number* decimal digits to form filenames for the file pieces. The default shall be  
10887 2.
- 10888 **-s** Suppress the output of file size messages.

## 10889 OPERANDS

10890 The following operands shall be supported:

- 10891 *file* The pathname of a text file to be split. If *file* is '**-**', the standard input shall be  
10892 used.

10893 The operands *arg1* ... *argn* can be a combination of the following:

10894 **/rexp/[offset]**

10895 A file shall be created using the content of the lines from the current line up to, but  
10896 not including, the line that results from the evaluation of the regular expression  
10897 with *offset*, if any, applied. The regular expression *rexp* shall follow the rules for  
10898 basic regular expressions described in the Base Definitions volume of  
10899 IEEE Std 1003.1-2001, Section 9.3, Basic Regular Expressions. The application shall  
10900 use the sequence "**\/**" to specify a slash character within the *rexp*. The optional  
10901 offset shall be a positive or negative integer value representing a number of lines.  
10902 A positive integer value can be preceded by '+'. If the selection of lines from an  
10903 *offset* expression of this type would create a file with zero lines, or one with greater  
10904 than the number of lines left in the input file, the results are unspecified. After the  
10905 section is created, the current line shall be set to the line that results from the  
10906 evaluation of the regular expression with any offset applied. If the current line is  
10907 the first line in the file and a regular expression operation has not yet been  
10908 performed, the pattern match of *rexp* shall be applied from the current line to the  
10909 end of the file. Otherwise, the pattern match of *rexp* shall be applied from the line  
10910 following the current line to the end of the file.

10911 **%rexp%[offset]**

10912 Equivalent to **/rexp/[offset]**, except that no file shall be created for the selected  
10913 section of the input file. The application shall use the sequence "**\%**" to specify a

|       |                              |                                                                                                                                                                                                                                                                                                                 |
|-------|------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 10914 |                              | percent-sign character within the <i>rexp</i> .                                                                                                                                                                                                                                                                 |
| 10915 | <i>line_no</i>               | Create a file from the current line up to (but not including) the line number <i>line_no</i> . Lines in the file shall be numbered starting at one. The current line becomes <i>line_no</i> .                                                                                                                   |
| 10916 |                              |                                                                                                                                                                                                                                                                                                                 |
| 10917 |                              |                                                                                                                                                                                                                                                                                                                 |
| 10918 | { <i>num</i> }               | Repeat operand. This operand can follow any of the operands described previously. If it follows a <i>rexp</i> type operand, that operand shall be applied <i>num</i> more times. If it follows a <i>line_no</i> operand, the file shall be split every <i>line_no</i> lines, <i>num</i> times, from that point. |
| 10919 |                              |                                                                                                                                                                                                                                                                                                                 |
| 10920 |                              |                                                                                                                                                                                                                                                                                                                 |
| 10921 |                              |                                                                                                                                                                                                                                                                                                                 |
| 10922 |                              | An error shall be reported if an operand does not reference a line between the current position and the end of the file.                                                                                                                                                                                        |
| 10923 |                              |                                                                                                                                                                                                                                                                                                                 |
| 10924 | <b>STDIN</b>                 |                                                                                                                                                                                                                                                                                                                 |
| 10925 |                              | See the INPUT FILES section.                                                                                                                                                                                                                                                                                    |
| 10926 | <b>INPUT FILES</b>           |                                                                                                                                                                                                                                                                                                                 |
| 10927 |                              | The input file shall be a text file.                                                                                                                                                                                                                                                                            |
| 10928 | <b>ENVIRONMENT VARIABLES</b> |                                                                                                                                                                                                                                                                                                                 |
| 10929 |                              | The following environment variables shall affect the execution of <i>csplit</i> :                                                                                                                                                                                                                               |
| 10930 | <i>LANG</i>                  | Provide a default value for the internationalization variables that are unset or null. (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2, Internationalization Variables for the precedence of internationalization variables used to determine the values of locale categories.)           |
| 10931 |                              |                                                                                                                                                                                                                                                                                                                 |
| 10932 |                              |                                                                                                                                                                                                                                                                                                                 |
| 10933 |                              |                                                                                                                                                                                                                                                                                                                 |
| 10934 | <i>LC_ALL</i>                | If set to a non-empty string value, override the values of all the other internationalization variables.                                                                                                                                                                                                        |
| 10935 |                              |                                                                                                                                                                                                                                                                                                                 |
| 10936 | <i>LC_COLLATE</i>            | Determine the locale for the behavior of ranges, equivalence classes, and multi-character collating elements within regular expressions.                                                                                                                                                                        |
| 10937 |                              |                                                                                                                                                                                                                                                                                                                 |
| 10938 |                              |                                                                                                                                                                                                                                                                                                                 |
| 10939 | <i>LC_CTYPE</i>              | Determine the locale for the interpretation of sequences of bytes of text data as characters (for example, single-byte as opposed to multi-byte characters in arguments and input files) and the behavior of character classes within regular expressions.                                                      |
| 10940 |                              |                                                                                                                                                                                                                                                                                                                 |
| 10941 |                              |                                                                                                                                                                                                                                                                                                                 |
| 10942 |                              |                                                                                                                                                                                                                                                                                                                 |
| 10943 | <i>LC_MESSAGES</i>           | Determine the locale that should be used to affect the format and contents of diagnostic messages written to standard error.                                                                                                                                                                                    |
| 10944 |                              |                                                                                                                                                                                                                                                                                                                 |
| 10945 |                              |                                                                                                                                                                                                                                                                                                                 |
| 10946 | XSI <i>NLSPATH</i>           | Determine the location of message catalogs for the processing of <i>LC_MESSAGES</i> .                                                                                                                                                                                                                           |
| 10947 | <b>ASYNCHRONOUS EVENTS</b>   |                                                                                                                                                                                                                                                                                                                 |
| 10948 |                              | If the <b>-k</b> option is specified, created files shall be retained. Otherwise, the default action occurs.                                                                                                                                                                                                    |
| 10949 | <b>STDOUT</b>                |                                                                                                                                                                                                                                                                                                                 |
| 10950 |                              | Unless the <b>-s</b> option is used, the standard output shall consist of one line per file created, with a format as follows:                                                                                                                                                                                  |
| 10951 |                              |                                                                                                                                                                                                                                                                                                                 |
| 10952 |                              | "%d\n", <file size in bytes>                                                                                                                                                                                                                                                                                    |
| 10953 | <b>STDERR</b>                |                                                                                                                                                                                                                                                                                                                 |
| 10954 |                              | The standard error shall be used only for diagnostic messages.                                                                                                                                                                                                                                                  |

**10955 OUTPUT FILES**

10956 The output files shall contain portions of the original input file; otherwise, unchanged.

**10957 EXTENDED DESCRIPTION**

10958 None.

**10959 EXIT STATUS**

10960 The following exit values shall be returned:

10961 0 Successful completion.

10962 >0 An error occurred.

**10963 CONSEQUENCES OF ERRORS**

10964 By default, created files shall be removed if an error occurs. When the **-k** option is specified,  
10965 created files shall not be removed if an error occurs.

**10966 APPLICATION USAGE**

10967 None.

**10968 EXAMPLES**

10969 1. This example creates four files, **cobol00** ... **cobol03**:

10970 `csplit -f cobol file '/procedure division/' /par5./ /par16./`

10971 After editing the split files, they can be recombined as follows:

10972 `cat cobol0[0-3] > file`

10973 Note that this example overwrites the original file.

10974 2. This example would split the file after the first 99 lines, and every 100 lines thereafter, up  
10975 to 9 999 lines; this is because lines in the file are numbered from 1 rather than zero, for  
10976 historical reasons:

10977 `csplit -k file 100 {99}`

10978 3. Assuming that **prog.c** follows the C-language coding convention of ending routines with a  
10979 '**}**' at the beginning of the line, this example creates a file containing each separate C  
10980 routine (up to 21) in **prog.c**:

10981 `csplit -k prog.c '%main(%' '/^}/+1' {20}`

**10982 RATIONALE**

10983 The **-n** option was added to extend the range of filenames that could be handled.

10984 Consideration was given to adding a **-a** flag to use the alphabetic filename generation used by  
10985 the historical *split* utility, but the functionality added by the **-n** option was deemed to make  
10986 alphabetic naming unnecessary.

**10987 FUTURE DIRECTIONS**

10988 None.

**10989 SEE ALSO**

10990 *sed*, *split*

**10991 CHANGE HISTORY**

10992 First released in Issue 2.

**10993 Issue 5**

10994 The FUTURE DIRECTIONS section is added.

**10995 Issue 6**

10996 This utility is marked as part of the User Portability Utilities option.

10997 The APPLICATION USAGE section is added.

10998 The description of regular expression operands is changed to align with the IEEE P1003.2b draft standard.  
10999

11000 The normative text is reworded to avoid use of the term “must” for application requirements.

**11001 NAME**

11002        **ctags** — create a tags file (**DEVELOPMENT, FORTRAN**)

**11003 SYNOPSIS**

11004 UP      **ctags [-a][-f tagsfile] pathname ...**

11005        **ctags -x pathname ...**

11006

**11007 DESCRIPTION**

11008        The **ctags** 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.

11011        The **ctags** utility shall write a **tagsfile** or an index of objects from C-language or FORTRAN source files specified by the **pathname** operands. The **tagsfile** shall list the locators of language-specific objects within the source files. A locator consists of a name, pathname, and either a search pattern or a line number that can be used in searching for the object definition. The objects that shall be recognized are specified in the EXTENDED DESCRIPTION section.

**11016 OPTIONS**

11017        The **ctags** utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section 12.2, Utility Syntax Guidelines.

11019        The following options shall be supported:

11020        **-a**        Append to **tagsfile**.

11021        **-f tagsfile**        Write the object locator lists into **tagsfile** instead of the default file named **tags** in the current directory.

11023        **-x**        Produce a list of object names, the line number, and filename in which each is defined, as well as the text of that line, and write this to the standard output. A **tagsfile** shall not be created when **-x** is specified.

**11026 OPERANDS**

11027        The following **pathname** operands are supported:

11028        **file.c**        Files with basenames ending with the **.c** suffix shall be treated as C-language source code. Such files that are not valid input to **c99** produce unspecified results.

11030        **file.h**        Files with basenames ending with the **.h** suffix shall be treated as C-language source code. Such files that are not valid input to **c99** produce unspecified results.

11032        **file.f**        Files with basenames ending with the **.f** suffix shall be treated as FORTRAN-language source code. Such files that are not valid input to **fort77** produce unspecified results.

11035        The handling of other files is implementation-defined.

**11036 STDIN**

11037        See the INPUT FILES section.

**11038 INPUT FILES**

11039        The input files shall be text files containing source code in the language indicated by the operand filename suffixes.

**11041 ENVIRONMENT VARIABLES**

11042 The following environment variables shall affect the execution of *ctags*:

11043 **LANG** Provide a default value for the internationalization variables that are unset or null.  
11044 (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2,  
11045 Internationalization Variables for the precedence of internationalization variables  
11046 used to determine the values of locale categories.)

11047 **LC\_ALL** If set to a non-empty string value, override the values of all the other  
11048 internationalization variables.

**11049 *LC\_COLLATE***

11050 Determine the order in which output is sorted for the **-x** option. The POSIX locale  
11051 determines the order in which the *tagsfile* is written.

11052 **LC\_CTYPE** Determine the locale for the interpretation of sequences of bytes of text data as  
11053 characters (for example, single-byte as opposed to multi-byte characters in  
11054 arguments and input files). When processing C-language source code, if the locale  
11055 is not compatible with the C locale described by the ISO C standard, the results are  
11056 unspecified.

**11057 *LC\_MESSAGES***

11058 Determine the locale that should be used to affect the format and contents of  
11059 diagnostic messages written to standard error.

11060 **XSI *NLSPATH*** Determine the location of message catalogs for the processing of *LC\_MESSAGES*.

**11061 ASYNCHRONOUS EVENTS**

11062 Default.

**11063 *STDOUT***

11064 The list of object name information produced by the **-x** option shall be written to standard  
11065 output in the following format:

11066 "%s %d %s %s", <object-name>, <line-number>, <filename>, <text>

11067 where <text> is the text of line <line-number> of file <filename>.

**11068 *STDERR***

11069 The standard error shall be used only for diagnostic messages.

**11070 *OUTPUT FILES***

11071 When the **-x** option is not specified, the format of the output file shall be:

11072 "%s\t%s\t/%s/\n", <identifier>, <filename>, <pattern>

11073 where <pattern> is a search pattern that could be used by an editor to find the defining instance  
11074 of <identifier> in <filename> (where *defining instance* is indicated by the declarations listed in the  
11075 EXTENDED DESCRIPTION).

11076 An optional circumflex ('^') can be added as a prefix to <pattern>, and an optional dollar sign  
11077 can be appended to <pattern> to indicate that the pattern is anchored to the beginning (end) of a  
11078 line of text. Any slash or backslash characters in <pattern> shall be preceded by a backslash  
11079 character. The anchoring circumflex, dollar sign, and escaping backslash characters shall not be  
11080 considered part of the search pattern. All other characters in the search pattern shall be  
11081 considered literal characters.

- 11082        An alternative format is:
- 11083        "%s\t%s\t?%s?\n", <identifier>, <filename>, <pattern>
- 11084        which is identical to the first format except that slashes in <pattern> shall not be preceded by  
11085        escaping backslash characters, and question mark characters in <pattern> shall be preceded by  
11086        backslash characters.
- 11087        A second alternative format is:
- 11088        "%s\t%s\t%d\n", <identifier>, <filename>, <lineno>
- 11089        where <lineno> is a decimal line number that could be used by an editor to find <identifier> in  
11090        <filename>.
- 11091        Neither alternative format shall be produced by *ctags* when it is used as described by  
11092        IEEE Std 1003.1-2001, but the standard utilities that process tags files shall be able to process  
11093        those formats as well as the first format.
- 11094        In any of these formats, the file shall be sorted by identifier, based on the collation sequence in  
11095        the POSIX locale.
- 11096 **EXTENDED DESCRIPTION**
- 11097        If the operand identifies C-language source, the *ctags* utility shall attempt to produce an output  
11098        line for each of the following objects:
- 11099            • Function definitions
- 11100            • Type definitions
- 11101            • Macros with arguments
- 11102        It may also produce output for any of the following objects:
- 11103            • Function prototypes
- 11104            • Structures
- 11105            • Unions
- 11106            • Global variable definitions
- 11107            • Enumeration types
- 11108            • Macros without arguments
- 11109            • #define statements
- 11110            • #line statements
- 11111        Any #if and #ifdef statements shall produce no output. The tag **main** is treated specially in C  
11112        programs. The tag formed shall be created by prefixing **M** to the name of the file, with the  
11113        trailing .c, and leading pathname components (if any) removed.
- 11114        On systems that do not support the C-Language Development Utilities option, *ctags* produces  
11115        unspecified results for C-language source code files. It should write to standard error a message  
11116        identifying this condition and cause a non-zero exit status to be produced.
- 11117        If the operand identifies FORTRAN source, the *ctags* utility shall produce an output line for each  
11118        function definition. It may also produce output for any of the following objects:
- 11119            • Subroutine definitions
- 11120            • COMMON statements

- 11121           • PARAMETER statements  
11122           • DATA and BLOCK DATA statements  
11123           • Statement numbers
- 11124         On systems that do not support the FORTRAN Development Utilities option, *ctags* produces  
11125         unspecified results for FORTRAN source code files. It should write to standard error a message  
11126         identifying this condition and cause a non-zero exit status to be produced.
- 11127         It is implementation-defined what other objects (including duplicate identifiers) produce output.
- 11128 **EXIT STATUS**
- 11129         The following exit values shall be returned:
- 11130           0 Successful completion.  
11131           >0 An error occurred.
- 11132 **CONSEQUENCES OF ERRORS**
- 11133         Default.
- 11134 **APPLICATION USAGE**
- 11135         The output with **-x** is meant to be a simple index that can be written out as an off-line readable  
11136         function index. If the input files to *ctags* (such as .c files) were not created using the same locale  
11137         as that in effect when *ctags -x* is run, results might not be as expected.
- 11138         The description of C-language processing says “attempts to” because the C language can be  
11139         greatly confused, especially through the use of #defines, and this utility would be of no use if  
11140         the real C preprocessor were run to identify them. The output from *ctags* may be fooled and  
11141         incorrect for various constructs.
- 11142 **EXAMPLES**
- 11143         None.
- 11144 **RATIONALE**
- 11145         The option list was significantly reduced from that provided by historical implementations. The  
11146         **-F** option was omitted as redundant, since it is the default. The **-B** option was omitted as being  
11147         of very limited usefulness. The **-t** option was omitted since the recognition of **typedefs** is now  
11148         required for C source files. The **-u** option was omitted because the update function was judged  
11149         to be not only inefficient, but also rarely needed.
- 11150         An early proposal included a **-w** option to suppress warning diagnostics. Since the types of such  
11151         diagnostics could not be described, the option was omitted as being not useful.
- 11152         The text for *LC\_CTYPE* about compatibility with the C locale acknowledges that the ISO C  
11153         standard imposes requirements on the locale used to process C source. This could easily be a  
11154         superset of that known as “the C locale” by way of implementation extensions, or one of a few  
11155         alternative locales for systems supporting different codesets. No statement is made for  
11156         FORTRAN because the ANSI X3.9-1978 standard (FORTRAN 77) does not (yet) define a similar  
11157         locale concept. However, a general rule in this volume of IEEE Std 1003.1-2001 is that any time  
11158         that locales do not match (preparing a file for one locale and processing it in another), the results  
11159         are suspect.
- 11160         The collation sequence of the tags file is not affected by *LC\_COLLATE* because it is typically not  
11161         used by human readers, but only by programs such as *vi* to locate the tag within the source files.  
11162         Using the POSIX locale eliminates some of the problems of coordinating locales between the  
11163         *ctags* file creator and the *vi* file reader.

11164 Historically, the tags file has been used only by *ex* and *vi*. However, the format of the tags file  
11165 has been published to encourage other programs to use the tags in new ways. The format allows  
11166 either patterns or line numbers to find the identifiers because the historical *vi* recognizes either.  
11167 The *ctags* utility does not produce the format using line numbers because it is not useful  
11168 following any source file changes that add or delete lines. The documented search patterns  
11169 match historical practice. It should be noted that literal leading circumflex or trailing dollar-sign  
11170 characters in the search pattern will only behave correctly if anchored to the beginning of the  
11171 line or end of the line by an additional circumflex or dollar-sign character.

11172 Historical implementations also understand the objects used by the languages Pascal and  
11173 sometimes LISP, and they understand the C source output by *lex* and *yacc*. The *ctags* utility is  
11174 not required to accommodate these languages, although implementors are encouraged to do so.

11175 The following historical option was not specified, as *vgrind* is not included in this volume of  
11176 IEEE Std 1003.1-2001:

11177 **-v** If the **-v** flag is given, an index of the form expected by *vgrind* is produced on the  
11178 standard output. This listing contains the function name, filename, and page  
11179 number (assuming 64-line pages). Since the output is sorted into lexicographic  
11180 order, it may be desired to run the output through *sort -f*. Sample use:

```
11181 ctags -v files | sort -f > index vgrind -x index
```

11182 The special treatment of the tag **main** makes the use of *ctags* practical in directories with more  
11183 than one program.

#### 11184 FUTURE DIRECTIONS

11185 None.

#### 11186 SEE ALSO

11187 *c99*, *fort77*, *vi*

#### 11188 CHANGE HISTORY

11189 First released in Issue 4.

#### 11190 Issue 5

11191 The FUTURE DIRECTIONS section is added.

#### 11192 Issue 6

11193 This utility is marked as part of the User Portability Utilities option.

11194 The OUTPUT FILES section is changed to align with the IEEE P1003.2b draft standard.

11195 The normative text is reworded to avoid use of the term “must” for application requirements.

11196 IEEE PASC Interpretation 1003.2 #168 is applied, changing “create” to “write” in the  
11197 DESCRIPTION.

## 11198 NAME

11199        cut — cut out selected fields of each line of a file

## 11200 SYNOPSIS

11201        cut -b *list* [-n] [*file* ...]11202        cut -c *list* [*file* ...]11203        cut -f *list* [-d *delim*] [-s] [*file* ...]

## 11204 DESCRIPTION

11205        The *cut* utility shall cut out bytes (-b option), characters (-c option), or character-delimited fields  
11206        (-f option) from each line in one or more files, concatenate them, and write them to standard  
11207        output.

## 11208 OPTIONS

11209        The *cut* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section  
11210        12.2, Utility Syntax Guidelines.

11211        The application shall ensure that the option-argument *list* (see options -b, -c, and -f below) is a  
11212        comma-separated list or <blank>-separated list of positive numbers and ranges. Ranges can be  
11213        in three forms. The first is two positive numbers separated by a hyphen (*low-high*), which  
11214        represents all fields from the first number to the second number. The second is a positive  
11215        number preceded by a hyphen (-*high*), which represents all fields from field number 1 to that  
11216        number. The third is a positive number followed by a hyphen (*low-*), which represents that  
11217        number to the last field, inclusive. The elements in *list* can be repeated, can overlap, and can be  
11218        specified in any order, but the bytes, characters, or fields selected shall be written in the order of  
11219        the input data. If an element appears in the selection list more than once, it shall be written  
11220        exactly once.

11221        The following options shall be supported:

11222        **-b** *list*        Cut based on a *list* of bytes. Each selected byte shall be output unless the -n option  
11223        is also specified. It shall not be an error to select bytes not present in the input line.

11224        **-c** *list*        Cut based on a *list* of characters. Each selected character shall be output. It shall  
11225        not be an error to select characters not present in the input line.

11226        **-d** *delim*      Set the field delimiter to the character *delim*. The default is the <tab>.

11227        **-f** *list*        Cut based on a *list* of fields, assumed to be separated in the file by a delimiter  
11228        character (see -d). Each selected field shall be output. Output fields shall be  
11229        separated by a single occurrence of the field delimiter character. Lines with no field  
11230        delimiters shall be passed through intact, unless -s is specified. It shall not be an  
11231        error to select fields not present in the input line.

11232        **-n**            Do not split characters. When specified with the -b option, each element in *list* of  
11233        the form *low-high* (hyphen-separated numbers) shall be modified as follows:

- 11234        • If the byte selected by *low* is not the first byte of a character, *low* shall be  
11235        decremented to select the first byte of the character originally selected by *low*.  
11236        If the byte selected by *high* is not the last byte of a character, *high* shall be  
11237        decremented to select the last byte of the character prior to the character  
11238        originally selected by *high*, or zero if there is no prior character. If the resulting  
11239        range element has *high* equal to zero or *low* greater than *high*, the list element  
11240        shall be dropped from *list* for that input line without causing an error.

11241        Each element in *list* of the form *low-* shall be treated as above with *high* set to the  
11242        number of bytes in the current line, not including the terminating <newline>. Each

11243                    element in *list* of the form *-high* shall be treated as above with *low* set to 1. Each  
11244                    element in *list* of the form *num* (a single number) shall be treated as above with *low*  
11245                    set to *num* and *high* set to *num*.

11246                **-s**                    Suppress lines with no delimiter characters, when used with the **-f** option. Unless  
11247                    specified, lines with no delimiters shall be passed through untouched.

## 11248 OPERANDS

11249                    The following operand shall be supported:

11250                *file*                    A pathname of an input file. If no *file* operands are specified, or if a *file* operand is  
11251                    '*-*', the standard input shall be used.

## 11252 STDIN

11253                    The standard input shall be used only if no *file* operands are specified, or if a *file* operand is '*-*'.  
11254                    See the INPUT FILES section.

## 11255 INPUT FILES

11256                    The input files shall be text files, except that line lengths shall be unlimited.

## 11257 ENVIRONMENT VARIABLES

11258                    The following environment variables shall affect the execution of *cut*:

11259                *LANG*                    Provide a default value for the internationalization variables that are unset or null.  
11260                    (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2,  
11261                    Internationalization Variables for the precedence of internationalization variables  
11262                    used to determine the values of locale categories.)

11263                *LC\_ALL*                    If set to a non-empty string value, override the values of all the other  
11264                    internationalization variables.

11265                *LC\_CTYPE*                    Determine the locale for the interpretation of sequences of bytes of text data as  
11266                    characters (for example, single-byte as opposed to multi-byte characters in  
11267                    arguments and input files).

### 11268 *LC\_MESSAGES*

11269                    Determine the locale that should be used to affect the format and contents of  
11270                    diagnostic messages written to standard error.

11271 XSI            *NLSPATH*                    Determine the location of message catalogs for the processing of *LC\_MESSAGES*.

## 11272 ASYNCHRONOUS EVENTS

11273                    Default.

## 11274 STDOUT

11275                    The *cut* utility output shall be a concatenation of the selected bytes, characters, or fields (one of  
11276                    the following):

11277                    "%s\n", <concatenation of bytes>

11278                    "%s\n", <concatenation of characters>

11279                    "%s\n", <concatenation of fields and field delimiters>

## 11280 STDERR

11281                    The standard error shall be used only for diagnostic messages.

## 11282 OUTPUT FILES

11283                    None.

## 11284 EXTENDED DESCRIPTION

11285 None.

## 11286 EXIT STATUS

11287 The following exit values shall be returned:

11288 0 All input files were output successfully.

11289 &gt;0 An error occurred.

## 11290 CONSEQUENCES OF ERRORS

11291 Default.

## 11292 APPLICATION USAGE

11293 Earlier versions of the *cut* utility worked in an environment where bytes and characters were  
11294 considered equivalent (modulo <backspace> and <tab> processing in some implementations). In  
11295 the extended world of multi-byte characters, the new **-b** option has been added. The **-n** option  
11296 (used with **-b**) allows it to be used to act on bytes rounded to character boundaries. The  
11297 algorithm specified for **-n** guarantees that:

11298 `cut -b 1-500 -n file > file1`11299 `cut -b 501- -n file > file2`

11300 ends up with all the characters in **file** appearing exactly once in **file1** or **file2**. (There is,  
11301 however, a <newline> in both **file1** and **file2** for each <newline> in **file**.)

## 11302 EXAMPLES

11303 Examples of the option qualifier list:

11304 1,4,7 Select the first, fourth, and seventh bytes, characters, or fields and field delimiters.

11305 1-3,8 Equivalent to 1,2,3,8.

11306 -5,10 Equivalent to 1,2,3,4,5,10.

11307 3- Equivalent to third to last, inclusive.

11308 The *low-high* forms are not always equivalent when used with **-b** and **-n** and multi-byte  
11309 characters; see the description of **-n**.

11310 The following command:

11311 `cut -d : -f 1,6 /etc/passwd`

11312 reads the System V password file (user database) and produces lines of the form:

11313 `<user ID>:<home directory>`

11314 Most utilities in this volume of IEEE Std 1003.1-2001 work on text files. The *cut* utility can be  
11315 used to turn files with arbitrary line lengths into a set of text files containing the same data. The  
11316 *paste* utility can be used to create (or recreate) files with arbitrary line lengths. For example, if **file**  
11317 contains long lines:

11318 `cut -b 1-500 -n file > file1`11319 `cut -b 501- -n file > file2`

11320 creates **file1** (a text file) with lines no longer than 500 bytes (plus the <newline>) and **file2** that  
11321 contains the remainder of the data from **file**. (Note that **file2** is not a text file if there are lines in  
11322 **file** that are longer than 500 + {LINE\_MAX} bytes.) The original file can be recreated from **file1**  
11323 and **file2** using the command:

11324 `paste -d "\0" file1 file2 > file`

**11325 RATIONALE**

11326 Some historical implementations do not count <backspace>s in determining character counts  
11327 with the **-c** option. This may be useful for using *cut* for processing *nroff* output. It was  
11328 deliberately decided not to have the **-c** option treat either <backspace>s or <tab>s in any special  
11329 fashion. The *fold* utility does treat these characters specially.

11330 Unlike other utilities, some historical implementations of *cut* exit after not finding an input file,  
11331 rather than continuing to process the remaining *file* operands. This behavior is prohibited by this  
11332 volume of IEEE Std 1003.1-2001, where only the exit status is affected by this problem.

11333 The behavior of *cut* when provided with either mutually-exclusive options or options that do  
11334 not work logically together has been deliberately left unspecified in favor of global wording in  
11335 Section 1.11 (on page 20).

11336 The OPTIONS section was changed in response to IEEE PASC Interpretation 1003.2 #149. The  
11337 change represents historical practice on all known systems. The original standard was  
11338 ambiguous on the nature of the output.

11339 The *list* option-arguments are historically used to select the portions of the line to be written, but  
11340 do not affect the order of the data. For example:

11341 `echo abcdefghi | cut -c6,2,4-7,1`

11342 yields "abdefg".

11343 A proposal to enhance *cut* with the following option:

11344 **-o** Preserve the selected field order. When this option is specified, each byte, character, or field  
11345 (or ranges of such) shall be written in the order specified by the *list* option-argument, even if  
11346 this requires multiple outputs of the same bytes, characters, or fields.

11347 was rejected because this type of enhancement is outside the scope of the IEEE P1003.2b draft  
11348 standard.

**11349 FUTURE DIRECTIONS**

11350 None.

**11351 SEE ALSO**

11352 *grep*, *paste*, Section 2.5 (on page 33)

**11353 CHANGE HISTORY**

11354 First released in Issue 2.

**11355 Issue 6**

11356 The OPTIONS section is changed to align with the IEEE P1003.2b draft standard.

11357 The normative text is reworded to avoid use of the term “must” for application requirements.

## 11358 NAME

11359 cxref — generate a C-language program cross-reference table (**DEVELOPMENT**)

## 11360 SYNOPSIS

```
11361 XSI cxref [-cs][-o file][-w num] [-D name[=def]]...[-I dir]...
11362 [-U name]... file ...
```

11363

## 11364 DESCRIPTION

11365 The *cxref* utility shall analyze a collection of C-language *files* and attempt to build a cross-  
 11366 reference table. Information from **#define** lines shall be included in the symbol table. A sorted  
 11367 listing shall be written to standard output of all symbols (auto, static, and global) in each *file*  
 11368 separately, or with the **-c** option, in combination. Each symbol shall contain an asterisk before  
 11369 the declaring reference.

## 11370 OPTIONS

11371 The *cxref* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section  
 11372 12.2, Utility Syntax Guidelines, except that the order of the **-D**, **-I**, and **-U** options (which are  
 11373 identical to their interpretation by *c99*) is significant. The following options shall be supported:

- |                             |                                                                                                                                                  |
|-----------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------|
| 11374 <b>-c</b>             | Write a combined cross-reference of all input files.                                                                                             |
| 11375 <b>-s</b>             | Operate silently; do not print input filenames.                                                                                                  |
| 11376 <b>-o</b> <i>file</i> | Direct output to named <i>file</i> .                                                                                                             |
| 11377 <b>-w</b> <i>num</i>  | Format output no wider than <i>num</i> (decimal) columns. This option defaults to 80 if<br>11378 <i>num</i> is not specified or is less than 51. |
| 11379 <b>-D</b>             | Equivalent to <i>c99</i> .                                                                                                                       |
| 11380 <b>-I</b>             | Equivalent to <i>c99</i> .                                                                                                                       |
| 11381 <b>-U</b>             | Equivalent to <i>c99</i> .                                                                                                                       |

## 11382 OPERANDS

11383 The following operand shall be supported:

11384 *file*       A pathname of a C-language source file.

## 11385 STDIN

11386 Not used.

## 11387 INPUT FILES

11388 The input files are C-language source files.

## 11389 ENVIRONMENT VARIABLES

11390 The following environment variables shall affect the execution of *cxref*:

- |                         |                                                                                                                                                                                                                                                                                                                                        |
|-------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 11391 <b>LANG</b>       | Provide a default value for the internationalization variables that are unset or null.<br>11392   (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2,<br>11393   Internationalization Variables for the precedence of internationalization variables<br>11394   used to determine the values of locale categories.) |
| 11395 <b>LC_ALL</b>     | If set to a non-empty string value, override the values of all the other<br>11396   internationalization variables.                                                                                                                                                                                                                    |
| 11397 <b>LC_COLLATE</b> | Determine the locale for the ordering of the output.                                                                                                                                                                                                                                                                                   |
| 11399 <b>LC_CTYPE</b>   | Determine the locale for the interpretation of sequences of bytes of text data as<br>11400   characters (for example, single-byte as opposed to multi-byte characters in                                                                                                                                                               |

- 11401 arguments and input files).
- 11402 ***LC\_MESSAGES***
- 11403 Determine the locale that should be used to affect the format and contents of
- 11404 diagnostic messages written to standard error.
- 11405 ***NLSPATH*** Determine the location of message catalogs for the processing of *LC\_MESSAGES*.
- 11406 **ASYNCHRONOUS EVENTS**
- 11407 Default.
- 11408 **STDOUT**
- 11409 The standard output shall be used for the cross-reference listing, unless the **-o** option is used to
- 11410 select a different output file.
- 11411 The format of standard output is unspecified, except that the following information shall be
- 11412 included:
- 11413 • If the **-c** option is not specified, each portion of the listing shall start with the name of the
- 11414 input file on a separate line.
- 11415 • The name line shall be followed by a sorted list of symbols, each with its associated location
- 11416 pathname, the name of the function in which it appears (if it is not a function name itself),
- 11417 and line number references.
- 11418 • Each line number may be preceded by an asterisk ('\*') flag, meaning that this is the
- 11419 declaring reference. Other single-character flags, with implementation-defined meanings,
- 11420 may be included.
- 11421 **STDERR**
- 11422 The standard error shall be used only for diagnostic messages.
- 11423 **OUTPUT FILES**
- 11424 The output file named by the **-o** option shall be used instead of standard output.
- 11425 **EXTENDED DESCRIPTION**
- 11426 None.
- 11427 **EXIT STATUS**
- 11428 The following exit values shall be returned:
- 11429 0 Successful completion.
- 11430 >0 An error occurred.
- 11431 **CONSEQUENCES OF ERRORS**
- 11432 Default.
- 11433 **APPLICATION USAGE**
- 11434 None.
- 11435 **EXAMPLES**
- 11436 None.
- 11437 **RATIONALE**
- 11438 None.
- 11439 **FUTURE DIRECTIONS**
- 11440 None.

11441 **SEE ALSO**

11442           *c99*

11443 **CHANGE HISTORY**

11444           First released in Issue 2.

11445 **Issue 5**

11446           In the SYNOPSIS, **[−U dir]** is changed to **[−U name]**.

11447 **Issue 6**

11448           The APPLICATION USAGE section is added.

**11449 NAME**

11450 date — write the date and time

**11451 SYNOPSIS**

11452 date [-u] [+format]

11453 XSI date [-u] mmddhhmm[[cc]YY]

11454

**11455 DESCRIPTION**

11456 XSI The *date* utility shall write the date and time to standard output or attempt to set the system date and time. By default, the current date and time shall be written. If an operand beginning with '+' is specified, the output format of *date* shall be controlled by the conversion specifications and other text in the operand.

**11460 OPTIONS**

11461 The *date* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section 12.2, Utility Syntax Guidelines.

11463 The following option shall be supported:

11464 **-u** Perform operations as if the *TZ* environment variable was set to the string "UTC0", or its equivalent historical value of "GMT0". Otherwise, *date* shall use the timezone indicated by the *TZ* environment variable or the system default if that variable is unset or null.

**11468 OPERANDS**

11469 The following operands shall be supported:

11470 **+format** When the format is specified, each conversion specifier shall be replaced in the standard output by its corresponding value. All other characters shall be copied to the output without change. The output shall always be terminated with a <newline>.

**11474 Conversion Specifications**

11475 %a Locale's abbreviated weekday name.

11476 %A Locale's full weekday name.

11477 %b Locale's abbreviated month name.

11478 %B Locale's full month name.

11479 %c Locale's appropriate date and time representation.

11480 %C Century (a year divided by 100 and truncated to an integer) as a decimal number [00,99].

11482 %d Day of the month as a decimal number [01,31].

11483 %D Date in the format *mm/dd/yy*.

11484 %e Day of the month as a decimal number [1,31] in a two-digit field with leading space character fill.

11486 %h A synonym for %b.

11487 %H Hour (24-hour clock) as a decimal number [00,23].

11488 %I Hour (12-hour clock) as a decimal number [01,12].

|       |    |                                                                                                                                                                                                                                                                                               |
|-------|----|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 11489 | %j | Day of the year as a decimal number [001,366].                                                                                                                                                                                                                                                |
| 11490 | %m | Month as a decimal number [01,12].                                                                                                                                                                                                                                                            |
| 11491 | %M | Minute as a decimal number [00,59].                                                                                                                                                                                                                                                           |
| 11492 | %n | A <newline>.                                                                                                                                                                                                                                                                                  |
| 11493 | %p | Locale's equivalent of either AM or PM.                                                                                                                                                                                                                                                       |
| 11494 | %r | 12-hour clock time [01,12] using the AM/PM notation; in the POSIX locale, this shall be equivalent to %I:%M:%S %p.                                                                                                                                                                            |
| 11495 |    |                                                                                                                                                                                                                                                                                               |
| 11496 | %S | Seconds as a decimal number [00,60].                                                                                                                                                                                                                                                          |
| 11497 | %t | A <tab>.                                                                                                                                                                                                                                                                                      |
| 11498 | %T | 24-hour clock time [00,23] in the format HH:MM:SS.                                                                                                                                                                                                                                            |
| 11499 | %u | Weekday as a decimal number [1,7] (1=Monday).                                                                                                                                                                                                                                                 |
| 11500 | %U | Week of the year (Sunday as the first day of the week) as a decimal number [00,53]. All days in a new year preceding the first Sunday shall be considered to be in week 0.                                                                                                                    |
| 11501 |    |                                                                                                                                                                                                                                                                                               |
| 11502 |    |                                                                                                                                                                                                                                                                                               |
| 11503 | %V | Week of the year (Monday as the first day of the week) as a decimal number [01,53]. If the week containing January 1 has four or more days in the new year, then it shall be considered week 1; otherwise, it shall be the last week of the previous year, and the next week shall be week 1. |
| 11504 |    |                                                                                                                                                                                                                                                                                               |
| 11505 |    |                                                                                                                                                                                                                                                                                               |
| 11506 |    |                                                                                                                                                                                                                                                                                               |
| 11507 | %w | Weekday as a decimal number [0,6] (0=Sunday).                                                                                                                                                                                                                                                 |
| 11508 | %W | Week of the year (Monday as the first day of the week) as a decimal number [00,53]. All days in a new year preceding the first Monday shall be considered to be in week 0.                                                                                                                    |
| 11509 |    |                                                                                                                                                                                                                                                                                               |
| 11510 |    |                                                                                                                                                                                                                                                                                               |
| 11511 | %x | Locale's appropriate date representation.                                                                                                                                                                                                                                                     |
| 11512 | %X | Locale's appropriate time representation.                                                                                                                                                                                                                                                     |
| 11513 | %Y | Year within century [00,99].                                                                                                                                                                                                                                                                  |
| 11514 | %Y | Year with century as a decimal number.                                                                                                                                                                                                                                                        |
| 11515 | %Z | Timezone name, or no characters if no timezone is determinable.                                                                                                                                                                                                                               |
| 11516 | %% | A percent sign character.                                                                                                                                                                                                                                                                     |
| 11517 |    | See the Base Definitions volume of IEEE Std 1003.1-2001, Section 7.3.5, LC_TIME                                                                                                                                                                                                               |
| 11518 |    | for the conversion specifier values in the POSIX locale.                                                                                                                                                                                                                                      |

### 11519 Modified Conversion Specifications

Some conversion specifiers can be modified by the `E` and `O` modifier characters to indicate a different format or specification as specified in the *LC\_TIME* locale description (see the Base Definitions volume of IEEE Std 1003.1-2001, Section 7.3.5, LC\_TIME). If the corresponding keyword (see `era`, `era_year`, `era_d_fmt`, and `alt_digits` in the Base Definitions volume of IEEE Std 1003.1-2001, Section 7.3.5, LC\_TIME) is not specified or not supported for the current locale, the unmodified conversion specifier value shall be used.

|       |     |                                                                |
|-------|-----|----------------------------------------------------------------|
| 11527 | %Ec | Locale's alternative appropriate date and time representation. |
|-------|-----|----------------------------------------------------------------|

|           |                        |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
|-----------|------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 11528     | %EC                    | The name of the base year (period) in the locale's alternative representation.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| 11529     |                        |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| 11530     | %Ex                    | Locale's alternative date representation.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
| 11531     | %EX                    | Locale's alternative time representation.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
| 11532     | %Ey                    | Offset from %EC (year only) in the locale's alternative representation.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| 11533     | %EY                    | Full alternative year representation.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| 11534     | %Od                    | Day of month using the locale's alternative numeric symbols.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
| 11535     | %Oe                    | Day of month using the locale's alternative numeric symbols.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
| 11536     | %OH                    | Hour (24-hour clock) using the locale's alternative numeric symbols.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
| 11537     | %OI                    | Hour (12-hour clock) using the locale's alternative numeric symbols.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
| 11538     | %Om                    | Month using the locale's alternative numeric symbols.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| 11539     | %OM                    | Minutes using the locale's alternative numeric symbols.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| 11540     | %OS                    | Seconds using the locale's alternative numeric symbols.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| 11541     | %Ou                    | Weekday as a number in the locale's alternative representation (Monday = 1).                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
| 11542     |                        |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| 11543     | %OU                    | Week number of the year (Sunday as the first day of the week) using the locale's alternative numeric symbols.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| 11544     |                        |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| 11545     | %OV                    | Week number of the year (Monday as the first day of the week, rules corresponding to %v), using the locale's alternative numeric symbols.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
| 11546     |                        |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| 11547     | %Ow                    | Weekday as a number in the locale's alternative representation (Sunday = 0).                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
| 11548     |                        |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| 11549     | %OW                    | Week number of the year (Monday as the first day of the week) using the locale's alternative numeric symbols.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| 11550     |                        |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| 11551     | %Oy                    | Year (offset from %C) in alternative representation.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
| 11552 XSI | <b>mddhhmm[[cc]yy]</b> | Attempt to set the system date and time from the value given in the operand. This is only possible if the user has appropriate privileges and the system permits the setting of the system date and time. The first <i>mm</i> is the month (number); <i>dd</i> is the day (number); <i>hh</i> is the hour (number, 24-hour system); the second <i>mm</i> is the minute (number); <i>cc</i> is the century and is the first two digits of the year (this is optional); <i>yy</i> is the last two digits of the year and is optional. If century is not specified, then values in the range [69,99] shall refer to years 1969 to 1999 inclusive, and values in the range [00,68] shall refer to years 2000 to 2068 inclusive. The current year is the default if <i>yy</i> is omitted. |
| 11553     |                        |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| 11554     |                        |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| 11555     |                        |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| 11556     |                        |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| 11557     |                        |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| 11558     |                        |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| 11559     |                        |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| 11560     |                        |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| 11561     |                        |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| 11562     | <b>Note:</b>           | It is expected that in a future version of IEEE Std 1003.1-2001 the default century inferred from a 2-digit year will change. (This would apply to all commands accepting a 2-digit year as input.)                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| 11563     |                        |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| 11564     |                        |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| 11565     | <b>STDIN</b>           |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| 11566     | Not used.              |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |

**11567 INPUT FILES**

11568 None.

**11569 ENVIRONMENT VARIABLES**

11570 The following environment variables shall affect the execution of *date*:

11571 *LANG* Provide a default value for the internationalization variables that are unset or null.  
(See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2, Internationalization Variables for the precedence of internationalization variables used to determine the values of locale categories.)

11575 *LC\_ALL* If set to a non-empty string value, override the values of all the other  
11576 internationalization variables.

11577 *LC\_CTYPE* Determine the locale for the interpretation of sequences of bytes of text data as  
11578 characters (for example, single-byte as opposed to multi-byte characters in  
11579 arguments).

**11580 *LC\_MESSAGES***

11581 Determine the locale that should be used to affect the format and contents of  
11582 diagnostic messages written to standard error.

11583 *LC\_TIME* Determine the format and contents of date and time strings written by *date*.

11584 XSI *NLSPATH* Determine the location of message catalogs for the processing of *LC\_MESSAGES*.

11585 *TZ* Determine the timezone in which the time and date are written, unless the **-u**  
11586 option is specified. If the *TZ* variable is unset or null and **-u** is not specified, an  
11587 unspecified system default timezone is used.

**11588 ASYNCHRONOUS EVENTS**

11589 Default.

**11590 STDOUT**

11591 When no formatting operand is specified, the output in the POSIX locale shall be equivalent to  
11592 specifying:

11593 `date "+%a %b %e %H:%M:%S %Z %Y"`

**11594 STDERR**

11595 The standard error shall be used only for diagnostic messages.

**11596 OUTPUT FILES**

11597 None.

**11598 EXTENDED DESCRIPTION**

11599 None.

**11600 EXIT STATUS**

11601 The following exit values shall be returned:

11602 0 The date was written successfully.

11603 >0 An error occurred.

**11604 CONSEQUENCES OF ERRORS**

11605 Default.

## 11606 APPLICATION USAGE

11607 Conversion specifiers are of unspecified format when not in the POSIX locale. Some of them can  
 11608 contain <newline>s in some locales, so it may be difficult to use the format shown in standard  
 11609 output for parsing the output of *date* in those locales.

11610 The range of values for %S extends from 0 to 60 seconds to accommodate the occasional leap  
 11611 second.

11612 Although certain of the conversion specifiers in the POSIX locale (such as the name of the  
 11613 month) are shown with initial capital letters, this need not be the case in other locales. Programs  
 11614 using these fields may need to adjust the capitalization if the output is going to be used at the  
 11615 beginning of a sentence.

11616 The date string formatting capabilities are intended for use in Gregorian-style calendars,  
 11617 possibly with a different starting year (or years). The %x and %c conversion specifications,  
 11618 however, are intended for local representation; these may be based on a different, non-Gregorian  
 11619 calendar.

11620 The %C conversion specification was introduced to allow a fallback for the %EC (alternative year  
 11621 format base year); it can be viewed as the base of the current subdivision in the Gregorian  
 11622 calendar. The century number is calculated as the year divided by 100 and truncated to an  
 11623 integer; it should not be confused with the use of ordinal numbers for centuries (for example,  
 11624 "twenty-first century"). Both the %Ey and %Y can then be viewed as the offset from %EC and %C,  
 11625 respectively.

11626 The E and O modifiers modify the traditional conversion specifiers, so that they can always be  
 11627 used, even if the implementation (or the current locale) does not support the modifier.

11628 The E modifier supports alternative date formats, such as the Japanese Emperor's Era, as long as  
 11629 these are based on the Gregorian calendar system. Extending the E modifiers to other date  
 11630 elements may provide an implementation-defined extension capable of supporting other  
 11631 calendar systems, especially in combination with the O modifier.

11632 The O modifier supports time and date formats using the locale's alternative numerical symbols,  
 11633 such as Kanji or Hindi digits or ordinal number representation.

11634 Non-European locales, whether they use Latin digits in computational items or not, often have  
 11635 local forms of the digits for use in date formats. This is not totally unknown even in Europe; a  
 11636 variant of dates uses Roman numerals for the months: the third day of September 1991 would be  
 11637 written as 3.IX.1991. In Japan, Kanji digits are regularly used for dates; in Arabic-speaking  
 11638 countries, Hindi digits are used. The %d, %e, %H, %I, %m, %S, %U, %w, %W, and %Y conversion  
 11639 specifications always return the date and time field in Latin digits (that is, 0 to 9). The %O  
 11640 modifier was introduced to support the use for display purposes of non-Latin digits. In the  
 11641 *LC\_TIME* category in *localedef*, the optional **alt\_digits** keyword is intended for this purpose. As  
 11642 an example, assume the following (partial) *localedef* source:

```
11643 alt_digits " " ; "I" ; "II" ; "III" ; "IV" ; "V" ; "VI" ; "VII" ; "VIII" \
11644 "IX" ; "X" ; "XI" ; "XII"
11645 d_fmt "%e.%Om.%Y"
```

11646 With the above date, the command:

```
11647 date "+%x"
```

11648 would yield 3.IX.1991. With the same **d\_fmt**, but without the **alt\_digits**, the command would  
 11649 yield 3.9.1991.

## 11650 EXAMPLES

- 11651     1. The following are input/output examples of *date* used at arbitrary times in the POSIX  
 11652        locale:

```
11653 $ date
11654 Tue Jun 26 09:58:10 PDT 1990
11655 $ date "+DATE: %m/%d/%Y%nTIME: %H:%M:%S"
11656 DATE: 11/02/91
11657 TIME: 13:36:16
11658 $ date "+TIME: %r"
11659 TIME: 01:36:32 PM
```

- 11660     2. Examples for Denmark, where the default date and time format is %a %d %b %Y %T %Z:

```
11661 $ LANG=da_DK.iso_8859-1 date
11662 ons 02 okt 1991 15:03:32 CET
11663 $ LANG=da_DK.iso_8859-1 \
11664 date "+DATO: %A den %e. %B %Y%nKLOKKEN: %H:%M:%S"
11665 DATO: onsdag den 2. oktober 1991
11666 KLOKKEN: 15:03:56
```

- 11667     3. Examples for Germany, where the default date and time format is %a %d.%h.%Y, %T %Z:

```
11668 $ LANG=De_DE.88591 date
11669 Mi 02.Okt.1991, 15:01:21 MEZ
11670 $ LANG=De_DE.88591 date "+DATUM: %A, %d. %B %Y%nZEIT: %H:%M:%S"
11671 DATUM: Mittwoch, 02. Oktober 1991
11672 ZEIT: 15:02:02
```

- 11673     4. Examples for France, where the default date and time format is %a %d %h %Y %Z %T:

```
11674 $ LANG=Fr_FR.88591 date
11675 Mer 02 oct 1991 MET 15:03:32
11676 $ LANG=Fr_FR.88591 date "+JOUR: %A %d %B %Y%nHEURE: %H:%M:%S"
11677 JOUR: Mercredi 02 octobre 1991
11678 HEURE: 15:03:56
```

## 11679 RATIONALE

11680     Some of the new options for formatting are from the ISO C standard. The **-u** option was  
 11681        introduced to allow portable access to Coordinated Universal Time (UTC). The string "GMT0" is  
 11682        allowed as an equivalent *TZ* value to be compatible with all of the systems using the BSD  
 11683        implementation, where this option originated.

11684     The %e format conversion specification (adopted from System V) was added because the ISO C  
 11685        standard conversion specifications did not provide any way to produce the historical default  
 11686        *date* output during the first nine days of any month.

11687     There are two varieties of day and week numbering supported (in addition to any others created  
 11688        with the locale-dependent %E and %O modifier characters):

- 11689        • The historical variety in which Sunday is the first day of the week and the weekdays  
 11690           preceding the first Sunday of the year are considered week 0. These are represented by %w  
 11691           and %U. A variant of this is %w, using Monday as the first day of the week, but still referring  
 11692           to week 0. This view of the calendar was retained because so many historical applications  
 11693           depend on it and the ISO C standard *strftime()* function, on which many *date*

|       |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |   |
|-------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---|
| 11694 | implementations are based, was defined in this way.                                                                                                                                                                                                                                                                                                                                                                                                                            |   |
| 11695 | • The international standard, based on the ISO 8601: 2000 standard where Monday is the first weekday and the algorithm for the first week number is more complex: If the week (Monday to Sunday) containing January 1 has four or more days in the new year, then it is week 1; otherwise, it is week 53 of the previous year, and the next week is week 1. These are represented by the new conversion specifications %u and %v, added as a result of international comments. |   |
| 11701 | <b>FUTURE DIRECTIONS</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                       |   |
| 11702 | None.                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |   |
| 11703 | <b>SEE ALSO</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                |   |
| 11704 | The System Interfaces volume of IEEE Std 1003.1-2001, <i>printf()</i> , <i>strftime()</i>                                                                                                                                                                                                                                                                                                                                                                                      |   |
| 11705 | <b>CHANGE HISTORY</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                          |   |
| 11706 | First released in Issue 2.                                                                                                                                                                                                                                                                                                                                                                                                                                                     |   |
| 11707 | <b>Issue 5</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |   |
| 11708 | Changes are made for Year 2000 alignment.                                                                                                                                                                                                                                                                                                                                                                                                                                      |   |
| 11709 | <b>Issue 6</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |   |
| 11710 | The following new requirements on POSIX implementations derive from alignment with the Single UNIX Specification:                                                                                                                                                                                                                                                                                                                                                              |   |
| 11712 | • The %EX modified conversion specification is added.                                                                                                                                                                                                                                                                                                                                                                                                                          | 1 |
| 11713 | The Open Group Corrigendum U048/2 is applied, correcting the examples.                                                                                                                                                                                                                                                                                                                                                                                                         |   |
| 11714 | The DESCRIPTION is updated to refer to conversion specifications, instead of field descriptors for consistency with the <i>LC_TIME</i> category.                                                                                                                                                                                                                                                                                                                               |   |
| 11716 | A clarification is made such that the current year is the default if the yy argument is omitted when setting the system date and time.                                                                                                                                                                                                                                                                                                                                         |   |
| 11718 | IEEE Std 1003.1-2001/Cor 1-2002, item XCU/TC1/D6/19 is applied, correcting the CHANGE HISTORY section.                                                                                                                                                                                                                                                                                                                                                                         | 1 |
| 11719 |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                | 1 |

## 11720 NAME

11721 dd — convert and copy a file

## 11722 SYNOPSIS

11723 dd [*operand* . . .]

## 11724 DESCRIPTION

11725 The **dd** utility shall copy the specified input file to the specified output file with possible  
11726 conversions using specific input and output block sizes. It shall read the input one block at a  
11727 time, using the specified input block size; it shall then process the block of data actually  
11728 returned, which could be smaller than the requested block size. It shall apply any conversions  
11729 that have been specified and write the resulting data to the output in blocks of the specified  
11730 output block size. If the **bs=expr** operand is specified and no conversions other than **sync**,  
11731 **noerror**, or **notrunc** are requested, the data returned from each input block shall be written as a  
11732 separate output block; if the read returns less than a full block and the **sync** conversion is not  
11733 specified, the resulting output block shall be the same size as the input block. If the **bs=expr**  
11734 operand is not specified, or a conversion other than **sync**, **noerror**, or **notrunc** is requested, the  
11735 input shall be processed and collected into full-sized output blocks until the end of the input is  
11736 reached.

11737 The processing order shall be as follows:

- 11738 1. An input block is read.
- 11739 2. If the input block is shorter than the specified input block size and the **sync** conversion is  
11740 specified, null bytes shall be appended to the input data up to the specified size. (If either  
11741 **block** or **unblock** is also specified, <space>s shall be appended instead of null bytes.) The  
11742 remaining conversions and output shall include the pad characters as if they had been read  
11743 from the input.
- 11744 3. If the **bs=expr** operand is specified and no conversion other than **sync** or **noerror** is  
11745 requested, the resulting data shall be written to the output as a single block, and the  
11746 remaining steps are omitted.
- 11747 4. If the **swab** conversion is specified, each pair of input data bytes shall be swapped. If there  
11748 is an odd number of bytes in the input block, the last byte in the input record shall not be  
11749 swapped.
- 11750 5. Any remaining conversions (**block**, **unblock**, **lcase**, and **ucase**) shall be performed. These  
11751 conversions shall operate on the input data independently of the input blocking; an input  
11752 or output fixed-length record may span block boundaries.
- 11753 6. The data resulting from input or conversion or both shall be aggregated into output blocks  
11754 of the specified size. After the end of input is reached, any remaining output shall be  
11755 written as a block without padding if **conv=sync** is not specified; thus, the final output  
11756 block may be shorter than the output block size.

## 11757 OPTIONS

11758 None.

## 11759 OPERANDS

11760 All of the operands shall be processed before any input is read. The following operands shall be  
11761 supported:

- 11762 **if=***file* Specify the input pathname; the default is standard input.
- 11763 **of=***file* Specify the output pathname; the default is standard output. If the **seek=expr**  
11764 conversion is not also specified, the output file shall be truncated before the copy  
11765 begins if an explicit **of=***file* operand is specified, unless **conv=notrunc** is specified.

|           |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
|-----------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 11766     | If <b>seek=expr</b> is specified, but <b>conv=notrunc</b> is not, the effect of the copy shall be to preserve the blocks in the output file over which <b>dd</b> seeks, but no other portion of the output file shall be preserved. (If the size of the seek plus the size of the input file is less than the previous size of the output file, the output file shall be shortened by the copy.)                                                                                                                                                                                                                                               |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| 11771     | <b>ibs=expr</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                | Specify the input block size, in bytes, by <i>expr</i> (default is 512).                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| 11772     | <b>obs=expr</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                | Specify the output block size, in bytes, by <i>expr</i> (default is 512).                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| 11773     | <b>bs=expr</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 | Set both input and output block sizes to <i>expr</i> bytes, superseding <b>ibs=</b> and <b>obs=</b> . If no conversion other than <b>sync</b> , <b>noerror</b> , and <b>notrunc</b> is specified, each input block shall be copied to the output as a single block without aggregating short blocks.                                                                                                                                                                                                                                                                                                                                  |
| 11776     | <b>cbs=expr</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                | Specify the conversion block size for <b>block</b> and <b>unblock</b> in bytes by <i>expr</i> (default is zero). If <b>cbs=</b> is omitted or given a value of zero, using <b>block</b> or <b>unblock</b> produces unspecified results.                                                                                                                                                                                                                                                                                                                                                                                               |
| 11779 XSI | The application shall ensure that this operand is also specified if the <b>conv=</b> operand is specified with a value of <b>ascii</b> , <b>ebcdic</b> , or <b>ibm</b> . For a <b>conv=</b> operand with an <b>ascii</b> value, the input is handled as described for the <b>unblock</b> value, except that characters are converted to ASCII before any trailing <space>s are deleted. For <b>conv=</b> operands with <b>ebcdic</b> or <b>ibm</b> values, the input is handled as described for the <b>block</b> value except that the characters are converted to EBCDIC or IBM EBCDIC, respectively, after any trailing <space>s are added. |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| 11786     | <b>skip=n</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  | Skip <i>n</i> input blocks (using the specified input block size) before starting to copy. On seekable files, the implementation shall read the blocks or seek past them; on non-seekable files, the blocks shall be read and the data shall be discarded.                                                                                                                                                                                                                                                                                                                                                                            |
| 11789     | <b>seek=n</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  | Skip <i>n</i> blocks (using the specified output block size) from the beginning of the output file before copying. On non-seekable files, existing blocks shall be read and space from the current end-of-file to the specified offset, if any, filled with null bytes; on seekable files, the implementation shall seek to the specified offset or read the blocks as described for non-seekable files.                                                                                                                                                                                                                              |
| 11794     | <b>count=n</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 | Copy only <i>n</i> input blocks.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| 11795     | <b>conv=value[,value...]</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| 11796     | Where <i>values</i> are comma-separated symbols from the following list:                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| 11797 XSI | <b>ascii</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   | Convert EBCDIC to ASCII; see Table 4-6 (on page 305).                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
| 11798 XSI | <b>ebcdic</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  | Convert ASCII to EBCDIC; see Table 4-6 (on page 305).                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
| 11799 XSI | <b>ibm</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     | Convert ASCII to a different EBCDIC set; see Table 4-7 (on page 306).                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
| 11800     | The <b>ascii</b> , <b>ebcdic</b> , and <b>ibm</b> values are mutually-exclusive.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| 11801     | <b>block</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   | Treat the input as a sequence of <newline>-terminated or end-of-file-terminated variable-length records independent of the input block boundaries. Each record shall be converted to a record with a fixed length specified by the conversion block size. Any <newline> shall be removed from the input line; <space>s shall be appended to lines that are shorter than their conversion block size to fill the block. Lines that are longer than the conversion block size shall be truncated to the largest number of characters that fit into that size; the number of truncated lines shall be reported (see the STDERR section). |

|       |                |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
|-------|----------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 11811 |                | The <b>block</b> and <b>unblock</b> values are mutually-exclusive.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| 11812 | <b>unblock</b> | Convert fixed-length records to variable length. Read a number of bytes equal to the conversion block size (or the number of bytes remaining in the input, if less than the conversion block size), delete all trailing <space>s, and append a <newline>.                                                                                                                                                                                                                                                                                                                                                                              |
| 11813 |                |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| 11814 |                |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| 11815 |                |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| 11816 | <b>lcase</b>   | Map uppercase characters specified by the <i>LC_CTYPE</i> keyword <b>tolower</b> to the corresponding lowercase character. Characters for which no mapping is specified shall not be modified by this conversion.                                                                                                                                                                                                                                                                                                                                                                                                                      |
| 11817 |                |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| 11818 |                |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| 11819 |                |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| 11820 |                | The <b>lcase</b> and <b>ucase</b> symbols are mutually-exclusive.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| 11821 | <b>ucase</b>   | Map lowercase characters specified by the <i>LC_CTYPE</i> keyword <b>toupper</b> to the corresponding uppercase character. Characters for which no mapping is specified shall not be modified by this conversion.                                                                                                                                                                                                                                                                                                                                                                                                                      |
| 11822 |                |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| 11823 |                |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| 11824 |                |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| 11825 | <b>swab</b>    | Swap every pair of input bytes.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| 11826 |                |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| 11827 | <b>noerror</b> | Do not stop processing on an input error. When an input error occurs, a diagnostic message shall be written on standard error, followed by the current input and output block counts in the same format as used at completion (see the STDERR section). If the <b>sync</b> conversion is specified, the missing input shall be replaced with null bytes and processed normally; otherwise, the input block shall be omitted from the output.                                                                                                                                                                                           |
| 11828 |                |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| 11829 |                |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| 11830 |                |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| 11831 |                |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| 11832 |                |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| 11833 | <b>notrunc</b> | Do not truncate the output file. Preserve blocks in the output file not explicitly written by this invocation of the <i>dd</i> utility. (See also the preceding <b>of=</b> <i>file</i> operand.)                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| 11834 |                |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| 11835 |                |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| 11836 | <b>sync</b>    | Pad every input block to the size of the <b>ibs=</b> buffer, appending null bytes. (If either <b>block</b> or <b>unblock</b> is also specified, append <space>s, rather than null bytes.)                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| 11837 |                |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| 11838 |                |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| 11839 |                | The behavior is unspecified if operands other than <b>conv=</b> are specified more than once.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| 11840 |                | For the <b>bs=</b> , <b>cbs=</b> , <b>ibs=</b> , and <b>obs=</b> operands, the application shall supply an expression specifying a size in bytes. The expression, <i>expr</i> , can be:                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| 11841 |                |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| 11842 | 1.             | A positive decimal number                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| 11843 | 2.             | A positive decimal number followed by <i>k</i> , specifying multiplication by 1 024                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
| 11844 | 3.             | A positive decimal number followed by <i>b</i> , specifying multiplication by 512                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| 11845 | 4.             | Two or more positive decimal numbers (with or without <i>k</i> or <i>b</i> ) separated by <i>x</i> , specifying the product of the indicated values                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
| 11846 |                |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| 11847 |                | All of the operands are processed before any input is read.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
| 11848 | <b>XSI</b>     | The following two tables display the octal number character values used for the <b>ascii</b> and <b>ebcdic</b> conversions (first table) and for the <b>ibm</b> conversion (second table). 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 differences between the two tables are highlighted by small boxes drawn around five entries. |
| 11849 |                |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| 11850 |                |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| 11851 |                |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| 11852 |                |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| 11853 |                |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| 11854 |                |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |

11855  
11856

Table 4-6 ASCII to EBCDIC Conversion

|             | 0        | 1        | 2        | 3        | 4        | 5        | 6        | 7        |
|-------------|----------|----------|----------|----------|----------|----------|----------|----------|
| <b>0000</b> | 0000 NUL | 0001 SOH | 0002 STX | 0003 ETX | 0067 EOT | 0055 ENQ | 0056 ACK | 0057 BEL |
| <b>0010</b> | 0026 BS  | 0005 HT  | 0045 LF  | 0013 VT  | 0014 FF  | 0015 CR  | 0016 SO  | 0017 SI  |
| <b>0020</b> | 0020 DLE | 0021 DC1 | 0022 DC2 | 0023 DC3 | 0074 DC4 | 0075 NAK | 0062 SYN | 0046 ETB |
| <b>0030</b> | 0030 CAN | 0031 EM  | 0077 SUB | 0047 ESC | 0034 IFS | 0035 IGS | 0036 IRS | 0037 ITB |
| <b>0040</b> | 0100 Sp  | 0132 !   | 0177 "   | 0173 #   | 0133 \$  | 0154 %   | 0120 &   | 0175 '   |
| <b>0050</b> | 0115 (   | 0135 )   | 0134 *   | 0116 +   | 0153 ,   | 0140 -   | 0113 .   | 0141 /   |
| <b>0060</b> | 0360 0   | 0361 1   | 0362 2   | 0363 3   | 0364 4   | 0365 5   | 0366 6   | 0367 7   |
| <b>0070</b> | 0370 8   | 0371 9   | 0172 :   | 0136 ;   | 0114 <   | 0176 =   | 0156 >   | 0157 ?   |
| <b>0100</b> | 0174 @   | 0301 A   | 0302 B   | 0303 C   | 0304 D   | 0305 E   | 0306 F   | 0307 G   |
| <b>0110</b> | 0310 H   | 0311 I   | 0321 J   | 0322 K   | 0323 L   | 0324 M   | 0325 N   | 0326 O   |
| <b>0120</b> | 0327 P   | 0330 Q   | 0331 R   | 0342 S   | 0343 T   | 0344 U   | 0345 V   | 0346 W   |
| <b>0130</b> | 0347 X   | 0350 Y   | 0351 Z   | 0255 [   | 0340 \   | 0275 ]   | 0232     | 0155 _   |
| <b>0140</b> | 0171 `   | 0201 a   | 0202 b   | 0203 c   | 0204 d   | 0205 e   | 0206 f   | 0207 g   |
| <b>0150</b> | 0210 h   | 0211 i   | 0221 j   | 0222 k   | 0223 ]   | 0224 m   | 0225 n   | 0226 o   |
| <b>0160</b> | 0227 p   | 0230 q   | 0231 r   | 0242 s   | 0243 t   | 0244 u   | 0245 v   | 0246 w   |
| <b>0170</b> | 0247 x   | 0250 y   | 0251 z   | 0300 {   | 0117     | 0320 }   | 0137 ~   | 0007 DEL |
| <b>0200</b> | 0040 DS  | 0041 SOS | 0042 FS  | 0043 WUS | 0044 BYP | 0025 NL  | 0006 RNL | 0027 POC |
| <b>0210</b> | 0050 SA  | 0051 SFE | 0052 SM  | 0053 CSP | 0054 MFA | 0011 SPS | 0012 RPT | 0033 CU1 |
| <b>0220</b> | 0060     | 0061     | 0032 UBS | 0063 IR  | 0064 PP  | 0065 TRN | 0066 NBS | 0010 GE  |
| <b>0230</b> | 0070 SBS | 0071 IT  | 0072 RFF | 0073 CU3 | 0004 SEL | 0024 RES | 0076     | 0341     |
| <b>0240</b> | 0101     | 0102     | 0103     | 0104     | 0105     | 0106     | 0107     | 0110     |
| <b>0250</b> | 0111     | 0121     | 0122     | 0123     | 0124     | 0125     | 0126     | 0127     |
| <b>0260</b> | 0130     | 0131     | 0142     | 0143     | 0144     | 0145     | 0146     | 0147     |
| <b>0270</b> | 0150     | 0151     | 0160     | 0161     | 0162     | 0163     | 0164     | 0165     |
| <b>0300</b> | 0166     | 0167     | 0170     | 0200     | 0212     | 0213     | 0214     | 0215     |
| <b>0310</b> | 0216     | 0217     | 0220     | 0152     | 0233     | 0234     | 0235     | 0236     |
| <b>0320</b> | 0237     | 0240     | 0252     | 0253     | 0254     | 0112 ¢   | 0256     | 0257     |
| <b>0330</b> | 0260     | 0261     | 0262     | 0263     | 0264     | 0265     | 0266     | 0267     |
| <b>0340</b> | 0270     | 0271     | 0272     | 0273     | 0274     | 0241     | 0276     | 0277     |
| <b>0350</b> | 0312     | 0313     | 0314 „   | 0315     | 0316 †   | 0317     | 0332     | 0333     |
| <b>0360</b> | 0334     | 0335     | 0336     | 0337     | 0352     | 0353     | 0354 „   | 0355     |
| <b>0370</b> | 0356     | 0357     | 0372     | 0373     | 0374     | 0375     | 0376     | 0377 EO  |

Table 4-7 ASCII to IBM EBCDIC Conversion

|             | 0        | 1        | 2        | 3        | 4        | 5        | 6        | 7        |
|-------------|----------|----------|----------|----------|----------|----------|----------|----------|
| <b>0000</b> | 0000 NUL | 0001 SOH | 0002 STX | 0003 ETX | 0067 EOT | 0055 ENQ | 0056 ACK | 0057 BEL |
| <b>0010</b> | 0026 BS  | 0005 HT  | 0045 LF  | 0013 VT  | 0014 FF  | 0015 CR  | 0016 SO  | 0017 SI  |
| <b>0020</b> | 0020 DLE | 0021 DC1 | 0022 DC2 | 0023 DC3 | 0074 DC4 | 0075 NAK | 0062 SYN | 0046 ETB |
| <b>0030</b> | 0030 CAN | 0031 EM  | 0077 SUB | 0047 ESC | 0034 IFS | 0035 IGS | 0036 IRS | 0037 ITB |
| <b>0040</b> | 0100 Sp  | 0132 !   | 0177 "   | 0173 #   | 0133 \$  | 0154 %   | 0120 &   | 0175 '   |
| <b>0050</b> | 0115 (   | 0135 )   | 0134 *   | 0116 +   | 0153 ,   | 0140 -   | 0113 .   | 0141 /   |
| <b>0060</b> | 0360 0   | 0361 1   | 0362 2   | 0363 3   | 0364 4   | 0365 5   | 0366 6   | 0367 7   |
| <b>0070</b> | 0370 8   | 0371 9   | 0172 :   | 0136 ;   | 0114 <   | 0176 =   | 0156 >   | 0157 ?   |
| <b>0100</b> | 0174 @   | 0301 A   | 0302 B   | 0303 C   | 0304 D   | 0305 E   | 0306 F   | 0307 G   |
| <b>0110</b> | 0310 H   | 0311 I   | 0321 J   | 0322 K   | 0323 L   | 0324 M   | 0325 N   | 0326 O   |
| <b>0120</b> | 0327 P   | 0330 Q   | 0331 R   | 0342 S   | 0343 T   | 0344 U   | 0345 V   | 0346 W   |
| <b>0130</b> | 0347 X   | 0350 Y   | 0351 Z   | 0255 [   | 0340 \   | 0275 ]   | 0137 ~   | 0155 _   |
| <b>0140</b> | 0171 `   | 0201 a   | 0202 b   | 0203 c   | 0204 d   | 0205 e   | 0206 f   | 0207 g   |
| <b>0150</b> | 0210 h   | 0211 i   | 0221 j   | 0222 k   | 0223 ]   | 0224 m   | 0225 n   | 0226 o   |
| <b>0160</b> | 0227 p   | 0230 q   | 0231 r   | 0242 s   | 0243 t   | 0244 u   | 0245 v   | 0246 w   |
| <b>0170</b> | 0247 x   | 0250 y   | 0251 z   | 0300 {   | 0117     | 0320 }   | 0241     | 0007 DEL |
| <b>0200</b> | 0040 DS  | 0041 SOS | 0042 FS  | 0043 WUS | 0044 BYP | 0025 NL  | 0006 RNL | 0027 POC |
| <b>0210</b> | 0050 SA  | 0051 SFE | 0052 SM  | 0053 CSP | 0054 MFA | 0011 SPS | 0012 RPT | 0033 CU1 |
| <b>0220</b> | 0060     | 0061     | 0032 UBS | 0063 IR  | 0064 PP  | 0065 TRN | 0066 NBS | 0010 GE  |
| <b>0230</b> | 0070 SBS | 0071 IT  | 0072 RFF | 0073 CU3 | 0004 SEL | 0024 RES | 0076     | 0341     |
| <b>0240</b> | 0101     | 0102     | 0103     | 0104     | 0105     | 0106     | 0107     | 0110     |
| <b>0250</b> | 0111     | 0121     | 0122     | 0123     | 0124     | 0125     | 0126     | 0127     |
| <b>0260</b> | 0130     | 0131     | 0142     | 0143     | 0144     | 0145     | 0146     | 0147     |
| <b>0270</b> | 0150     | 0151     | 0160     | 0161     | 0162     | 0163     | 0164     | 0165     |
| <b>0300</b> | 0166     | 0167     | 0170     | 0200     | 0212     | 0213     | 0214     | 0215     |
| <b>0310</b> | 0216     | 0217     | 0220     | 0232     | 0233     | 0234     | 0235     | 0236     |
| <b>0320</b> | 0237     | 0240     | 0252     | 0253     | 0254     | 0255 [   | 0256     | 0257     |
| <b>0330</b> | 0260     | 0261     | 0262     | 0263     | 0264     | 0265     | 0266     | 0267     |
| <b>0340</b> | 0270     | 0271     | 0272     | 0273     | 0274     | 0275 ]   | 0276     | 0277     |
| <b>0350</b> | 0312     | 0313     | 0314 ↴   | 0315     | 0316 ↵   | 0317     | 0332     | 0333     |
| <b>0360</b> | 0334     | 0335     | 0336     | 0337     | 0352     | 0353     | 0354 ↪   | 0355     |
| <b>0370</b> | 0356     | 0357     | 0372     | 0373     | 0374     | 0375     | 0376     | 0377 EO  |

**11859 STDIN**

11860 If no **if=** operand is specified, the standard input shall be used. See the INPUT FILES section.

**11861 INPUT FILES**

11862 The input file can be any file type.

**11863 ENVIRONMENT VARIABLES**

11864 The following environment variables shall affect the execution of *dd*:

11865 **LANG** Provide a default value for the internationalization variables that are unset or null.  
11866 (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2,  
11867 Internationalization Variables for the precedence of internationalization variables  
11868 used to determine the values of locale categories.)

11869 **LC\_ALL** If set to a non-empty string value, override the values of all the other  
11870 internationalization variables.

11871 **LC\_CTYPE** Determine the locale for the interpretation of sequences of bytes of text data as  
11872 characters (for example, single-byte as opposed to multi-byte characters in  
11873 arguments and input files), the classification of characters as uppercase or  
11874 lowercase, and the mapping of characters from one case to the other.

**11875 *LC\_MESSAGES***

11876 Determine the locale that should be used to affect the format and contents of  
11877 diagnostic messages written to standard error and informative messages written to  
11878 standard output.

11879 XSI **NLSPATH** Determine the location of message catalogs for the processing of *LC\_MESSAGES*.

**11880 ASYNCHRONOUS EVENTS**

11881 For SIGINT, the *dd* utility shall interrupt its current processing, write status information to  
11882 standard error, and exit as though terminated by SIGINT. It shall take the standard action for all  
11883 other signals; see the ASYNCHRONOUS EVENTS section in Section 1.11 (on page 20).

**11884 STDOUT**

11885 If no **of=** operand is specified, the standard output shall be used. The nature of the output  
11886 depends on the operands selected.

**11887 STDERR**

11888 On completion, *dd* shall write the number of input and output blocks to standard error. In the  
11889 POSIX locale the following formats shall be used:

11890 "%u+%u records in\n", <number of whole input blocks>,  
11891 <number of partial input blocks>

11892 "%u+%u records out\n", <number of whole output blocks>,  
11893 <number of partial output blocks>

11894 A partial input block is one for which *read()* returned less than the input block size. A partial  
11895 output block is one that was written with fewer bytes than specified by the output block size.

11896 In addition, when there is at least one truncated block, the number of truncated blocks shall be  
11897 written to standard error. In the POSIX locale, the format shall be:

11898 "%u truncated %s\n", <number of truncated blocks>, "record" (if  
11899 <number of truncated blocks> is one) "records" (otherwise)

11900 Diagnostic messages may also be written to standard error.

**11901 OUTPUT FILES**

11902 If the **of=** operand is used, the output shall be the same as described in the STDOUT section.

**11903 EXTENDED DESCRIPTION**

11904 None.

**11905 EXIT STATUS**

11906 The following exit values shall be returned:

11907 0 The input file was copied successfully.

11908 >0 An error occurred.

**11909 CONSEQUENCES OF ERRORS**

11910 If an input error is detected and the **noerror** conversion has not been specified, any partial  
11911 output block shall be written to the output file, a diagnostic message shall be written, and the  
11912 copy operation shall be discontinued. If some other error is detected, a diagnostic message shall  
11913 be written and the copy operation shall be discontinued.

**11914 APPLICATION USAGE**

11915 The input and output block size can be specified to take advantage of raw physical I/O.

11916 There are many different versions of the EBCDIC codesets. The ASCII and EBCDIC conversions  
11917 specified for the *dd* utility perform conversions for the version specified by the tables.

**11918 EXAMPLES**

11919 The following command:

11920 `dd if=/dev/rmt0h of=/dev/rmt1h`

11921 copies from tape drive 0 to tape drive 1, using a common historical device naming convention.

11922 The following command:

11923 `dd ibs=10 skip=1`

11924 strips the first 10 bytes from standard input.

11925 This example reads an EBCDIC tape blocked ten 80-byte EBCDIC card images per block into the  
11926 ASCII file **x**:

11927 `dd if=/dev/tape of=x ibs=800 cbs=80 conv=ascii,lcase`

**11928 RATIONALE**

11929 The OPTIONS section is listed as “None” because there are no options recognized by historical  
11930 *dd* utilities. Certainly, many of the operands could have been designed to use the Utility Syntax  
11931 Guidelines, which would have resulted in the classic hyphenated option letters. In this version  
11932 of this volume of IEEE Std 1003.1-2001, *dd* retains its curious JCL-like syntax due to the large  
11933 number of applications that depend on the historical implementation.

11934 A suggested implementation technique for **conv=noerror,sync** is to zero (or <space>-fill, if  
11935 **blocking** or **unblocking**) the input buffer before each read and to write the contents of the input  
11936 buffer to the output even after an error. In this manner, any data transferred to the input buffer  
11937 before the error was detected is preserved. Another point is that a failed read on a regular file or  
11938 a disk generally does not increment the file offset, and *dd* must then seek past the block on which  
11939 the error occurred; otherwise, the input error occurs repetitively. When the input is a magnetic  
11940 tape, however, the tape normally has passed the block containing the error when the error is  
11941 reported, and thus no seek is necessary.

11942 The default **ibs=** and **obs=** sizes are specified as 512 bytes because there are historical (largely  
11943 portable) scripts that assume these values. If they were left unspecified, unusual results could

11944 occur if an implementation chose an odd block size.

11945 Historical implementations of *dd* used *creat()* when processing **of=***file*. This makes the **seek=**  
11946 operand unusable except on special files. The **conv=notrunc** feature was added because more  
11947 recent BSD-based implementations use *open()* (without O\_TRUNC) instead of *creat()*, but they  
11948 fail to delete output file contents after the data copied.

11949 The **w** multiplier (historically meaning *word*), is used in System V to mean 2 and in 4.2 BSD to  
11950 mean 4. Since *word* is inherently non-portable, its use is not supported by this volume of  
11951 IEEE Std 1003.1-2001.

11952 Standard EBCDIC does not have the characters '[' and ']'. The values used in the table are  
11953 taken from a common print train that does contain them. Other than those characters, the print  
11954 train values are not filled in, but appear to provide some of the motivation for the historical  
11955 choice of translations reflected here.

11956 The Standard EBCDIC table provides a 1:1 translation for all 256 bytes.

11957 The IBM EBCDIC table does not provide such a translation. The marked cells in the tables differ  
11958 in such a way that:

- 11959 1. EBCDIC 0112 ('¤') and 0152 (broken pipe) do not appear in the table.
- 11960 2. EBCDIC 0137 ('¬') translates to/from ASCII 0236 ('^'). In the standard table, EBCDIC  
11961 0232 (no graphic) is used.
- 11962 3. EBCDIC 0241 ('~') translates to/from ASCII 0176 ('~'). In the standard table, EBCDIC  
11963 0137 ('¬') is used.
- 11964 4. 0255 ('[') and 0275 ('])' appear twice, once in the same place as for the standard table  
11965 and once in place of 0112 ('¤') and 0241 ('~').

11966 In net result:

11967   EBCDIC 0275 (']') displaced EBCDIC 0241 ('~') in cell 0345.

11968   That displaced EBCDIC 0137 ('¬') in cell 0176.

11969   That displaced EBCDIC 0232 (no graphic) in cell 0136.

11970   That replaced EBCDIC 0152 (broken pipe) in cell 0313.

11971   EBCDIC 0255 ('[') replaced EBCDIC 0112 ('¤').

11972 This translation, however, reflects historical practice that (ASCII) '~' and '¬' were often  
11973 mapped to each other, as were '[' and '¤'; and ']' and (EBCDIC) '~'.

11974 The **cbs** operand is required if any of the **ascii**, **ebcdic**, or **ibm** operands are specified. For the  
11975 **ascii** operand, the input is handled as described for the **unblock** operand except that characters  
11976 are converted to ASCII before the trailing <space>s are deleted. For the **ebcdic** and **ibm**  
11977 operands, the input is handled as described for the **block** operand except that the characters are  
11978 converted to EBCDIC or IBM EBCDIC after the trailing <space>s are added.

11979 The **block** and **unblock** keywords are from historical BSD practice.

11980 The consistent use of the word **record** in standard error messages matches most historical  
11981 practice. An earlier version of System V used **block**, but this has been updated in more recent  
11982 releases.

11983 Early proposals only allowed two numbers separated by **x** to be used in a product when  
11984 specifying **bs=**, **cbs=**, **ibs=**, and **obs=** sizes. This was changed to reflect the historical practice of  
11985 allowing multiple numbers in the product as provided by Version 7 and all releases of System V

- 11986 and BSD.
- 11987 A change to the **swab** conversion is required to match historical practice and is the result of IEEE  
11988 PASC Interpretations 1003.2 #03 and #04, submitted for the ISO POSIX-2:1993 standard.
- 11989 A change to the handling of SIGINT is required to match historical practice and is the result of  
11990 IEEE PASC Interpretation 1003.2 #06 submitted for the ISO POSIX-2:1993 standard.
- 11991 **FUTURE DIRECTIONS**
- 11992 None.
- 11993 **SEE ALSO**
- 11994 Section 1.11 (on page 20), *sed*, *tr*
- 11995 **CHANGE HISTORY**
- 11996 First released in Issue 2.
- 11997 **Issue 5**
- 11998 The second paragraph of the **cbs=** description is reworded and marked EX.
- 11999 The FUTURE DIRECTIONS section is added.
- 12000 **Issue 6**
- 12001 Changes are made to **swab** conversion and SIGINT handling to align with the IEEE P1003.2b  
12002 draft standard.
- 12003 The normative text is reworded to avoid use of the term “must” for application requirements.
- 12004 IEEE PASC Interpretation 1003.2 #209 is applied, clarifying the interaction between **dd of=file** and  
12005 **conv=notrunc**.

## 12006 NAME

12007 delta — make a delta (change) to an SCCS file (**DEVELOPMENT**)

## 12008 SYNOPSIS

12009 XSI delta [-nps][-g list][-m mrlist][-r SID][-y[comment]] file...

12010

## 12011 DESCRIPTION

12012 The *delta* utility shall be used to permanently introduce into the named SCCS files changes that  
12013 were made to the files retrieved by *get* (called the *g-files*, or generated files).

## 12014 OPTIONS

12015 The *delta* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section  
12016 12.2, Utility Syntax Guidelines, except that the **-y** option has an optional option-argument. This  
12017 optional option-argument shall not be presented as a separate argument.

12018 The following options shall be supported:

- 12019 **-r SID** Uniquely identify which delta is to be made to the SCCS file. The use of this option  
12020 shall be necessary only if two or more outstanding *get* commands for editing (*get*  
12021 **-e**) on the same SCCS file were done by the same person (login name). The SID  
12022 value specified with the **-r** option can be either the SID specified on the *get*  
12023 command line or the SID to be made as reported by the *get* utility; see *get* (on page  
12024 476).
- 12025 **-s** Suppress the report to standard output of the activity associated with each *file*.  
12026 See the STDOUT section.
- 12027 **-n** Specify retention of the edited *g-file* (normally removed at completion of delta  
12028 processing).
- 12029 **-g list** Specify a *list* (see *get* for the definition of *list*) of deltas that shall be ignored when  
12030 the file is accessed at the change level (SID) created by this delta.
- 12031 **-m mrlist** Specify a modification request (MR) number that the application shall supply as  
12032 the reason for creating the new delta. This shall be used if the SCCS file has the **v**  
12033 flag set; see *admin*.
- 12034 If **-m** is not used and '**-**' is not specified as a file argument, and the standard  
12035 input is a terminal, the prompt described in the STDOUT section shall be written  
12036 to standard output before the standard input is read; if the standard input is not a  
12037 terminal, no prompt shall be issued.
- 12038 MRs in a list shall be separated by <blank>s or escaped <newline>s. An  
12039 unescaped <newline> shall terminate the MR list. The escape character is  
12040 <backslash>.
- 12041 If the **v** flag has a value, it shall be taken to be the name of a program which  
12042 validates the correctness of the MR numbers. If a non-zero exit status is returned  
12043 from the MR number validation program, the *delta* utility shall terminate. (It is  
12044 assumed that the MR numbers were not all valid.)
- 12045 **-y[comment]** Describe the reason for making the delta. The *comment* shall be an arbitrary group  
12046 of lines that would meet the definition of a text file. Implementations shall support  
12047 *comments* from zero to 512 bytes and may support longer values. A null string  
12048 (specified as either **-y**, **-y" "**, or in response to a prompt for a comment) shall be  
12049 considered a valid *comment*.

12050            If **-y** is not specified and '**-**' is not specified as a file argument, and the standard  
 12051            input is a terminal, the prompt described in the STDOUT section shall be written  
 12052            to standard output before the standard input is read; if the standard input is not a  
 12053            terminal, no prompt shall be issued. An unescaped <newline> shall terminate the  
 12054            comment text. The escape character is <backslash>.

12055            The **-y** option shall be required if the *file* operand is specified as '**-**'.

12056            **-p**            Write (to standard output) the SCCS file differences before and after the delta is  
 12057            applied in *diff* format; see *diff*.

## 12058 OPERANDS

12059            The following operand shall be supported:

12060            *file*          A pathname of an existing SCCS file or a directory. If *file* is a directory, the *delta*  
 12061            utility shall behave as though each file in the directory were specified as a named  
 12062            file, except that non-SCCS files (last component of the pathname does not begin  
 12063            with s.) and unreadable files shall be silently ignored.

12064            If exactly one *file* operand appears, and it is '**-**', the standard input shall be read;  
 12065            each line of the standard input shall be taken to be the name of an SCCS file to be  
 12066            processed. Non-SCCS files and unreadable files shall be silently ignored.

## 12067 STDIN

12068            The standard input shall be a text file used only in the following cases:

- 12069            • To read an *mrlist* or a *comment* (see the **-m** and **-y** options).
- 12070            • A *file* operand shall be specified as '**-**'. In this case, the **-y** option must be used to specify  
 12071            the *comment*, and if the SCCS file has the **v** flag set, the **-m** option must also be used to  
 12072            specify the MR list.

## 12073 INPUT FILES

12074            Input files shall be text files whose data is to be included in the SCCS files. If the first character of  
 12075            any line of an input file is <SOH> in the POSIX locale, the results are unspecified. If this file  
 12076            contains more than 99 999 lines, the number of lines recorded in the header for this file shall be  
 12077            99 999 for this delta.

## 12078 ENVIRONMENT VARIABLES

12079            The following environment variables shall affect the execution of *delta*:

12080            **LANG**        Provide a default value for the internationalization variables that are unset or null.  
 12081            (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2,  
 12082            Internationalization Variables for the precedence of internationalization variables  
 12083            used to determine the values of locale categories.)

12084            **LC\_ALL**      If set to a non-empty string value, override the values of all the other  
 12085            internationalization variables.

12086            **LC\_CTYPE**     Determine the locale for the interpretation of sequences of bytes of text data as  
 12087            characters (for example, single-byte as opposed to multi-byte characters in  
 12088            arguments and input files).

## 12089 **LC\_MESSAGES**

12090            Determine the locale that should be used to affect the format and contents of  
 12091            diagnostic messages written to standard error, and informative messages written  
 12092            to standard output.

12093            **NLSPATH**     Determine the location of message catalogs for the processing of **LC\_MESSAGES**.

12094        **TZ**        Determine the timezone in which the time and date are written in the SCCS file. If  
12095                  the *TZ* variable is unset or NULL, an unspecified system default timezone is used.

## 12096 ASYNCHRONOUS EVENTS

12097        If SIGINT is caught, temporary files shall be cleaned up and *delta* shall exit with a non-zero exit  
12098                  code. The standard action shall be taken for all other signals; see Section 1.11 (on page 20).

## 12099 STDOUT

12100        The standard output shall be used only for the following messages in the POSIX locale:

- Prompts (see the **-m** and **-y** options) in the following formats:

12102        "MRs? "

12103        "comments? "

12104        The MR prompt, if written, shall always precede the comments prompt.

- A report of each file's activities (unless the **-s** option is specified) in the following format:

12106        "%s\n%d inserted\n%d deleted\n%d unchanged\n", <New SID>,

12107                  <number of lines inserted>, <number of lines deleted>,

12108                  <number of lines unchanged>

## 12109 STDERR

12110        The standard error shall be used only for diagnostic messages.

## 12111 OUTPUT FILES

12112        Any SCCS files updated shall be files of an unspecified format.

## 12113 EXTENDED DESCRIPTION

### 12114 System Date and Time

12115        When a delta is added to an SCCS file, the system date and time shall be recorded for the new  
12116                  delta. If a *get* is performed using an SCCS file with a date recorded apparently in the future, the  
12117                  behavior is unspecified.

### 12118 EXIT STATUS

12119        The following exit values shall be returned:

12120        0 Successful completion.

12121        >0 An error occurred.

## 12122 CONSEQUENCES OF ERRORS

12123        Default.

## 12124 APPLICATION USAGE

12125        Problems can arise if the system date and time have been modified (for example, put forward  
12126                  and then back again, or unsynchronized clocks across a network) and can also arise when  
12127                  different values of the *TZ* environment variable are used.

12128        Problems of a similar nature can also arise for the operation of the *get* utility, which records the  
12129                  date and time in the file body.

## 12130 EXAMPLES

12131        None.

**12132 RATIONALE**

12133 None.

**12134 FUTURE DIRECTIONS**

12135 None.

**12136 SEE ALSO**

12137 Section 1.11 (on page 20), *admin*, *diff*, *get*, *prs*, *rmdel*

**12138 CHANGE HISTORY**

12139 First released in Issue 2.

**12140 Issue 5**

12141 The output format description in the STDOUT section is corrected.

**12142 Issue 6**

12143 The APPLICATION USAGE section is added.

12144 The normative text is reworded to avoid use of the term “must” for application requirements.

12145 The Open Group Base Resolution bwg2001-007 is applied as follows:

12146 • The use of ‘–’ as a file argument is clarified.

12147 • The use of STDIN is added.

12148 • The ASYNCHRONOUS EVENTS section is updated to remove the implicit requirement that  
12149 implementations re-signal themselves when catching a normally fatal signal.

12150 • New text is added to the INPUT FILES section warning that the maximum lines recorded in  
12151 the file is 99 999.

12152 New text is added to the EXTENDED DESCRIPTION and APPLICATION USAGE sections  
12153 regarding how the system date and time may be taken into account, and the TZ environment  
12154 variable is added to the ENVIRONMENT VARIABLES section as per The Open Group Base  
12155 Resolution bwg2001-007.

**12156 NAME**

12157 df — report free disk space

**12158 SYNOPSIS**

12159 UP XSI df [-k] [-P | -t] [file...]

12160

**12161 DESCRIPTION**

12162 XSI The *df* utility shall write the amount of available space and file slots for file systems on which the invoking user has appropriate read access. File systems shall be specified by the *file* operands; when none are specified, information shall be written for all file systems. The format of the default output from *df* is unspecified, but all space figures are reported in 512-byte units, unless the **-k** option is specified. This output shall contain at least the file system names, amount of available space on each of these file systems, and the number of free file slots, or *inodes*, available; when **-t** is specified, the output shall contain the total allocated space as well.

**12169 OPTIONS**

12170 The *df* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section 12.2, Utility Syntax Guidelines.

12172 The following options shall be supported:

12173 **-k** Use 1024-byte units, instead of the default 512-byte units, when writing space figures.

12175 **-P** Produce output in the format described in the STDOUT section.

12176 XSI **-t** Include total allocated-space figures in the output.

**12177 OPERANDS**

12178 The following operand shall be supported:

12179 *file* A pathname of a file within the hierarchy of the desired file system. If a file other than a FIFO, a regular file, a directory, or a special file representing the device containing the file system (for example, */dev/dsk/0s1*) is specified, the results are unspecified. Otherwise, *df* shall write the amount of free space in the file system containing the specified *file* operand.

**12184 STDIN**

12185 Not used.

**12186 INPUT FILES**

12187 None.

**12188 ENVIRONMENT VARIABLES**

12189 The following environment variables shall affect the execution of *df*:

12190 *LANG* Provide a default value for the internationalization variables that are unset or null. (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2, Internationalization Variables for the precedence of internationalization variables used to determine the values of locale categories.)

12194 *LC\_ALL* If set to a non-empty string value, override the values of all the other internationalization variables.

12196 *LC\_CTYPE* Determine the locale for the interpretation of sequences of bytes of text data as characters (for example, single-byte as opposed to multi-byte characters in arguments).

12199           ***LC\_MESSAGES***  
 12200           Determine the locale that should be used to affect the format and contents of  
 12201           diagnostic messages written to standard error and informative messages written to  
 12202           standard output.

12203 XSI       ***NLSPATH***   Determine the location of message catalogs for the processing of *LC\_MESSAGES*.

12204 **ASYNCHRONOUS EVENTS**

12205           Default.

12206 **STDOUT**

12207           When both the **-k** and **-P** options are specified, the following header line shall be written (in the  
 12208           POSIX locale):

12209           "Filesystem 1024-blocks Used Available Capacity Mounted on\n"

12210           When the **-P** option is specified without the **-k** option, the following header line shall be written  
 12211           (in the POSIX locale):

12212           "Filesystem 512-blocks Used Available Capacity Mounted on\n"

12213           The implementation may adjust the spacing of the header line and the individual data lines so  
 12214           that the information is presented in orderly columns.

12215           The remaining output with **-P** shall consist of one line of information for each specified file  
 12216           system. These lines shall be formatted as follows:

12217           "%s %d %d %d%% %s\n", <file system name>, <total space>,  
 12218           <space used>, <space free>, <percentage used>,  
 12219           <file system root>

12220           In the following list, all quantities expressed in 512-byte units (1 024-byte when **-k** is specified)  
 12221           shall be rounded up to the next higher unit. The fields are:

12222           <*file system name*>  
 12223           The name of the file system, in an implementation-defined format.

12224           <*total space*> The total size of the file system in 512-byte units. The exact meaning of this figure  
 12225           is implementation-defined, but should include <*space used*>, <*space free*>, plus any  
 12226           space reserved by the system not normally available to a user.

12227           <*space used*> The total amount of space allocated to existing files in the file system, in 512-byte  
 12228           units.

12229           <*space free*> The total amount of space available within the file system for the creation of new  
 12230           files by unprivileged users, in 512-byte units. When this figure is less than or equal  
 12231           to zero, it shall not be possible to create any new files on the file system without  
 12232           first deleting others, unless the process has appropriate privileges. The figure  
 12233           written may be less than zero.

12234           <*percentage used*>  
 12235           The percentage of the normally available space that is currently allocated to all  
 12236           files on the file system. This shall be calculated using the fraction:

12237           <*space used*> / ( <*space used*> + <*space free*> )

12238           expressed as a percentage. This percentage may be greater than 100 if <*space free*>  
 12239           is less than zero. The percentage value shall be expressed as a positive integer,  
 12240           with any fractional result causing it to be rounded to the next highest integer.

- 12241        <file system root>  
12242              The directory below which the file system hierarchy appears.
- 12243 XSI        The output format is unspecified when **-t** is used.
- 12244 **STDERR**  
12245              The standard error shall be used only for diagnostic messages.
- 12246 **OUTPUT FILES**  
12247              None.
- 12248 **EXTENDED DESCRIPTION**  
12249              None.
- 12250 **EXIT STATUS**  
12251              The following exit values shall be returned:  
12252                  0    Successful completion.  
12253                  >0   An error occurred.
- 12254 **CONSEQUENCES OF ERRORS**  
12255              Default.
- 12256 **APPLICATION USAGE**  
12257              On most systems, the “name of the file system, in an implementation-defined format” is the  
12258              special file on which the file system is mounted.  
12259              On large file systems, the calculation specified for percentage used can create huge rounding  
12260              errors.
- 12261 **EXAMPLES**  
12262              1. The following example writes portable information about the **/usr** file system:  
12263                      df -P /usr  
12264              2. Assuming that **/usr/src** is part of the **/usr** file system, the following produces the same  
12265              output as the previous example:  
12266                      df -P /usr/src
- 12267 **RATIONALE**  
12268              The behavior of *df* with the **-P** option is the default action of the 4.2 BSD *df* utility. The uppercase  
12269              **-P** was selected to avoid collision with a known industry extension using **-p**.  
12270              Historical *df* implementations vary considerably in their default output. It was therefore  
12271              necessary to describe the default output in a loose manner to accommodate all known historical  
12272              implementations and to add a portable option (**-P**) to provide information in a portable format.  
12273              The use of 512-byte units is historical practice and maintains compatibility with *ls* and other  
12274              utilities in this volume of IEEE Std 1003.1-2001. This does not mandate that the file system itself  
12275              be based on 512-byte blocks. The **-k** option was added as a compromise measure. It was agreed  
12276              by the standard developers that 512 bytes was the best default unit because of its complete  
12277              historical consistency on System V (versus the mixed 512/1 024-byte usage on BSD systems), and  
12278              that a **-k** option to switch to 1 024-byte units was a good compromise. Users who prefer the  
12279              more logical 1 024-byte quantity can easily alias *df* to *df -k* without breaking many historical  
12280              scripts relying on the 512-byte units.  
12281              It was suggested that *df* and the various related utilities be modified to access a **BLOCKSIZE**  
12282              environment variable to achieve consistency and user acceptance. Since this is not historical  
12283              practice on any system, it is left as a possible area for system extensions and will be re-evaluated

12284       in a future version if it is widely implemented.

12285 **FUTURE DIRECTIONS**

12286       None.

12287 **SEE ALSO**

12288       *find*

12289 **CHANGE HISTORY**

12290       First released in Issue 2.

12291 **Issue 6**

12292       This utility is marked as part of the User Portability Utilities option.

12293 **NAME**

12294       **diff** — compare two files

12295 **SYNOPSIS**

12296       **diff** [**-c** | **-e** | **-f** | **-C** *n*] [**-br**] *file1* *file2*

12297 **DESCRIPTION**

12298       The *diff* utility shall compare the contents of *file1* and *file2* and write to standard output a list of  
12299       changes necessary to convert *file1* into *file2*. This list should be minimal. No output shall be  
12300       produced if the files are identical.

12301 **OPTIONS**

12302       The *diff* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section  
12303       12.2, Utility Syntax Guidelines.

12304       The following options shall be supported:

- 12305       **-b**       Cause any amount of white space at the end of a line to be treated as a single  
12306                  <newline> (that is, the white-space characters preceding the <newline> are  
12307                  ignored) and other strings of white-space characters, not including <newline>s, to  
12308                  compare equal.
- 12309       **-c**       Produce output in a form that provides three lines of context.
- 12310       **-C** *n*    Produce output in a form that provides *n* lines of context (where *n* shall be  
12311                  interpreted as a positive decimal integer).
- 12312       **-e**       Produce output in a form suitable as input for the *ed* utility, which can then be  
12313                  used to convert *file1* into *file2*.
- 12314       **-f**       Produce output in an alternative form, similar in format to **-e**, but not intended to  
12315                  be suitable as input for the *ed* utility, and in the opposite order.
- 12316       **-r**       Apply *diff* recursively to files and directories of the same name when *file1* and *file2*  
12317                  are both directories.

12318 **OPERANDS**

12319       The following operands shall be supported:

12320       *file1*, *file2*   A pathname of a file to be compared. If either the *file1* or *file2* operand is '**-**', the  
12321                  standard input shall be used in its place.

12322       If both *file1* and *file2* are directories, *diff* shall not compare block special files, character special  
12323                  files, or FIFO special files to any files and shall not compare regular files to directories. Further  
12324                  details are as specified in **Diff Directory Comparison Format** (on page 320). The behavior of *diff*  
12325                  on other file types is implementation-defined when found in directories.

12326       If only one of *file1* and *file2* is a directory, *diff* shall be applied to the non-directory file and the file  
12327                  contained in the directory file with a filename that is the same as the last component of the non-  
12328                  directory file.

12329 **STDIN**

12330       The standard input shall be used only if one of the *file1* or *file2* operands references standard  
12331                  input. See the INPUT FILES section.

12332 **INPUT FILES**

12333       The input files may be of any type.

## 12334 ENVIRONMENT VARIABLES

12335 The following environment variables shall affect the execution of *diff*:

|           |                    |                                                                                                                                                                                                                                                                                                          |
|-----------|--------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 12336     | <i>LANG</i>        | Provide a default value for the internationalization variables that are unset or null.<br>(See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2, Internationalization Variables for the precedence of internationalization variables used to determine the values of locale categories.) |
| 12340     | <i>LC_ALL</i>      | If set to a non-empty string value, override the values of all the other internationalization variables.                                                                                                                                                                                                 |
| 12342     | <i>LC_CTYPE</i>    | Determine the locale for the interpretation of sequences of bytes of text data as characters (for example, single-byte as opposed to multi-byte characters in arguments and input files).                                                                                                                |
| 12345     | <i>LC_MESSAGES</i> | Determine the locale that should be used to affect the format and contents of diagnostic messages written to standard error and informative messages written to standard output.                                                                                                                         |
| 12349     | <i>LC_TIME</i>     | Determine the locale for affecting the format of file timestamps written with the <b>-C</b> and <b>-c</b> options.                                                                                                                                                                                       |
| 12351 XSI | <i>NLSPATH</i>     | Determine the location of message catalogs for the processing of <i>LC_MESSAGES</i> .                                                                                                                                                                                                                    |
| 12352     | <i>TZ</i>          | Determine the timezone used for calculating file timestamps written with the <b>-C</b> and <b>-c</b> options. If <i>TZ</i> is unset or null, an unspecified default timezone shall be used.                                                                                                              |

## 12355 ASYNCHRONOUS EVENTS

12356 Default.

## 12357 STDOUT

## 12358 Diff Directory Comparison Format

12359 If both *file1* and *file2* are directories, the following output formats shall be used.

12360 In the POSIX locale, each file that is present in only one directory shall be reported using the following format:

12362 "Only in %s: %s\n", <directory pathname>, <filename>

12363 In the POSIX locale, subdirectories that are common to the two directories may be reported with the following format:

12365 "Common subdirectories: %s and %s\n", <directory1 pathname>,  
12366 <directory2 pathname>

12367 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:

12369 "File %s is a %s while file %s is a %s\n", <directory1 pathname>,  
12370 <file type of directory1 pathname>, <directory2 pathname>,  
12371 <file type of directory2 pathname>

12372 For each file common to the two directories, if the files are compared and are identical, no output shall be written. If the two files differ, the following format is written:

12374 "diff %s %s %s\n", <diff\_options>, <filename1>, <filename2>

12375 where <diff\_options> are the options as specified on the command line.  
12376 All directory pathnames listed in this section shall be relative to the original command line  
12377 arguments. All other names of files listed in this section shall be filenames (pathname  
12378 components).

### 12379 Diff Binary Output Format

12380 In the POSIX locale, if one or both of the files being compared are not text files, an unspecified  
12381 format shall be used that contains the pathnames of two files being compared and the string  
12382 "differ".

12383 If both files being compared are text files, depending on the options specified, one of the  
12384 following formats shall be used to write the differences.

### 12385 Diff Default Output Format

12386 The default (without -e, -f, -c, or -C options) *diff* utility output shall contain lines of these  
12387 forms:

12388 "%da%d\n", <num1>, <num2>  
12389 "%da%d,%d\n", <num1>, <num2>, <num3>  
12390 "%dd%d\n", <num1>, <num2>  
12391 "%d,%dd%d\n", <num1>, <num2>, <num3>  
12392 "%dc%d\n", <num1>, <num2>  
12393 "%d,%dc%d\n", <num1>, <num2>, <num3>  
12394 "%dc%d,%d\n", <num1>, <num2>, <num3>  
12395 "%d,%dc%d,%d\n", <num1>, <num2>, <num3>, <num4>

12396 These lines resemble *ed* subcommands to convert *file1* into *file2*. The line numbers before the  
12397 action letters shall pertain to *file1*; those after shall pertain to *file2*. Thus, by exchanging a for d  
12398 and reading the line in reverse order, one can also determine how to convert *file2* into *file1*. As in  
12399 *ed*, identical pairs (where num1= num2) are abbreviated as a single number.

12400 Following each of these lines, *diff* shall write to standard output all lines affected in the first file  
12401 using the format:

12402 "<Δ%>s", <line>

12403 and all lines affected in the second file using the format:

12404 ">Δ%>s", <line>

12405 If there are lines affected in both *file1* and *file2* (as with the c subcommand), the changes are  
12406 separated with a line consisting of three hyphens:

12407 "---\n"

12408      **Diff -e Output Format**

12409      With the **-e** option, a script shall be produced that shall, when provided as input to *ed*, along  
 12410      with an appended **w** (write) command, convert *file1* into *file2*. Only the **a** (append), **c** (change), **d**  
 12411      (delete), **i** (insert), and **s** (substitute) commands of *ed* shall be used in this script. Text lines,  
 12412      except those consisting of the single character period ('.'), shall be output as they appear in the  
 12413      file.

12414      **Diff -f Output Format**

12415      With the **-f** option, an alternative format of script shall be produced. It is similar to that  
 12416      produced by **-e**, with the following differences:

- 12417      1. It is expressed in reverse sequence; the output of **-e** orders changes from the end of the file  
     12418      to the beginning; the **-f** from beginning to end.
- 12419      2. The command form *<lines> <command-letter>* used by **-e** is reversed. For example,  
     12420      10c with **-e** would be c10 with **-f**.
- 12421      3. The form used for ranges of line numbers is *<space>*-separated, rather than comma-  
     12422      separated.

12423      **Diff -c or -C Output Format**

12424      With the **-c** or **-C** option, the output format shall consist of affected lines along with  
 12425      surrounding lines of context. The affected lines shall show which ones need to be deleted or  
 12426      changed in *file1*, and those added from *file2*. With the **-c** option, three lines of context, if  
 12427      available, shall be written before and after the affected lines. With the **-C** option, the user can  
 12428      specify how many lines of context are written. The exact format follows.

12429      The name and last modification time of each file shall be output in the following format:

```
12430 " *** %s %s\n", file1, <file1 timestamp>
12431 "--- %s %s\n", file2, <file2 timestamp>
```

12432      Each *<file>* field shall be the pathname of the corresponding file being compared. The pathname  
 12433      written for standard input is unspecified.

12434      In the POSIX locale, each *<timestamp>* field shall be equivalent to the output from the following  
 12435      command:

```
12436 date "+%a %b %e %T %Y"
```

12437      without the trailing *<newline>*, executed at the time of last modification of the corresponding  
 12438      file (or the current time, if the file is standard input).

12439      Then, the following output formats shall be applied for every set of changes.

12440      First, a line shall be written in the following format:

```
12441 "*****\n"
```

12442      Next, the range of lines in *file1* shall be written in the following format if the range contains two 1
 12443      or more lines:

```
12444 " *** %d,%d ****\n", <beginning line number>, <ending line number>
```

12445      and the following format otherwise:

```
12446 " *** %d ****\n", <ending line number>
```

12447 The ending line number of an empty range shall be the number of the preceding line, or 0 if the 1  
12448 range is at the start of the file.

12449 Next, the affected lines along with lines of context (unaffected lines) shall be written. Unaffected 1  
12450 lines shall be written in the following format:

12451 " $\Delta\Delta\%s$ ", <unaffected\_line>

12452 Deleted lines shall be written as:

12453 " $-\Delta\%s$ ", <deleted\_line>

12454 Changed lines shall be written as:

12455 " $!\Delta\%s$ ", <changed\_line>

12456 Next, the range of lines in *file2* shall be written in the following format if the range contains two 1  
12457 or more lines:

12458 " $--- \%d, \%d ---\n$ ", <beginning line number>, <ending line number>

12459 and the following format otherwise: 1

12460 " $--- \%d ---\n$ ", <ending line number>

12461 Then, lines of context and changed lines shall be written as described in the previous formats.  
12462 Lines added from *file2* shall be written in the following format:

12463 " $+\Delta\%s$ ", <added\_line>

12464 **STDERR**

12465 The standard error shall be used only for diagnostic messages.

12466 **OUTPUT FILES**

12467 None.

12468 **EXTENDED DESCRIPTION**

12469 None.

12470 **EXIT STATUS**

12471 The following exit values shall be returned:

12472 0 No differences were found.

12473 1 Differences were found.

12474 >1 An error occurred.

12475 **CONSEQUENCES OF ERRORS**

12476 Default.

12477 **APPLICATION USAGE**

12478 If lines at the end of a file are changed and other lines are added, *diff* output may show this as a  
12479 delete and add, as a change, or as a change and add; *diff* is not expected to know which  
12480 happened and users should not care about the difference in output as long as it clearly shows the  
12481 differences between the files.

12482 **EXAMPLES**

12483 If **dir1** is a directory containing a directory named **x**, **dir2** is a directory containing a directory  
12484 named **x**, **dir1/x** and **dir2/x** both contain files named **date.out**, and **dir2/x** contains a file named **y**,  
12485 the command:

12486 `diff -r dir1 dir2`

12487 could produce output similar to:  
12488 Common subdirectories: dir1/x and dir2/x  
12489 Only in dir2/x: y  
12490 diff -r dir1/x/date.out dir2/x/date.out  
12491 1c1  
12492 < Mon Jul 2 13:12:16 PDT 1990  
12493 ---  
12494 > Tue Jun 19 21:41:39 PDT 1990

#### 12495 RATIONALE

12496 The **-h** option was omitted because it was insufficiently specified and does not add to  
12497 applications portability.

12498 Historical implementations employ algorithms that do not always produce a minimum list of  
12499 differences; the current language about making every effort is the best this volume of  
12500 IEEE Std 1003.1-2001 can do, as there is no metric that could be employed to judge the quality of  
12501 implementations against any and all file contents. The statement “This list should be minimal”  
12502 clearly implies that implementations are not expected to provide the following output when  
12503 comparing two 100-line files that differ in only one character on a single line:

12504 1,100c1,100  
12505 all 100 lines from file1 preceded with "< "  
12506 ---  
12507 all 100 lines from file2 preceded with "> "

12508 The “Only in” messages required when the **-r** option is specified are not used by most historical  
12509 implementations if the **-e** option is also specified. It is required here because it provides useful  
12510 information that must be provided to update a target directory hierarchy to match a source  
12511 hierarchy. The “Common subdirectories” messages are written by System V and 4.3 BSD when  
12512 the **-r** option is specified. They are allowed here but are not required because they are reporting  
12513 on something that is the same, not reporting a difference, and are not needed to update a target  
12514 hierarchy.

12515 The **-c** option, which writes output in a format using lines of context, has been included. The  
12516 format is useful for a variety of reasons, among them being much improved readability and the  
12517 ability to understand difference changes when the target file has line numbers that differ from  
12518 another similar, but slightly different, copy. The *patch* utility is most valuable when working  
12519 with difference listings using the context format. The BSD version of **-c** takes an optional  
12520 argument specifying the amount of context. Rather than overloading **-c** and breaking the Utility  
12521 Syntax Guidelines for *diff*, the standard developers decided to add a separate option for  
12522 specifying a context diff with a specified amount of context (**-C**). Also, the format for context  
12523 *diffs* was extended slightly in 4.3 BSD to allow multiple changes that are within context lines  
12524 from each other to be merged together. The output format contains an additional four asterisks  
12525 after the range of affected lines in the first filename. This was to provide a flag for old programs  
12526 (like old versions of *patch*) that only understand the old context format. The version of context  
12527 described here does not require that multiple changes within context lines be merged, but it does  
12528 not prohibit it either. The extension is upwards-compatible, so any vendors that wish to retain  
12529 the old version of *diff* can do so by adding the extra four asterisks (that is, utilities that currently  
12530 use *diff* and understand the new merged format will also understand the old unmerged format,  
12531 but not *vice versa*).

12532 The substitute command was added as an additional format for the **-e** option. This was added to  
12533 provide implementations with a way to fix the classic “dot alone on a line” bug present in many  
12534 versions of *diff*. Since many implementations have fixed this bug, the standard developers  
12535 decided not to standardize broken behavior, but rather to provide the necessary tool for fixing

12536 the bug. One way to fix this bug is to output two periods whenever a lone period is needed, then  
12537 terminate the append command with a period, and then use the substitute command to convert  
12538 the two periods into one period.

12539 The BSD-derived **-r** option was added to provide a mechanism for using *diff* to compare two file  
12540 system trees. This behavior is useful, is standard practice on all BSD-derived systems, and is not  
12541 easily reproducible with the *find* utility.

12542 The requirement that *diff* not compare files in some circumstances, even though they have the  
12543 same name, is based on the actual output of historical implementations. The message specified  
12544 here is already in use when a directory is being compared to a non-directory. It is extended here  
12545 to preclude the problems arising from running into FIFOs and other files that would cause *diff* to  
12546 hang waiting for input with no indication to the user that *diff* was hung. In most common usage,  
12547 *diff -r* should indicate differences in the file hierarchies, not the difference of contents of devices  
12548 pointed to by the hierarchies.

12549 Many early implementations of *diff* require seekable files. Since the System Interfaces volume of  
12550 IEEE Std 1003.1-2001 supports named pipes, the standard developers decided that such a  
12551 restriction was unreasonable. Note also that the allowed filename – almost always refers to a  
12552 pipe.

12553 No directory search order is specified for *diff*. The historical ordering is, in fact, not optimal, in  
12554 that it prints out all of the differences at the current level, including the statements about all  
12555 common subdirectories before recursing into those subdirectories.

12556 The message:

12557 "diff %s %s %s\n", <diff\_options>, <filename1>, <filename2>

12558 does not vary by locale because it is the representation of a command, not an English sentence.

## 12559 FUTURE DIRECTIONS

12560 None.

## 12561 SEE ALSO

12562 *cmp*, *comm*, *ed*, *find*

## 12563 CHANGE HISTORY

12564 First released in Issue 2.

## 12565 Issue 5

12566 The FUTURE DIRECTIONS section is added.

## 12567 Issue 6

12568 The following new requirements on POSIX implementations derive from alignment with the  
12569 Single UNIX Specification:

- 12570 • The **-f** option is added.

12571 The output format for **-c** or **-C** format is changed to align with changes to the IEEE P1003.2b  
12572 draft standard resulting from IEEE PASC Interpretation 1003.2 #71.

12573 The normative text is reworded to avoid use of the term “must” for application requirements.

12574 IEEE Std 1003.1-2001/Cor 1-2002, item XCU/TC1/D6/20 is applied, changing the STDOUT 1  
12575 section. This changes the specification of *diff -c* so that it agrees with existing practice when 1  
12576 contexts contain zero lines or one line. 1

12577 **NAME**

12578        dirname — return the directory portion of a pathname

12579 **SYNOPSIS**12580        dirname *string*12581 **DESCRIPTION**

12582        The *string* operand shall be treated as a pathname, as defined in the Base Definitions volume of  
12583        IEEE Std 1003.1-2001, Section 3.266, Pathname. The string *string* shall be converted to the name  
12584        of the directory containing the filename corresponding to the last pathname component in  
12585        *string*, performing actions equivalent to the following steps in order:

- 12586        1. If *string* is //, skip steps 2 to 5.
- 12587        2. If *string* consists entirely of slash characters, *string* shall be set to a single slash character. In  
12588        this case, skip steps 3 to 8.
- 12589        3. If there are any trailing slash characters in *string*, they shall be removed.
- 12590        4. If there are no slash characters remaining in *string*, *string* shall be set to a single period  
12591        character. In this case, skip steps 5 to 8.
- 12592        5. If there are any trailing non-slash characters in *string*, they shall be removed.
- 12593        6. If the remaining *string* is //, it is implementation-defined whether steps 7 and 8 are skipped  
12594        or processed.
- 12595        7. If there are any trailing slash characters in *string*, they shall be removed.
- 12596        8. If the remaining *string* is empty, *string* shall be set to a single slash character.

12597        The resulting string shall be written to standard output.

12598 **OPTIONS**

12599        None.

12600 **OPERANDS**

12601        The following operand shall be supported:

12602        *string*        A string.12603 **STDIN**

12604        Not used.

12605 **INPUT FILES**

12606        None.

12607 **ENVIRONMENT VARIABLES**12608        The following environment variables shall affect the execution of *dirname*:

- 12609        *LANG*        Provide a default value for the internationalization variables that are unset or null.  
12610                  (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2,  
12611                  Internationalization Variables for the precedence of internationalization variables  
12612                  used to determine the values of locale categories.)
- 12613        *LC\_ALL*      If set to a non-empty string value, override the values of all the other  
12614                  internationalization variables.
- 12615        *LC\_CTYPE*     Determine the locale for the interpretation of sequences of bytes of text data as  
12616                  characters (for example, single-byte as opposed to multi-byte characters in  
12617                  arguments).

| 12618                   | <b>LC_MESSAGES</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        | Determine the locale that should be used to affect the format and contents of diagnostic messages written to standard error.                                                                                                                                                                                                                             |         |                  |   |                   |         |                      |    |                         |     |                |             |                  |             |                   |             |                   |   |                     |    |                    |   |  |
|-------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---------|------------------|---|-------------------|---------|----------------------|----|-------------------------|-----|----------------|-------------|------------------|-------------|-------------------|-------------|-------------------|---|---------------------|----|--------------------|---|--|
| 12619                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |                                                                                                                                                                                                                                                                                                                                                          |         |                  |   |                   |         |                      |    |                         |     |                |             |                  |             |                   |             |                   |   |                     |    |                    |   |  |
| 12620                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |                                                                                                                                                                                                                                                                                                                                                          |         |                  |   |                   |         |                      |    |                         |     |                |             |                  |             |                   |             |                   |   |                     |    |                    |   |  |
| 12621 XSI               | <b>NLSPATH</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            | Determine the location of message catalogs for the processing of <i>LC_MESSAGES</i> .                                                                                                                                                                                                                                                                    |         |                  |   |                   |         |                      |    |                         |     |                |             |                  |             |                   |             |                   |   |                     |    |                    |   |  |
| 12622                   | <b>ASYNCHRONOUS EVENTS</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |                                                                                                                                                                                                                                                                                                                                                          |         |                  |   |                   |         |                      |    |                         |     |                |             |                  |             |                   |             |                   |   |                     |    |                    |   |  |
| 12623                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           | Default.                                                                                                                                                                                                                                                                                                                                                 |         |                  |   |                   |         |                      |    |                         |     |                |             |                  |             |                   |             |                   |   |                     |    |                    |   |  |
| 12624                   | <b>STDOUT</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |                                                                                                                                                                                                                                                                                                                                                          |         |                  |   |                   |         |                      |    |                         |     |                |             |                  |             |                   |             |                   |   |                     |    |                    |   |  |
| 12625                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           | The <i>dirname</i> utility shall write a line to the standard output in the following format:                                                                                                                                                                                                                                                            |         |                  |   |                   |         |                      |    |                         |     |                |             |                  |             |                   |             |                   |   |                     |    |                    |   |  |
| 12626                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           | "%s\n", <resulting string>                                                                                                                                                                                                                                                                                                                               |         |                  |   |                   |         |                      |    |                         |     |                |             |                  |             |                   |             |                   |   |                     |    |                    |   |  |
| 12627                   | <b>STDERR</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |                                                                                                                                                                                                                                                                                                                                                          |         |                  |   |                   |         |                      |    |                         |     |                |             |                  |             |                   |             |                   |   |                     |    |                    |   |  |
| 12628                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           | The standard error shall be used only for diagnostic messages.                                                                                                                                                                                                                                                                                           |         |                  |   |                   |         |                      |    |                         |     |                |             |                  |             |                   |             |                   |   |                     |    |                    |   |  |
| 12629                   | <b>OUTPUT FILES</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |                                                                                                                                                                                                                                                                                                                                                          |         |                  |   |                   |         |                      |    |                         |     |                |             |                  |             |                   |             |                   |   |                     |    |                    |   |  |
| 12630                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           | None.                                                                                                                                                                                                                                                                                                                                                    |         |                  |   |                   |         |                      |    |                         |     |                |             |                  |             |                   |             |                   |   |                     |    |                    |   |  |
| 12631                   | <b>EXTENDED DESCRIPTION</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |                                                                                                                                                                                                                                                                                                                                                          |         |                  |   |                   |         |                      |    |                         |     |                |             |                  |             |                   |             |                   |   |                     |    |                    |   |  |
| 12632                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           | None.                                                                                                                                                                                                                                                                                                                                                    |         |                  |   |                   |         |                      |    |                         |     |                |             |                  |             |                   |             |                   |   |                     |    |                    |   |  |
| 12633                   | <b>EXIT STATUS</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |                                                                                                                                                                                                                                                                                                                                                          |         |                  |   |                   |         |                      |    |                         |     |                |             |                  |             |                   |             |                   |   |                     |    |                    |   |  |
| 12634                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           | The following exit values shall be returned:                                                                                                                                                                                                                                                                                                             |         |                  |   |                   |         |                      |    |                         |     |                |             |                  |             |                   |             |                   |   |                     |    |                    |   |  |
| 12635                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           | 0 Successful completion.                                                                                                                                                                                                                                                                                                                                 |         |                  |   |                   |         |                      |    |                         |     |                |             |                  |             |                   |             |                   |   |                     |    |                    |   |  |
| 12636                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           | >0 An error occurred.                                                                                                                                                                                                                                                                                                                                    |         |                  |   |                   |         |                      |    |                         |     |                |             |                  |             |                   |             |                   |   |                     |    |                    |   |  |
| 12637                   | <b>CONSEQUENCES OF ERRORS</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |                                                                                                                                                                                                                                                                                                                                                          |         |                  |   |                   |         |                      |    |                         |     |                |             |                  |             |                   |             |                   |   |                     |    |                    |   |  |
| 12638                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           | Default.                                                                                                                                                                                                                                                                                                                                                 |         |                  |   |                   |         |                      |    |                         |     |                |             |                  |             |                   |             |                   |   |                     |    |                    |   |  |
| 12639                   | <b>APPLICATION USAGE</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |                                                                                                                                                                                                                                                                                                                                                          |         |                  |   |                   |         |                      |    |                         |     |                |             |                  |             |                   |             |                   |   |                     |    |                    |   |  |
| 12640                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           | The definition of <i>pathname</i> specifies implementation-defined behavior for pathnames starting with two slash characters. Therefore, applications shall not arbitrarily add slashes to the beginning of a pathname unless they can ensure that there are more or less than two or are prepared to deal with the implementation-defined consequences. |         |                  |   |                   |         |                      |    |                         |     |                |             |                  |             |                   |             |                   |   |                     |    |                    |   |  |
| 12641                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |                                                                                                                                                                                                                                                                                                                                                          |         |                  |   |                   |         |                      |    |                         |     |                |             |                  |             |                   |             |                   |   |                     |    |                    |   |  |
| 12642                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |                                                                                                                                                                                                                                                                                                                                                          |         |                  |   |                   |         |                      |    |                         |     |                |             |                  |             |                   |             |                   |   |                     |    |                    |   |  |
| 12643                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |                                                                                                                                                                                                                                                                                                                                                          |         |                  |   |                   |         |                      |    |                         |     |                |             |                  |             |                   |             |                   |   |                     |    |                    |   |  |
| 12644                   | <b>EXAMPLES</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |                                                                                                                                                                                                                                                                                                                                                          |         |                  |   |                   |         |                      |    |                         |     |                |             |                  |             |                   |             |                   |   |                     |    |                    |   |  |
| 12645                   | <table border="1"> <thead> <tr> <th>Command</th> <th>Results</th> </tr> </thead> <tbody> <tr> <td><i>dirname</i> /</td> <td>/</td></tr> <tr> <td><i>dirname</i> //</td> <td>/ or //</td></tr> <tr> <td><i>dirname</i> /a/b/</td> <td>/a</td></tr> <tr> <td><i>dirname</i> //a//b//</td> <td>//a</td></tr> <tr> <td><i>dirname</i></td> <td>Unspecified</td></tr> <tr> <td><i>dirname</i> a</td> <td>. (\$? = 0)</td></tr> <tr> <td><i>dirname</i> ""</td> <td>. (\$? = 0)</td></tr> <tr> <td><i>dirname</i> /a</td> <td>/</td></tr> <tr> <td><i>dirname</i> /a/b</td> <td>/a</td></tr> <tr> <td><i>dirname</i> a/b</td> <td>a</td></tr> </tbody> </table> | Command                                                                                                                                                                                                                                                                                                                                                  | Results | <i>dirname</i> / | / | <i>dirname</i> // | / or // | <i>dirname</i> /a/b/ | /a | <i>dirname</i> //a//b// | //a | <i>dirname</i> | Unspecified | <i>dirname</i> a | . (\$? = 0) | <i>dirname</i> "" | . (\$? = 0) | <i>dirname</i> /a | / | <i>dirname</i> /a/b | /a | <i>dirname</i> a/b | a |  |
| Command                 | Results                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |                                                                                                                                                                                                                                                                                                                                                          |         |                  |   |                   |         |                      |    |                         |     |                |             |                  |             |                   |             |                   |   |                     |    |                    |   |  |
| <i>dirname</i> /        | /                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |                                                                                                                                                                                                                                                                                                                                                          |         |                  |   |                   |         |                      |    |                         |     |                |             |                  |             |                   |             |                   |   |                     |    |                    |   |  |
| <i>dirname</i> //       | / or //                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |                                                                                                                                                                                                                                                                                                                                                          |         |                  |   |                   |         |                      |    |                         |     |                |             |                  |             |                   |             |                   |   |                     |    |                    |   |  |
| <i>dirname</i> /a/b/    | /a                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |                                                                                                                                                                                                                                                                                                                                                          |         |                  |   |                   |         |                      |    |                         |     |                |             |                  |             |                   |             |                   |   |                     |    |                    |   |  |
| <i>dirname</i> //a//b// | //a                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |                                                                                                                                                                                                                                                                                                                                                          |         |                  |   |                   |         |                      |    |                         |     |                |             |                  |             |                   |             |                   |   |                     |    |                    |   |  |
| <i>dirname</i>          | Unspecified                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |                                                                                                                                                                                                                                                                                                                                                          |         |                  |   |                   |         |                      |    |                         |     |                |             |                  |             |                   |             |                   |   |                     |    |                    |   |  |
| <i>dirname</i> a        | . (\$? = 0)                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |                                                                                                                                                                                                                                                                                                                                                          |         |                  |   |                   |         |                      |    |                         |     |                |             |                  |             |                   |             |                   |   |                     |    |                    |   |  |
| <i>dirname</i> ""       | . (\$? = 0)                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |                                                                                                                                                                                                                                                                                                                                                          |         |                  |   |                   |         |                      |    |                         |     |                |             |                  |             |                   |             |                   |   |                     |    |                    |   |  |
| <i>dirname</i> /a       | /                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |                                                                                                                                                                                                                                                                                                                                                          |         |                  |   |                   |         |                      |    |                         |     |                |             |                  |             |                   |             |                   |   |                     |    |                    |   |  |
| <i>dirname</i> /a/b     | /a                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |                                                                                                                                                                                                                                                                                                                                                          |         |                  |   |                   |         |                      |    |                         |     |                |             |                  |             |                   |             |                   |   |                     |    |                    |   |  |
| <i>dirname</i> a/b      | a                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |                                                                                                                                                                                                                                                                                                                                                          |         |                  |   |                   |         |                      |    |                         |     |                |             |                  |             |                   |             |                   |   |                     |    |                    |   |  |
| 12646                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |                                                                                                                                                                                                                                                                                                                                                          |         |                  |   |                   |         |                      |    |                         |     |                |             |                  |             |                   |             |                   |   |                     |    |                    |   |  |
| 12647                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |                                                                                                                                                                                                                                                                                                                                                          |         |                  |   |                   |         |                      |    |                         |     |                |             |                  |             |                   |             |                   |   |                     |    |                    |   |  |
| 12648                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |                                                                                                                                                                                                                                                                                                                                                          |         |                  |   |                   |         |                      |    |                         |     |                |             |                  |             |                   |             |                   |   |                     |    |                    |   |  |
| 12649                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |                                                                                                                                                                                                                                                                                                                                                          |         |                  |   |                   |         |                      |    |                         |     |                |             |                  |             |                   |             |                   |   |                     |    |                    |   |  |
| 12650                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |                                                                                                                                                                                                                                                                                                                                                          |         |                  |   |                   |         |                      |    |                         |     |                |             |                  |             |                   |             |                   |   |                     |    |                    |   |  |
| 12651                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |                                                                                                                                                                                                                                                                                                                                                          |         |                  |   |                   |         |                      |    |                         |     |                |             |                  |             |                   |             |                   |   |                     |    |                    |   |  |
| 12652                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |                                                                                                                                                                                                                                                                                                                                                          |         |                  |   |                   |         |                      |    |                         |     |                |             |                  |             |                   |             |                   |   |                     |    |                    |   |  |
| 12653                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |                                                                                                                                                                                                                                                                                                                                                          |         |                  |   |                   |         |                      |    |                         |     |                |             |                  |             |                   |             |                   |   |                     |    |                    |   |  |
| 12654                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |                                                                                                                                                                                                                                                                                                                                                          |         |                  |   |                   |         |                      |    |                         |     |                |             |                  |             |                   |             |                   |   |                     |    |                    |   |  |
| 12655                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |                                                                                                                                                                                                                                                                                                                                                          |         |                  |   |                   |         |                      |    |                         |     |                |             |                  |             |                   |             |                   |   |                     |    |                    |   |  |
| 12656                   | <b>RATIONALE</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |                                                                                                                                                                                                                                                                                                                                                          |         |                  |   |                   |         |                      |    |                         |     |                |             |                  |             |                   |             |                   |   |                     |    |                    |   |  |
| 12657                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           | The <i>dirname</i> utility originated in System III. It has evolved through the System V releases to a version that matches the requirements specified in this description in System V Release 3. 4.3                                                                                                                                                    |         |                  |   |                   |         |                      |    |                         |     |                |             |                  |             |                   |             |                   |   |                     |    |                    |   |  |
| 12658                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           | BSD and earlier versions did not include <i>dirname</i> .                                                                                                                                                                                                                                                                                                |         |                  |   |                   |         |                      |    |                         |     |                |             |                  |             |                   |             |                   |   |                     |    |                    |   |  |
| 12659                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |                                                                                                                                                                                                                                                                                                                                                          |         |                  |   |                   |         |                      |    |                         |     |                |             |                  |             |                   |             |                   |   |                     |    |                    |   |  |
| 12660                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           | The behaviors of <i>basename</i> and <i>dirname</i> in this volume of IEEE Std 1003.1-2001 have been coordinated so that when <i>string</i> is a valid pathname:                                                                                                                                                                                         |         |                  |   |                   |         |                      |    |                         |     |                |             |                  |             |                   |             |                   |   |                     |    |                    |   |  |
| 12661                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |                                                                                                                                                                                                                                                                                                                                                          |         |                  |   |                   |         |                      |    |                         |     |                |             |                  |             |                   |             |                   |   |                     |    |                    |   |  |

12662        \$(basename "string")  
12663        would be a valid filename for the file in the directory:  
12664        \$(dirname "string")  
12665        This would not work for the versions of these utilities in early proposals due to the way  
12666        processing of trailing slashes was specified. Consideration was given to leaving processing  
12667        unspecified if there were trailing slashes, but this cannot be done; the Base Definitions volume of  
12668        IEEE Std 1003.1-2001, Section 3.266, Pathname allows trailing slashes. The *basename* and *dirname*  
12669        utilities have to specify consistent handling for all valid pathnames.

**12670 FUTURE DIRECTIONS**

12671        None.

**12672 SEE ALSO**

12673        *basename*, Section 2.5 (on page 33)

**12674 CHANGE HISTORY**

12675        First released in Issue 2.

**12676 NAME**

12677 du — estimate file space usage

**12678 SYNOPSIS**

12679 UP du [-a | -s][-kx][-H | -L][file ...]

12680

**12681 DESCRIPTION**

12682 By default, the *du* utility shall write to standard output the size of the file space allocated to, and  
12683 the size of the file space allocated to each subdirectory of, the file hierarchy rooted in each of the  
12684 specified files. By default, when a symbolic link is encountered on the command line or in the  
12685 file hierarchy, *du* shall count the size of the symbolic link (rather than the file referenced by the  
12686 link), and shall not follow the link to another portion of the file hierarchy. The size of the file  
12687 space allocated to a file of type directory shall be defined as the sum total of space allocated to  
12688 all files in the file hierarchy rooted in the directory plus the space allocated to the directory itself.

12689 When *du* cannot *stat()* files or *stat()* or read directories, it shall report an error condition and the  
12690 final exit status is affected. Files with multiple links shall be counted and written for only one  
12691 entry. The directory entry that is selected in the report is unspecified. By default, file sizes shall  
12692 be written in 512-byte units, rounded up to the next 512-byte unit.

**12693 OPTIONS**

12694 The *du* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section 12.2,  
12695 Utility Syntax Guidelines.

12696 The following options shall be supported:

12697 **-a** In addition to the default output, report the size of each file not of type directory in  
12698 the file hierarchy rooted in the specified file. Regardless of the presence of the **-a**  
12699 option, non-directories given as *file* operands shall always be listed.

12700 **-H** If a symbolic link is specified on the command line, *du* shall count the size of the  
12701 file or file hierarchy referenced by the link.

12702 **-k** Write the files sizes in units of 1 024 bytes, rather than the default 512-byte units.

12703 **-L** If a symbolic link is specified on the command line or encountered during the  
12704 traversal of a file hierarchy, *du* shall count the size of the file or file hierarchy  
12705 referenced by the link.

12706 **-s** Instead of the default output, report only the total sum for each of the specified  
12707 files.

12708 **-x** When evaluating file sizes, evaluate only those files that have the same device as  
12709 the file specified by the *file* operand.

12710 Specifying more than one of the mutually-exclusive options **-H** and **-L** shall not be considered  
12711 an error. The last option specified shall determine the behavior of the utility.

**12712 OPERANDS**

12713 The following operand shall be supported:

12714 *file* The pathname of a file whose size is to be written. If no *file* is specified, the current  
12715 directory shall be used.

**12716 STDIN**

12717 Not used.

**12718 INPUT FILES**

12719 None.

**12720 ENVIRONMENT VARIABLES**

12721 The following environment variables shall affect the execution of *du*:

12722 **LANG** Provide a default value for the internationalization variables that are unset or null.  
(See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2, Internationalization Variables for the precedence of internationalization variables used to determine the values of locale categories.)

12726 **LC\_ALL** If set to a non-empty string value, override the values of all the other  
12727 internationalization variables.

12728 **LC\_CTYPE** Determine the locale for the interpretation of sequences of bytes of text data as  
12729 characters (for example, single-byte as opposed to multi-byte characters in  
12730 arguments).

**12731 *LC\_MESSAGES***

12732 Determine the locale that should be used to affect the format and contents of  
12733 diagnostic messages written to standard error.

12734 XSI **NLSPATH** Determine the location of message catalogs for the processing of *LC\_MESSAGES*.

**12735 ASYNCHRONOUS EVENTS**

12736 Default.

**12737 STDOUT**

12738 The output from *du* shall consist of the amount of space allocated to a file and the name of the  
12739 file, in the following format:

12740 "%d %s\n", <size>, <pathname>

**12741 STDERR**

12742 The standard error shall be used only for diagnostic messages.

**12743 OUTPUT FILES**

12744 None.

**12745 EXTENDED DESCRIPTION**

12746 None.

**12747 EXIT STATUS**

12748 The following exit values shall be returned:

12749 0 Successful completion.

12750 >0 An error occurred.

**12751 CONSEQUENCES OF ERRORS**

12752 Default.

## 12753 APPLICATION USAGE

12754 None.

## 12755 EXAMPLES

12756 None.

## 12757 RATIONALE

12758 The use of 512-byte units is historical practice and maintains compatibility with *ls* and other  
12759 utilities in this volume of IEEE Std 1003.1-2001. This does not mandate that the file system itself  
12760 be based on 512-byte blocks. The **-k** option was added as a compromise measure. It was agreed  
12761 by the standard developers that 512 bytes was the best default unit because of its complete  
12762 historical consistency on System V (*versus* the mixed 512/1 024-byte usage on BSD systems), and  
12763 that a **-k** option to switch to 1 024-byte units was a good compromise. Users who prefer the  
12764 1 024-byte quantity can easily alias *du* to *du -k* without breaking the many historical scripts  
12765 relying on the 512-byte units.

12766 The **-b** option was added to an early proposal to provide a resolution to the situation where  
12767 System V and BSD systems give figures for file sizes in *blocks*, which is an implementation-  
12768 defined concept. (In common usage, the block size is 512 bytes for System V and 1 024 bytes for  
12769 BSD systems.) However, **-b** was later deleted, since the default was eventually decided as 512-  
12770 byte units.

12771 Historical file systems provided no way to obtain exact figures for the space allocation given to  
12772 files. There are two known areas of inaccuracies in historical file systems: cases of *indirect blocks*  
12773 being used by the file system or *sparse* files yielding incorrectly high values. An indirect block is  
12774 space used by the file system in the storage of the file, but that need not be counted in the space  
12775 allocated to the file. A *sparse* file is one in which an *Iseek()* call has been made to a position  
12776 beyond the end of the file and data has subsequently been written at that point. A file system  
12777 need not allocate all the intervening zero-filled blocks to such a file. It is up to the  
12778 implementation to define exactly how accurate its methods are.

12779 The **-a** and **-s** options were mutually-exclusive in the original version of *du*. The POSIX Shell  
12780 and Utilities description is implied by the language in the SVID where **-s** is described as causing  
12781 “only the grand total” to be reported. Some systems may produce output for **-sa**, but a Strictly  
12782 Conforming POSIX Shell and Utilities Application cannot use that combination.

12783 The **-a** and **-s** options were adopted from the SVID except that the System V behavior of not  
12784 listing non-directories explicitly given as operands, unless the **-a** option is specified, was  
12785 considered a bug; the BSD-based behavior (report for all operands) is mandated. The default  
12786 behavior of *du* in the SVID with regard to reporting the failure to read files (it produces no  
12787 messages) was considered counter-intuitive, and thus it was specified that the POSIX Shell and  
12788 Utilities default behavior shall be to produce such messages. These messages can be turned off  
12789 with shell redirection to achieve the System V behavior.

12790 The **-x** option is historical practice on recent BSD systems. It has been adopted by this volume of  
12791 IEEE Std 1003.1-2001 because there was no other historical method of limiting the *du* search to a  
12792 single file hierarchy. This limitation of the search is necessary to make it possible to obtain file  
12793 space usage information about a file system on which other file systems are mounted, without  
12794 having to resort to a lengthy *find* and *awk* script.

## 12795 FUTURE DIRECTIONS

12796 None.

## 12797 SEE ALSO

12798        *ls*, the System Interfaces volume of IEEE Std 1003.1-2001, *stat()*

## 12799 CHANGE HISTORY

12800        First released in Issue 2.

## 12801 Issue 6

12802        This utility is marked as part of the User Portability Utilities option.

12803        The APPLICATION USAGE section is added.

12804        The obsolescent **-r** option has been removed.

12805        The Open Group Corrigendum U025/3 is applied. The *du* utility is reinstated, as it had  
12806        incorrectly been marked LEGACY in Issue 5.

12807        The **-H** and **-L** options for symbolic links are added as described in the IEEE P1003.2b draft  
12808        standard.

12809 **NAME**

12810 echo — write arguments to standard output

12811 **SYNOPSIS**12812 echo [*string* ...]12813 **DESCRIPTION**12814 The *echo* utility writes its arguments to standard output, followed by a <newline>. If there are  
12815 no arguments, only the <newline> is written.12816 **OPTIONS**12817 The *echo* utility shall not recognize the "—" argument in the manner specified by Guideline 10  
12818 of the Base Definitions volume of IEEE Std 1003.1-2001, Section 12.2, Utility Syntax Guidelines;  
12819 "—" shall be recognized as a string operand.

12820 Implementations shall not support any options.

12821 **OPERANDS**

12822 The following operands shall be supported:

12823 *string* A string to be written to standard output. If the first operand is **-n**, or if any of the 1  
12824 operands contain a backslash ('\') character, the results are implementation- 1  
12825 defined. 112826 XSI On XSI-conformant systems, if the first operand is **-n**, it shall be treated as a string, 1  
12827 not an option. The following character sequences shall be recognized on XSI- 1  
12828 conformant systems within any of the arguments: 1

12829 \a Write an &lt;alert&gt;.

12830 \b Write a &lt;backspace&gt;.

12831 \c Suppress the <newline> that otherwise follows the final argument in the 1  
12832 output. All characters following the '\c' in the arguments shall be 1  
12833 ignored.

12834 \f Write a &lt;form-feed&gt;.

12835 \n Write a &lt;newline&gt;.

12836 \r Write a &lt;carriage-return&gt;.

12837 \t Write a &lt;tab&gt;.

12838 \v Write a &lt;vertical-tab&gt;.

12839 \\ Write a backslash character.

12840 \0*num* Write an 8-bit value that is the zero, one, two, or three-digit octal number 1  
12841 *num*. 1

12842

12843 **STDIN**

12844 Not used.

12845 **INPUT FILES**

12846 None.

12847 **ENVIRONMENT VARIABLES**12848 The following environment variables shall affect the execution of *echo*:12849 *LANG* Provide a default value for the internationalization variables that are unset or null.  
12850 (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2,

|           |                                                                                                                                                                                                                                  |                                                                                                                                                                           |   |  |  |
|-----------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---|--|--|
| 12851     | Internationalization Variables for the precedence of internationalization variables used to determine the values of locale categories.)                                                                                          |                                                                                                                                                                           |   |  |  |
| 12852     |                                                                                                                                                                                                                                  |                                                                                                                                                                           |   |  |  |
| 12853     | <i>LC_ALL</i>                                                                                                                                                                                                                    | If set to a non-empty string value, override the values of all the other internationalization variables.                                                                  |   |  |  |
| 12854     |                                                                                                                                                                                                                                  |                                                                                                                                                                           |   |  |  |
| 12855 XSI | <i>LC_CTYPE</i>                                                                                                                                                                                                                  | Determine the locale for the interpretation of sequences of bytes of text data as characters (for example, single-byte as opposed to multi-byte characters in arguments). | 1 |  |  |
| 12856     |                                                                                                                                                                                                                                  |                                                                                                                                                                           |   |  |  |
| 12857     |                                                                                                                                                                                                                                  |                                                                                                                                                                           |   |  |  |
| 12858     | <i>LC_MESSAGES</i>                                                                                                                                                                                                               | Determine the locale that should be used to affect the format and contents of diagnostic messages written to standard error.                                              | 1 |  |  |
| 12859     |                                                                                                                                                                                                                                  |                                                                                                                                                                           |   |  |  |
| 12860     |                                                                                                                                                                                                                                  |                                                                                                                                                                           |   |  |  |
| 12861 XSI | <i>NLSPATH</i>                                                                                                                                                                                                                   | Determine the location of message catalogs for the processing of <i>LC_MESSAGES</i> .                                                                                     | 1 |  |  |
| 12862     | <b>ASYNCHRONOUS EVENTS</b>                                                                                                                                                                                                       |                                                                                                                                                                           |   |  |  |
| 12863     | Default.                                                                                                                                                                                                                         |                                                                                                                                                                           |   |  |  |
| 12864     | <b>STDOUT</b>                                                                                                                                                                                                                    |                                                                                                                                                                           |   |  |  |
| 12865     | The <i>echo</i> utility arguments shall be separated by single <space>s and a <newline> shall follow the last argument. Output transformations shall occur based on the escape sequences in the input. See the OPERANDS section. | 1                                                                                                                                                                         | 1 |  |  |
| 12866 XSI |                                                                                                                                                                                                                                  |                                                                                                                                                                           |   |  |  |
| 12867     |                                                                                                                                                                                                                                  |                                                                                                                                                                           |   |  |  |
| 12868     | <b>STDERR</b>                                                                                                                                                                                                                    |                                                                                                                                                                           |   |  |  |
| 12869     | The standard error shall be used only for diagnostic messages.                                                                                                                                                                   |                                                                                                                                                                           |   |  |  |
| 12870     | <b>OUTPUT FILES</b>                                                                                                                                                                                                              |                                                                                                                                                                           |   |  |  |
| 12871     | None.                                                                                                                                                                                                                            |                                                                                                                                                                           |   |  |  |
| 12872     | <b>EXTENDED DESCRIPTION</b>                                                                                                                                                                                                      |                                                                                                                                                                           |   |  |  |
| 12873     | None.                                                                                                                                                                                                                            |                                                                                                                                                                           |   |  |  |
| 12874     | <b>EXIT STATUS</b>                                                                                                                                                                                                               |                                                                                                                                                                           |   |  |  |
| 12875     | The following exit values shall be returned:                                                                                                                                                                                     |                                                                                                                                                                           |   |  |  |
| 12876     | 0 Successful completion.                                                                                                                                                                                                         |                                                                                                                                                                           |   |  |  |
| 12877     | >0 An error occurred.                                                                                                                                                                                                            |                                                                                                                                                                           |   |  |  |
| 12878     | <b>CONSEQUENCES OF ERRORS</b>                                                                                                                                                                                                    |                                                                                                                                                                           |   |  |  |
| 12879     | Default.                                                                                                                                                                                                                         |                                                                                                                                                                           |   |  |  |
| 12880     | <b>APPLICATION USAGE</b>                                                                                                                                                                                                         |                                                                                                                                                                           |   |  |  |
| 12881     | It is not possible to use <i>echo</i> portably across all POSIX systems unless both <b>-n</b> (as the first argument) and escape sequences are omitted.                                                                          | 1                                                                                                                                                                         | 1 |  |  |
| 12882     |                                                                                                                                                                                                                                  |                                                                                                                                                                           |   |  |  |
| 12883     | The <i>printf</i> utility can be used portably to emulate any of the traditional behaviors of the <i>echo</i> utility as follows (assuming that IFS has its standard value or is unset):                                         |                                                                                                                                                                           |   |  |  |
| 12884     |                                                                                                                                                                                                                                  |                                                                                                                                                                           |   |  |  |
| 12885     | • The historic System V <i>echo</i> and the requirements on XSI implementations in this volume of IEEE Std 1003.1-2001 are equivalent to:                                                                                        | 1                                                                                                                                                                         | 1 |  |  |
| 12886     |                                                                                                                                                                                                                                  |                                                                                                                                                                           |   |  |  |
| 12887     | <i>printf "%b\n" "\$*"</i>                                                                                                                                                                                                       |                                                                                                                                                                           |   |  |  |
| 12888     | • The BSD <i>echo</i> is equivalent to:                                                                                                                                                                                          |                                                                                                                                                                           |   |  |  |
| 12889     | <i>if [ "X\$1" = "X-n" ]</i>                                                                                                                                                                                                     | 1                                                                                                                                                                         | 1 |  |  |
| 12890     | <i>then</i>                                                                                                                                                                                                                      |                                                                                                                                                                           |   |  |  |
| 12891     | <i>shift</i>                                                                                                                                                                                                                     |                                                                                                                                                                           |   |  |  |
| 12892     | <i>printf "%s" "\$*"</i>                                                                                                                                                                                                         |                                                                                                                                                                           |   |  |  |
| 12893     | <i>else</i>                                                                                                                                                                                                                      |                                                                                                                                                                           |   |  |  |

12894                *printf "%s\n" "\$@"*  
12895                fi

12896      New applications are encouraged to use *printf* instead of *echo*.

## 12897 EXAMPLES

12898      None.

## 12899 RATIONALE

12900      The *echo* utility has not been made obsolescent because of its extremely widespread use in  
12901      historical applications. Conforming applications that wish to do prompting without <newline>s  
12902      or that could possibly be expecting to echo a **-n**, should use the *printf* utility derived from the  
12903      Ninth Edition system.

12904      As specified, *echo* writes its arguments in the simplest of ways. The two different historical  
12905      versions of *echo* vary in fatally incompatible ways.

12906      The BSD *echo* checks the first argument for the string **-n** which causes it to suppress the  
12907      <newline> that would otherwise follow the final argument in the output.

12908      The System V *echo* does not support any options, but allows escape sequences within its  
12909      operands, as described for XSI implementations in the OPERANDS section. 1

12910      The *echo* utility does not support Utility Syntax Guideline 10 because historical applications  
12911      depend on *echo* to echo *all* of its arguments, except for the **-n** option in the BSD version.

## 12912 FUTURE DIRECTIONS

12913      None.

## 12914 SEE ALSO

12915      *printf*

## 12916 CHANGE HISTORY

12917      First released in Issue 2.

### 12918 Issue 5

12919      In the OPTIONS section, the last sentence is changed to indicate that implementations “do not”  
12920      support any options; in the previous issue this said “need not”.

### 12921 Issue 6

12922      The following new requirements on POSIX implementations derive from alignment with the  
12923      Single UNIX Specification:

- 12924      • A set of character sequences is defined as *string* operands.
- 12925      • *LC\_CTYPE* is added to the list of environment variables affecting *echo*.
- 12926      • In the OPTIONS section, implementations shall not support any options.

12927      IEEE Std 1003.1-2001/Cor 1-2002, item XCU/TC1/D6/21 is applied, so that the *echo* utility can 1  
12928      accommodate historical BSD behavior. 1

12929 **NAME**

12930        *ed* — edit text

12931 **SYNOPSIS**

12932        *ed* [*-p string*] [*-s*] [*file*]

12933 **DESCRIPTION**

12934        The *ed* utility is a line-oriented text editor that uses two modes: *command mode* and *input mode*.  
12935        In command mode the input characters shall be interpreted as commands, and in input mode  
12936        they shall be interpreted as text. See the EXTENDED DESCRIPTION section.

12937 **OPTIONS**

12938        The *ed* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section 12.2,  
12939        Utility Syntax Guidelines.

12940        The following options shall be supported:

12941        **-p** *string*      Use *string* as the prompt string when in command mode. By default, there shall be  
12942        no prompt string.

12943        **-s**              Suppress the writing of byte counts by **e**, **E**, **r**, and **w** commands and of the '!'  
12944        prompt after a *!command*.

12945 **OPERANDS**

12946        The following operand shall be supported:

12947        *file*             If the *file* argument is given, *ed* shall simulate an **e** command on the file named by  
12948        the pathname, *file*, before accepting commands from the standard input. If the *file*  
12949        operand is '**-**', the results are unspecified.

12950 **STDIN**

12951        The standard input shall be a text file consisting of commands, as described in the EXTENDED  
12952        DESCRIPTION section.

12953 **INPUT FILES**

12954        The input files shall be text files.

12955 **ENVIRONMENT VARIABLES**

12956        The following environment variables shall affect the execution of *ed*:

12957        **HOME**          Determine the pathname of the user's home directory.

12958        **LANG**          Provide a default value for the internationalization variables that are unset or null.  
12959        (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2,  
12960        Internationalization Variables for the precedence of internationalization variables  
12961        used to determine the values of locale categories.)

12962        **LC\_ALL**        If set to a non-empty string value, override the values of all the other  
12963        internationalization variables.

12964        **LC\_COLLATE**

12965        Determine the locale for the behavior of ranges, equivalence classes, and multi-  
12966        character collating elements within regular expressions.

12967        **LC\_CTYPE**       Determine the locale for the interpretation of sequences of bytes of text data as  
12968        characters (for example, single-byte as opposed to multi-byte characters in  
12969        arguments and input files) and the behavior of character classes within regular  
12970        expressions.

12971        **LC\_MESSAGES**

12972        Determine the locale that should be used to affect the format and contents of

12973 diagnostic messages written to standard error and informative messages written to  
12974 standard output.

12975 XSI **NLSPATH** Determine the location of message catalogs for the processing of *LC\_MESSAGES*.

## 12976 ASYNCHRONOUS EVENTS

12977 The *ed* utility shall take the standard action for all signals (see the ASYNCHRONOUS EVENTS  
12978 section in Section 1.11 (on page 20)) with the following exceptions:

12979 SIGINT The *ed* utility shall interrupt its current activity, write the string "?\n" to standard  
12980 output, and return to command mode (see the EXTENDED DESCRIPTION  
12981 section).

12982 SIGHUP If the buffer is not empty and has changed since the last write, the *ed* utility shall  
12983 attempt to write a copy of the buffer in a file. First, the file named **ed.hup** in the  
12984 current directory shall be used; if that fails, the file named **ed.hup** in the directory  
12985 named by the *HOME* environment variable shall be used. In any case, the *ed* utility  
12986 shall exit without returning to command mode.

12987 SIGQUIT The *ed* utility shall ignore this event.

## 12988 STDOUT

12989 Various editing commands and the prompting feature (see **-p**) write to standard output, as  
12990 described in the EXTENDED DESCRIPTION section.

## 12991 STDERR

12992 The standard error shall be used only for diagnostic messages.

## 12993 OUTPUT FILES

12994 The output files shall be text files whose formats are dependent on the editing commands given.

## 12995 EXTENDED DESCRIPTION

12996 The *ed* utility shall operate on a copy of the file it is editing; changes made to the copy shall have  
12997 no effect on the file until a **w** (write) command is given. The copy of the text is called the *buffer*.

12998 Commands to *ed* have a simple and regular structure: zero, one, or two *addresses* followed by a  
12999 single-character *command*, possibly followed by parameters to that command. These addresses  
13000 specify one or more lines in the buffer. Every command that requires addresses has default  
13001 addresses, so that the addresses very often can be omitted. If the **-p** option is specified, the  
13002 prompt string shall be written to standard output before each command is read.

13003 In general, only one command can appear on a line. Certain commands allow text to be input.  
13004 This text is placed in the appropriate place in the buffer. While *ed* is accepting text, it is said to be  
13005 in *input mode*. In this mode, no commands shall be recognized; all input is merely collected.  
13006 Input mode is terminated by entering a line consisting of two characters: a period ('.')  
13007 followed by a <newline>. This line is not considered part of the input text.

## 13008 Regular Expressions in ed

13009 The *ed* utility shall support basic regular expressions, as described in the Base Definitions  
13010 volume of IEEE Std 1003.1-2001, Section 9.3, Basic Regular Expressions. Since regular  
13011 expressions in *ed* are always matched against single lines (excluding the terminating  
13012 <newline>s), never against any larger section of text, there is no way for a regular expression to  
13013 match a <newline>.

13014 A null RE shall be equivalent to the last RE encountered.

13015 Regular expressions are used in addresses to specify lines, and in some commands (for example,  
13016 the **s** substitute command) to specify portions of a line to be substituted.

13017      **Addresses in ed**

13018      Addressing in *ed* relates to the current line. Generally, the current line is the last line affected by a  
13019      command. The current line number is the address of the current line. If the edit buffer is not  
13020      empty, the initial value for the current line shall be the last line in the edit buffer; otherwise, zero.

13021      Addresses shall be constructed as follows:

- 13022      1. The period character ('.') shall address the current line.
- 13023      2. The dollar sign character ('\$') shall address the last line of the edit buffer.
- 13024      3. The positive decimal number *n* shall address the *n*th line of the edit buffer.
- 13025      4. The apostrophe-x character pair ("'x") shall address the line marked with the mark name  
13026      character *x*, which shall be a lowercase letter from the portable character set. It shall be an  
13027      error if the character has not been set to mark a line or if the line that was marked is not  
13028      currently present in the edit buffer.
- 13029      5. A BRE enclosed by slash characters ('/') shall address the first line found by searching  
13030      forwards from the line following the current line toward the end of the edit buffer and  
13031      stopping at the first line for which the line excluding the terminating <newline> matches  
13032      the BRE. The BRE consisting of a null BRE delimited by a pair of slash characters shall  
13033      address the next line for which the line excluding the terminating <newline> matches the  
13034      last BRE encountered. In addition, the second slash can be omitted at the end of a  
13035      command line. Within the BRE, a backslash-slash pair ("\/") shall represent a literal slash  
13036      instead of the BRE delimiter. If necessary, the search shall wrap around to the beginning of  
13037      the buffer and continue up to and including the current line, so that the entire buffer is  
13038      searched.
- 13039      6. A BRE enclosed by question-mark characters ('?') shall address the first line found by  
13040      searching backwards from the line preceding the current line toward the beginning of the  
13041      edit buffer and stopping at the first line for which the line excluding the terminating  
13042      <newline> matches the BRE. The BRE consisting of a null BRE delimited by a pair of  
13043      question-mark characters ("??") shall address the previous line for which the line  
13044      excluding the terminating <newline> matches the last BRE encountered. In addition, the  
13045      second question-mark can be omitted at the end of a command line. Within the BRE, a  
13046      backslash-question-mark pair ("\?") shall represent a literal question mark instead of the  
13047      BRE delimiter. If necessary, the search shall wrap around to the end of the buffer and  
13048      continue up to and including the current line, so that the entire buffer is searched.
- 13049      7. A plus-sign ('+') or hyphen character ('-') followed by a decimal number shall address  
13050      the current line plus or minus the number. A plus-sign or hyphen character not followed  
13051      by a decimal number shall address the current line plus or minus 1.

13052      Addresses can be followed by zero or more address offsets, optionally <blank>-separated.  
13053      Address offsets are constructed as follows:

- 13054      • A plus-sign or hyphen character followed by a decimal number shall add or subtract,  
13055      respectively, the indicated number of lines to or from the address. A plus-sign or hyphen  
13056      character not followed by a decimal number shall add or subtract 1 to or from the address.
- 13057      • A decimal number shall add the indicated number of lines to the address.

13058      It shall not be an error for an intermediate address value to be less than zero or greater than the  
13059      last line in the edit buffer. It shall be an error for the final address value to be less than zero or  
13060      greater than the last line in the edit buffer. It shall be an error if a search for a BRE fails to find a  
13061      matching line.

13062 Commands accept zero, one, or two addresses. If more than the required number of addresses  
 13063 are provided to a command that requires zero addresses, it shall be an error. Otherwise, if more  
 13064 than the required number of addresses are provided to a command, the addresses specified first  
 13065 shall be evaluated and then discarded until the maximum number of valid addresses remain, for  
 13066 the specified command.

13067 Addresses shall be separated from each other by a comma (',') or semicolon character (';').  
 13068 In the case of a semicolon separator, the current line ('.') shall be set to the first address, and  
 13069 only then will the second address be calculated. This feature can be used to determine the  
 13070 starting line for forwards and backwards searches; see rules 5. and 6.

13071 Addresses can be omitted on either side of the comma or semicolon separator, in which case the  
 13072 resulting address pairs shall be as follows:

| Specified | Resulting   |
|-----------|-------------|
| ,         | 1 , \$      |
| , addr    | 1 , addr    |
| addr ,    | addr , addr |
| ;         | . ; \$      |
| ; addr    | . ; addr    |
| addr ;    | addr ; addr |

13080 Any <blank>s included between addresses, address separators, or address offsets shall be  
 13081 ignored.

## 13082 Commands in ed

13083 In the following list of *ed* commands, the default addresses are shown in parentheses. The  
 13084 number of addresses shown in the default shall be the number expected by the command. The  
 13085 parentheses are not part of the address; they show that the given addresses are the default.

13086 It is generally invalid for more than one command to appear on a line. However, any command  
 13087 (except **e**, **E**, **f**, **q**, **Q**, **r**, **w**, and **l**) can be suffixed by the letter **l**, **n**, or **p**; in which case, except for  
 13088 the **l**, **n**, and **p** commands, the command shall be executed and then the new current line shall be  
 13089 written as described below under the **l**, **n**, and **p** commands. When an **l**, **n**, or **p** suffix is used  
 13090 with an **l**, **n**, or **p** command, the command shall write to standard output as described below, but  
 13091 it is unspecified whether the suffix writes the current line again in the requested format or  
 13092 whether the suffix has no effect. For example, the **pl** command (base **p** command with an **l**  
 13093 suffix) shall either write just the current line or write it twice—once as specified for **p** and once  
 13094 as specified for **l**. Also, the **g**, **G**, **v**, and **V** commands shall take a command as a parameter.

13095 Each address component can be preceded by zero or more <blank>s. The command letter can be  
 13096 preceded by zero or more <blank>s. If a suffix letter (**l**, **n**, or **p**) is given, the application shall  
 13097 ensure that it immediately follows the command.

13098 The **e**, **E**, **f**, **r**, and **w** commands shall take an optional *file* parameter, separated from the  
 13099 command letter by one or more <blank>s.

13100 If changes have been made in the buffer since the last **w** command that wrote the entire buffer,  
 13101 *ed* shall warn the user if an attempt is made to destroy the editor buffer via the **e** or **q** commands.  
 13102 The *ed* utility shall write the string:

13103 " ?\n "

13104 (followed by an explanatory message if *help mode* has been enabled via the **H** command) to  
 13105 standard output and shall continue in command mode with the current line number unchanged.  
 13106 If the **e** or **q** command is repeated with no intervening command, it shall take effect.

13107 If a terminal disconnect is detected:

- 13108 • If the buffer is not empty and has changed since the last write, the *ed* utility shall attempt to  
13109 write a copy of the buffer to a file named **ed.hup** in the current directory. If this write fails, *ed*  
13110 shall attempt to write a copy of the buffer to a filename **ed.hup** in the directory named by the  
13111 *HOME* environment variable. If both these attempts fail, *ed* shall exit without saving the  
13112 buffer.
- 13113 • The *ed* utility shall not write the file to the currently remembered pathname or return to  
13114 command mode, and shall terminate with a non-zero exit status.

13115 If an end-of-file is detected on standard input:

- 13116 • If the *ed* utility is in input mode, *ed* shall terminate input mode and return to command mode.  
13117 It is unspecified if any partially entered lines (that is, input text without a terminating  
13118 <newline>) are discarded from the input text.
- 13119 • If the *ed* utility is in command mode, it shall act as if a **q** command had been entered.

13120 If the closing delimiter of an RE or of a replacement string (for example, ' / ') in a **g**, **G**, **s**, **v**, or **V**  
13121 command would be the last character before a <newline>, that delimiter can be omitted, in  
13122 which case the addressed line shall be written. For example, the following pairs of commands  
13123 are equivalent:

13124    s/s1/s2    s/s1/s2/p  
13125    g/s1       g/s1/p  
13126    ?s1?       ?s1?

13127 If an invalid command is entered, *ed* shall write the string:

13128    "?\n"

13129 (followed by an explanatory message if *help mode* has been enabled via the **H** command) to  
13130 standard output and shall continue in command mode with the current line number unchanged.

### 13131 Append Command

13132    *Synopsis:*    ( . )a  
13133                  <text>  
13134                  .

13135 The **a** command shall read the given text and append it after the addressed line; the current line  
13136 number shall become the address of the last inserted line or, if there were none, the addressed  
13137 line. Address 0 shall be valid for this command; it shall cause the appended text to be placed at  
13138 the beginning of the buffer.

### 13139 Change Command

13140    *Synopsis:*    ( . . . )c  
13141                  <text>  
13142                  .

13143 The **c** command shall delete the addressed lines, then accept input text that replaces these lines;  
13144 the current line shall be set to the address of the last line input; or, if there were none, at the line  
13145 after the last line deleted; if the lines deleted were originally at the end of the buffer, the current  
13146 line number shall be set to the address of the new last line; if no lines remain in the buffer, the  
13147 current line number shall be set to zero. Address 0 shall be valid for this command; it shall be  
13148 interpreted as if address 1 were specified.

13149       **Delete Command**

13150       *Synopsis:*     ( . , . )d

13151       The **d** command shall delete the addressed lines from the buffer. The address of the line after the  
13152       last line deleted shall become the current line number; if the lines deleted were originally at the  
13153       end of the buffer, the current line number shall be set to the address of the new last line; if no  
13154       lines remain in the buffer, the current line number shall be set to zero.

13155       **Edit Command**

13156       *Synopsis:*     e [file]

13157       The **e** command shall delete the entire contents of the buffer and then read in the file named by  
13158       the pathname *file*. The current line number shall be set to the address of the last line of the  
13159       buffer. If no pathname is given, the currently remembered pathname, if any, shall be used (see  
13160       the **f** command). The number of bytes read shall be written to standard output, unless the **-s**  
13161       option was specified, in the following format:

13162       "%d\n" , <number of bytes read>

13163       The name *file* shall be remembered for possible use as a default pathname in subsequent **e**, **E**, **r**,  
13164       and **w** commands. If *file* is replaced by '!', the rest of the line shall be taken to be a shell  
13165       command line whose output is to be read. Such a shell command line shall not be remembered  
13166       as the current *file*. All marks shall be discarded upon the completion of a successful **e** command.  
13167       If the buffer has changed since the last time the entire buffer was written, the user shall be  
13168       warned, as described previously.

13169       **Edit Without Checking Command**

13170       *Synopsis:*     E [file]

13171       The **E** command shall possess all properties and restrictions of the **e** command except that the  
13172       editor shall not check to see whether any changes have been made to the buffer since the last **w**  
13173       command.

13174       **Filename Command**

13175       *Synopsis:*     f [file]

13176       If *file* is given, the **f** command shall change the currently remembered pathname to *file*; whether  
13177       the name is changed or not, it shall then write the (possibly new) currently remembered  
13178       pathname to the standard output in the following format:

13179       "%s\n" , <pathname>

13180       The current line number shall be unchanged.

13181       **Global Command**

13182       *Synopsis:*     (1,\$)g/RE/command list

13183       In the **g** command, the first step shall be to mark every line for which the line excluding the  
13184       terminating <newline> matches the given RE. Then, going sequentially from the beginning of  
13185       the file to the end of the file, the given *command list* shall be executed for each marked line, with  
13186       the current line number set to the address of that line. Any line modified by the *command list*  
13187       shall be unmarked. When the **g** command completes, the current line number shall have the  
13188       value assigned by the last command in the *command list*. If there were no matching lines, the  
13189       current line number shall not be changed. A single command or the first of a list of commands

13190 shall appear on the same line as the global command. All lines of a multi-line list except the last  
13191 line shall be ended with a backslash preceding the terminating <newline>; the **a**, **i**, and **c**  
13192 commands and associated input are permitted. The ' . ' terminating input mode can be omitted  
13193 if it would be the last line of the *command list*. An empty *command list* shall be equivalent to the **p**  
13194 command. The use of the **g**, **G**, **v**, **V**, and **!** commands in the *command list* produces undefined  
13195 results. Any character other than <space> or <newline> can be used instead of a slash to delimit  
13196 the RE. Within the RE, the RE delimiter itself can be used as a literal character if it is preceded by  
13197 a backslash.

### 13198 **Interactive Global Command**

13199 *Synopsis:* ( 1 , \$ )G/RE/

13200 In the **G** command, the first step shall be to mark every line for which the line excluding the  
13201 terminating <newline> matches the given RE. Then, for every such line, that line shall be  
13202 written, the current line number shall be set to the address of that line, and any one command  
13203 (other than one of the **a**, **c**, **i**, **g**, **G**, **v**, and **V** commands) shall be read and executed. A <newline>  
13204 shall act as a null command (causing no action to be taken on the current line); an '&' shall  
13205 cause the re-execution of the most recent non-null command executed within the current  
13206 invocation of **G**. Note that the commands input as part of the execution of the **G** command can  
13207 address and affect any lines in the buffer. Any line modified by the command shall be  
13208 unmarked. The final value of the current line number shall be the value set by the last command  
13209 successfully executed. (Note that the last command successfully executed shall be the **G**  
13210 command itself if a command fails or the null command is specified.) If there were no matching  
13211 lines, the current line number shall not be changed. The **G** command can be terminated by a  
13212 SIGINT signal. Any character other than <space> or <newline> can be used instead of a slash to  
13213 delimit the RE and the replacement. Within the RE, the RE delimiter itself can be used as a literal  
13214 character if it is preceded by a backslash.

### 13215 **Help Command**

13216 *Synopsis:* h

13217 The **h** command shall write a short message to standard output that explains the reason for the  
13218 most recent '?' notification. The current line number shall be unchanged.

### 13219 **Help-Mode Command**

13220 *Synopsis:* H

13221 The **H** command shall cause **ed** to enter a mode in which help messages (see the **h** command)  
13222 shall be written to standard output for all subsequent '?' notifications. The **H** command  
13223 alternately shall turn this mode on and off; it is initially off. If the help-mode is being turned on,  
13224 the **H** command also explains the previous '?' notification, if there was one. The current line  
13225 number shall be unchanged.

### 13226 **Insert Command**

13227 *Synopsis:* ( . )i  
13228           <text>  
13229           .

13230 The **i** command shall insert the given text before the addressed line; the current line is set to the  
13231 last inserted line or, if there was none, to the addressed line. This command differs from the **a**  
13232 command only in the placement of the input text. Address 0 shall be valid for this command; it  
13233 shall be interpreted as if address 1 were specified.

13234       **Join Command**

13235       *Synopsis:*     ( . , . +1 ) j

13236       The **j** command shall join contiguous lines by removing the appropriate <newline>s. If exactly  
13237       one address is given, this command shall do nothing. If lines are joined, the current line number  
13238       shall be set to the address of the joined line; otherwise, the current line number shall be  
13239       unchanged.

13240       **Mark Command**

13241       *Synopsis:*     ( . ) kx

13242       The **k** command shall mark the addressed line with name *x*, which the application shall ensure is  
13243       a lowercase letter from the portable character set. The address "'x'" shall then refer to this line;  
13244       the current line number shall be unchanged.

13245       **List Command**

13246       *Synopsis:*     ( . , . ) l

13247       The **l** command shall write to standard output the addressed lines in a visually unambiguous  
13248       form. The characters listed in the Base Definitions volume of IEEE Std 1003.1-2001, Table 5-1,  
13249       Escape Sequences and Associated Actions ('\\', '\\a', '\\b', '\\f', '\\r', '\\t', '\\v') shall  
13250       be written as the corresponding escape sequence; the '\\n' in that table is not applicable. Non-  
13251       printable characters not in the table shall be written as one three-digit octal number (with a  
13252       preceding backslash character) for each byte in the character (most significant byte first).      2

13253       Long lines shall be folded, with the point of folding indicated by <newline> preceded by a  
13254       backslash; the length at which folding occurs is unspecified, but should be appropriate for the  
13255       output device. The end of each line shall be marked with a '\$', and '\$' characters within the  
13256       text shall be written with a preceding backslash. An **l** command can be appended to any other  
13257       command other than **e**, **E**, **f**, **q**, **Q**, **r**, **w**, or **!**. The current line number shall be set to the address of  
13258       the last line written.

13259       **Move Command**

13260       *Synopsis:*     ( . , . ) maddress

13261       The **m** command shall reposition the addressed lines after the line addressed by *address*.  
13262       Address 0 shall be valid for *address* and cause the addressed lines to be moved to the beginning  
13263       of the buffer. It shall be an error if *address* *address* falls within the range of moved lines. The  
13264       current line number shall be set to the address of the last line moved.

13265       **Number Command**

13266       *Synopsis:*     ( . , . ) n

13267       The **n** command shall write to standard output the addressed lines, preceding each line by its  
13268       line number and a <tab>; the current line number shall be set to the address of the last line  
13269       written. The **n** command can be appended to any command other than **e**, **E**, **f**, **q**, **Q**, **r**, **w**, or **!**.

13270 **Print Command**13271 *Synopsis:* ( . . . )**p**

13272 The **p** command shall write to standard output the addressed lines; the current line number shall  
13273 be set to the address of the last line written. The **p** command can be appended to any command  
13274 other than **e**, **E**, **f**, **q**, **Q**, **r**, **w**, or **!**.

13275 **Prompt Command**13276 *Synopsis:* **P**

13277 The **P** command shall cause *ed* to prompt with an asterisk (' \* ') (or *string*, if **-p** is specified) for  
13278 all subsequent commands. The **P** command alternatively shall turn this mode on and off; it shall  
13279 be initially on if the **-p** option is specified; otherwise, off. The current line number shall be  
13280 unchanged.

13281 **Quit Command**13282 *Synopsis:* **q**

13283 The **q** command shall cause *ed* to exit. If the buffer has changed since the last time the entire  
13284 buffer was written, the user shall be warned, as described previously.

13285 **Quit Without Checking Command**13286 *Synopsis:* **Q**

13287 The **Q** command shall cause *ed* to exit without checking whether changes have been made in the  
13288 buffer since the last **w** command.

13289 **Read Command**13290 *Synopsis:* (\$)**r** [*file*]

13291 The **r** command shall read in the file named by the pathname *file* and append it after the  
13292 addressed line. If no *file* argument is given, the currently remembered pathname, if any, shall be  
13293 used (see the **e** and **f** commands). The currently remembered pathname shall not be changed  
13294 unless there is no remembered pathname. Address 0 shall be valid for **r** and shall cause the file to  
13295 be read at the beginning of the buffer. If the read is successful, and **-s** was not specified, the  
13296 number of bytes read shall be written to standard output in the following format:

13297 "%d\n", &lt;number of bytes read&gt;

13298 The current line number shall be set to the address of the last line read in. If *file* is replaced by  
13299 ' ! ', the rest of the line shall be taken to be a shell command line whose output is to be read.  
13300 Such a shell command line shall not be remembered as the current pathname.

13301 **Substitute Command**13302 *Synopsis:* ( . . . )**s**/*RE/replacement/flags*

13303 The **s** command shall search each addressed line for an occurrence of the specified *RE* and  
13304 replace either the first or all (non-overlapped) matched strings with the *replacement*; see the  
13305 following description of the **g** suffix. It is an error if the substitution fails on every addressed  
13306 line. Any character other than <space> or <newline> can be used instead of a slash to delimit the  
13307 *RE* and the *replacement*. Within the *RE*, the *RE* delimiter itself can be used as a literal character  
13308 if it is preceded by a backslash. The current line shall be set to the address of the last line on  
13309 which a substitution occurred.

13310 An ampersand ('&') appearing in the *replacement* shall be replaced by the string matching the  
13311 RE on the current line. The special meaning of '&' in this context can be suppressed by  
13312 preceding it by backslash. As a more general feature, the characters '\n', where *n* is a digit,  
13313 shall be replaced by the text matched by the corresponding back-reference expression. When the  
13314 character '%' is the only character in the *replacement*, the *replacement* used in the most recent  
13315 substitute command shall be used as the *replacement* in the current substitute command; if there  
13316 was no previous substitute command, the use of '%' in this manner shall be an error. The '%'  
13317 shall lose its special meaning when it is in a replacement string of more than one character or is  
13318 preceded by a backslash. For each backslash ('\') encountered in scanning *replacement* from  
13319 beginning to end, the following character shall lose its special meaning (if any). It is unspecified  
13320 what special meaning is given to any character other than '&', '\', '%', or digits.

13321 A line can be split by substituting a <newline> into it. The application shall ensure it escapes the  
13322 <newline> in the *replacement* by preceding it by backslash. Such substitution cannot be done as  
13323 part of a **g** or **v** *command list*. The current line number shall be set to the address of the last line  
13324 on which a substitution is performed. If no substitution is performed, the current line number  
13325 shall be unchanged. If a line is split, a substitution shall be considered to have been performed  
13326 on each of the new lines for the purpose of determining the new current line number. A  
13327 substitution shall be considered to have been performed even if the replacement string is  
13328 identical to the string that it replaces.

13329 The application shall ensure that the value of *flags* is zero or more of:

- 13330 **count** Substitute for the *count*th occurrence only of the RE found on each addressed line.
- 13331 **g** Globally substitute for all non-overlapping instances of the RE rather than just the first  
13332 one. If both **g** and **count** are specified, the results are unspecified.
- 13333 **I** Write to standard output the final line in which a substitution was made. The line shall  
13334 be written in the format specified for the **I** command.
- 13335 **n** Write to standard output the final line in which a substitution was made. The line shall  
13336 be written in the format specified for the **n** command.
- 13337 **p** Write to standard output the final line in which a substitution was made. The line shall  
13338 be written in the format specified for the **p** command.

### 13339 **Copy Command**

13340 *Synopsis:* ( . . . )taddress

13341 The **t** command shall be equivalent to the **m** command, except that a copy of the addressed lines  
13342 shall be placed after address *address* (which can be 0); the current line number shall be set to the  
13343 address of the last line added.

### 13344 **Undo Command**

13345 *Synopsis:* u

13346 The **u** command shall nullify the effect of the most recent command that modified anything in  
13347 the buffer, namely the most recent **a**, **c**, **d**, **g**, **i**, **j**, **m**, **r**, **s**, **t**, **u**, **v**, **G**, or **V** command. All changes  
13348 made to the buffer by a **g**, **G**, **v**, or **V** global command shall be undone as a single change; if no  
13349 changes were made by the global command (such as with **g/RE/p**), the **u** command shall have  
13350 no effect. The current line number shall be set to the value it had immediately before the  
13351 command being undone started.

13352       **Global Non-Matched Command**

13353       *Synopsis:*     (1,\$)v/*RE*/command list

13354       This command shall be equivalent to the global command g except that the lines that are marked  
13355       during the first step shall be those for which the line excluding the terminating <newline> does  
13356       not match the RE.

13357       **Interactive Global Not-Matched Command**

13358       *Synopsis:*     (1,\$)V/*RE*/

13359       This command shall be equivalent to the interactive global command G except that the lines that  
13360       are marked during the first step shall be those for which the line excluding the terminating  
13361       <newline> does not match the RE.

13362       **Write Command**

13363       *Synopsis:*     (1,\$)w [*file*]

13364       The w command shall write the addressed lines into the file named by the pathname *file*. The  
13365       command shall create the file, if it does not exist, or shall replace the contents of the existing file.  
13366       The currently remembered pathname shall not be changed unless there is no remembered  
13367       pathname. If no pathname is given, the currently remembered pathname, if any, shall be used  
13368       (see the e and f commands); the current line number shall be unchanged. If the command is  
13369       successful, the number of bytes written shall be written to standard output, unless the -s option  
13370       was specified, in the following format:

13371       "%d\n", <number of bytes written>

13372       If *file* begins with '!', the rest of the line shall be taken to be a shell command line whose  
13373       standard input shall be the addressed lines. Such a shell command line shall not be remembered  
13374       as the current pathname. This usage of the write command with '!' shall not be considered as a  
13375       "last w command that wrote the entire buffer", as described previously; thus, this alone shall not  
13376       prevent the warning to the user if an attempt is made to destroy the editor buffer via the e or q  
13377       commands.

13378       **Line Number Command**

13379       *Synopsis:*     ( \$ )=

13380       The line number of the addressed line shall be written to standard output in the following  
13381       format:

13382       "%d\n", <line number>

13383       The current line number shall be unchanged by this command.

13384       **Shell Escape Command**

13385       *Synopsis:*     !*command*

13386       The remainder of the line after the '!' shall be sent to the command interpreter to be  
13387       interpreted as a shell command line. Within the text of that shell command line, the unescaped  
13388       character '%' shall be replaced with the remembered pathname; if a '!' appears as the first  
13389       character of the command, it shall be replaced with the text of the previous shell command  
13390       executed via '!'. Thus, "!!" shall repeat the previous !*command*. If any replacements of '%' or  
13391       '!' are performed, the modified line shall be written to the standard output before *command* is  
13392       executed. The ! command shall write:

13393       "! \n"  
13394       to standard output upon completion, unless the **-s** option is specified. The current line number  
13395       shall be unchanged.

### 13396       **Null Command**

13397       *Synopsis:*     ( .+1 )

13398       An address alone on a line shall cause the addressed line to be written. A <newline> alone shall  
13399       be equivalent to "+1p". The current line number shall be set to the address of the written line.

### 13400       **EXIT STATUS**

13401       The following exit values shall be returned:

13402       0      Successful completion without any file or command errors.

13403       >0     An error occurred.

### 13404       **CONSEQUENCES OF ERRORS**

13405       When an error in the input script is encountered, or when an error is detected that is a  
13406       consequence of the data (not) present in the file or due to an external condition such as a read or  
13407       write error:

- 13408       • If the standard input is a terminal device file, all input shall be flushed, and a new command  
13409       read.
- 13410       • If the standard input is a regular file, *ed* shall terminate with a non-zero exit status.

### 13411       **APPLICATION USAGE**

13412       Because of the extremely terse nature of the default error messages, the prudent script writer  
13413       begins the *ed* input commands with an **H** command, so that if any errors do occur at least some  
13414       clue as to the cause is made available.

13415       In previous versions, an obsolescent – option was described. This is no longer specified.  
13416       Applications should use the **-s** option. Using – as a *file* operand now produces unspecified  
13417       results. This allows implementations to continue to support the former required behavior.

### 13418       **EXAMPLES**

13419       None.

### 13420       **RATIONALE**

13421       The initial description of this utility was adapted from the SVID. It contains some features not  
13422       found in Version 7 or BSD-derived systems. Some of the differences between the POSIX and  
13423       BSD *ed* utilities include, but need not be limited to:

- 13424       • The BSD – option does not suppress the '!' prompt after a ! command.
- 13425       • BSD does not support the special meanings of the '%' and '!' characters within a !  
13426       command.
- 13427       • BSD does not support the addresses ';' and ',', '.
- 13428       • BSD allows the command/suffix pairs **pp**, **ll**, and so on, which are unspecified in this volume  
13429       of IEEE Std 1003.1-2001.
- 13430       • BSD does not support the '!' character part of the **e**, **r**, or **w** commands.
- 13431       • A failed **g** command in BSD sets the line number to the last line searched if there are no  
13432       matches.

- 13433     • BSD does not default the *command list* to the **p** command.
- 13434     • BSD does not support the **G**, **h**, **H**, **n**, or **V** commands.
- 13435     • On BSD, if there is no inserted text, the **insert** command changes the current line to the  
13436         referenced line -1; that is, the line before the specified line.
- 13437     • On BSD, the **join** command with only a single address changes the current line to that  
13438         address.
- 13439     • BSD does not support the **P** command; moreover, in BSD it is synonymous with the **p**  
13440         command.
- 13441     • BSD does not support the *undo* of the commands **j**, **m**, **r**, **s**, or **t**.
- 13442     • The Version 7 *ed* command **W**, and the BSD *ed* commands **W**, **wq**, and **z** are not present in this  
13443         volume of IEEE Std 1003.1-2001.

13444 The **-s** option was added to allow the functionality of the now withdrawn – option in a manner  
13445 compatible with the Utility Syntax Guidelines.

13446 In early proposals there was a limit, {ED\_FILE\_MAX}, that described the historical limitations of  
13447 some *ed* utilities in their handling of large files; some of these have had problems with files larger  
13448 than 100 000 bytes. It was this limitation that prompted much of the desire to include a *split*  
13449 command in this volume of IEEE Std 1003.1-2001. Since this limit was removed, this volume of  
13450 IEEE Std 1003.1-2001 requires that implementations document the file size limits imposed by *ed*  
13451 in the conformance document. The limit {ED\_LINE\_MAX} was also removed; therefore, the  
13452 global limit {LINE\_MAX} is used for input and output lines.

13453 The manner in which the **I** command writes non-printable characters was changed to avoid the  
13454 historical backspace-overstrike method. On video display terminals, the overstrike is ambiguous  
13455 because most terminals simply replace overstruck characters, making the **I** format not useful for  
13456 its intended purpose of unambiguously understanding the content of the line. The historical  
13457 backslash escapes were also ambiguous. (The string "a\0011" could represent a line containing  
13458 those six characters or a line containing the three characters 'a', a byte with a binary value of 1,  
13459 and a 1.) In the format required here, a backslash appearing in the line is written as "\\" so that  
13460 the output is truly unambiguous. The method of marking the ends of lines was adopted from the  
13461 *ex* editor and is required for any line ending in <space>s; the '\$' is placed on all lines so that a  
13462 real '\$' at the end of a line cannot be misinterpreted.

13463 Previous versions of this standard allowed for implementations with bytes other than eight bits, 2  
13464 but this has been modified in this version. 2

13465 The description of how a NUL is written was removed. The NUL character cannot be in text  
13466 files, and this volume of IEEE Std 1003.1-2001 should not dictate behavior in the case of  
13467 undefined, erroneous input.

13468 Unlike some of the other editing utilities, the filenames accepted by the **E**, **e**, **R**, and **r** commands  
13469 are not patterns.

13470 Early proposals stated that the **-p** option worked only when standard input was associated with  
13471 a terminal device. This has been changed to conform to historical implementations, thereby  
13472 allowing applications to interpose themselves between a user and the *ed* utility.

13473 The form of the substitute command that uses the **n** suffix was limited in some historical  
13474 documentation (where this was described incorrectly as "backreferencing"). This limit has been  
13475 omitted because there is no reason why an editor processing lines of {LINE\_MAX} length should  
13476 have this restriction. The command **s/x/X/2047** should be able to substitute the 2047th  
13477 occurrence of 'x' on a line.

13478 The use of printing commands with printing suffixes (such as **pn**, **lp**, and so on) was made  
13479 unspecified because BSD-based systems allow this, whereas System V does not.

13480 Some BSD-based systems exit immediately upon receipt of end-of-file if all of the lines in the file  
13481 have been deleted. Since this volume of IEEE Std 1003.1-2001 refers to the **q** command in this  
13482 instance, such behavior is not allowed.

13483 Some historical implementations returned exit status zero even if command errors had occurred;  
13484 this is not allowed by this volume of IEEE Std 1003.1-2001.

13485 Some historical implementations contained a bug that allowed a single period to be entered in  
13486 input mode as <backslash> <period> <newline>. This is not allowed by **ed** because there is no  
13487 description of escaping any of the characters in input mode; backslashes are entered into the  
13488 buffer exactly as typed. The typical method of entering a single period has been to precede it  
13489 with another character and then use the substitute command to delete that character.

13490 It is difficult under some modes of some versions of historical operating system terminal drivers  
13491 to distinguish between an end-of-file condition and terminal disconnect. IEEE Std 1003.1-2001  
13492 does not require implementations to distinguish between the two situations, which permits  
13493 historical implementations of the **ed** utility on historical platforms to conform. Implementations  
13494 are encouraged to distinguish between the two, if possible, and take appropriate action on  
13495 terminal disconnect.

13496 Historically, **ed** accepted a zero address for the **a** and **r** commands in order to insert text at the  
13497 start of the edit buffer. When the buffer was empty the command **.=** returned zero.  
13498 IEEE Std 1003.1-2001 requires conformance to historical practice.

13499 For consistency with the **a** and **r** commands and better user functionality, the **i** and **c** commands  
13500 must also accept an address of 0, in which case **0i** is treated as **1i** and likewise for the **c**  
13501 command.

13502 All of the following are valid addresses:

- |                        |                                                    |
|------------------------|----------------------------------------------------|
| 13503       +++        | Three lines after the current line.                |
| 13504       /pattern/- | One line before the next occurrence of pattern.    |
| 13505       -2         | Two lines before the current line.                 |
| 13506       3 ----- 2  | Line one (note the intermediate negative address). |
| 13507       1 2 3      | Line six.                                          |

13508 Any number of addresses can be provided to commands taking addresses; for example,  
13509 "**1,2,3,4,5p**" prints lines 4 and 5, because two is the greatest valid number of addresses  
13510 accepted by the **print** command. This, in combination with the semicolon delimiter, permits  
13511 users to create commands based on ordered patterns in the file. For example, the command  
13512 "**3;/foo/;+2p**" will display the first line after line 3 that contains the pattern **foo**, plus the next  
13513 two lines. Note that the address "**3;**" must still be evaluated before being discarded, because  
13514 the search origin for the "**/foo/**" command depends on this.

13515 Historically, **ed** disallowed address chains, as discussed above, consisting solely of comma or  
13516 semicolon separators; for example, "**, , ,**" or "**; ; ;**" were considered an error. For consistency of  
13517 address specification, this restriction is removed. The following table lists some of the address  
13518 forms now possible:

13519  
13520  
13521  
13522  
13523  
13524  
13525  
13526  
13527  
13528  
13529  
13530  
13531  
13532  
13533  
13534  
13535  
13536  
13537  
13538  
13539  
13540

|       | <b>Address</b> | <b>Addr1</b> | <b>Addr2</b> | <b>Status</b> | <b>Comment</b> |
|-------|----------------|--------------|--------------|---------------|----------------|
| 13521 | 7 ,            | 7            | 7            | Historical    |                |
| 13522 | 7 , 5 ,        | 5            | 5            | Historical    |                |
| 13523 | 7 , 5 , 9      | 5            | 9            | Historical    |                |
| 13524 | 7 , 9          | 7            | 9            | Historical    |                |
| 13525 | 7 , +          | 7            | 8            | Historical    |                |
| 13526 | ,              | 1            | \$           | Historical    |                |
| 13527 | , 7            | 1            | 7            | Extension     |                |
| 13528 | , ,            | \$           | \$           | Extension     |                |
| 13529 | , ;            | \$           | \$           | Extension     |                |
| 13530 | 7 ;            | 7            | 7            | Historical    |                |
| 13531 | 7 ; 5 ;        | 5            | 5            | Historical    |                |
| 13532 | 7 ; 5 ; 9      | 5            | 9            | Historical    |                |
| 13533 | 7 ; 5 , 9      | 5            | 9            | Historical    |                |
| 13534 | 7 ; \$ ; 4     | \$           | 4            | Historical    |                |
| 13535 | 7 ; 9          | 7            | 9            | Historical    |                |
| 13536 | 7 ; +          | 7            | 8            | Historical    |                |
| 13537 | ;              | .            | \$           | Historical    |                |
| 13538 | ; 7            | .            | 7            | Extension     |                |
| 13539 | ; ;            | \$           | \$           | Extension     |                |
| 13540 | ; ,            | \$           | \$           | Extension     |                |

Valid, but erroneous.

13541 Historically, values could be added to addresses by including them after one or more <blank>s;  
13542 for example, "3 – 5p" wrote the seventh line of the file, and "/foo/ 5" was the same as  
13543 "5 /foo/". However, only absolute values could be added; for example, "5 /foo/" was an  
13544 error. IEEE Std 1003.1-2001 requires conformance to historical practice.

13545 Historically, *ed* accepted the '^' character as an address, in which case it was identical to the  
13546 hyphen character. IEEE Std 1003.1-2001 does not require or prohibit this behavior.

## 13547 FUTURE DIRECTIONS

13548 None.

## 13549 SEE ALSO

13550 Section 1.11 (on page 20), *ex*, *sed*, *sh*, *vi*

## 13551 CHANGE HISTORY

13552 First released in Issue 2.

### 13553 Issue 5

13554 In the OPTIONS section, the meaning of –s and – is clarified.

13555 A second FUTURE DIRECTION is added.

### 13556 Issue 6

13557 The obsolescent single-minus form is removed.

13558 A second APPLICATION USAGE note is added.

13559 The Open Group Corrigendum U025/2 is applied, correcting the description of the Edit section.

13560 The *ed* utility is updated to align with the IEEE P1003.2b draft standard. This includes addition of  
13561 the treatment of the SIGQUIT signal, changes to *ed* addressing, and changes to processing when  
13562 end-of-file is detected and when terminal disconnect is detected.

13563 The normative text is reworded to avoid use of the term “must” for application requirements.

|       |                                                                                                                                                                                                                                                                                      |   |
|-------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---|
| 13564 | IEEE Std 1003.1-2001/Cor 1-2002, item XCU/TC1/D6/22 is applied, adding the text: “Any line modified by the <i>command list</i> shall be unmarked.” to the G command. This change corresponds to a similar change made to the g command in the first version of IEEE Std 1003.1-2001. | 1 |
| 13565 |                                                                                                                                                                                                                                                                                      | 1 |
| 13566 |                                                                                                                                                                                                                                                                                      | 1 |
| 13567 | IEEE Std 1003.1-2001/Cor 2-2004, item XCU/TC2/D6/7 is applied, removing text describing behavior on systems with bytes consisting of more than eight bits.                                                                                                                           | 2 |
| 13568 |                                                                                                                                                                                                                                                                                      | 2 |

**13569 NAME**

13570 env — set the environment for command invocation

**13571 SYNOPSIS**

13572 `env [-i][name=value]... [utility [argument...]]`

**13573 DESCRIPTION**

13574 The *env* utility shall obtain the current environment, modify it according to its arguments, then  
13575 invoke the utility named by the *utility* operand with the modified environment.

13576 Optional arguments shall be passed to *utility*.

13577 If no *utility* operand is specified, the resulting environment shall be written to the standard  
13578 output, with one *name=value* pair per line.

**13579 OPTIONS**

13580 The *env* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section  
13581 12.2, Utility Syntax Guidelines.

13582 The following options shall be supported:

13583 **-i** Invoke *utility* with exactly the environment specified by the arguments; the  
13584 inherited environment shall be ignored completely.

**13585 OPERANDS**

13586 The following operands shall be supported:

13587 *name=value* Arguments of the form *name=value* shall modify the execution environment, and  
13588 shall be placed into the inherited environment before the *utility* is invoked.

13589 *utility* The name of the utility to be invoked. If the *utility* operand names any of the  
13590 special built-in utilities in Section 2.14 (on page 64), the results are undefined.

13591 *argument* A string to pass as an argument for the invoked utility.

**13592 STDIN**

13593 Not used.

**13594 INPUT FILES**

13595 None.

**13596 ENVIRONMENT VARIABLES**

13597 The following environment variables shall affect the execution of *env*:

13598 *LANG* Provide a default value for the internationalization variables that are unset or null.  
13599 (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2,  
13600 Internationalization Variables for the precedence of internationalization variables  
13601 used to determine the values of locale categories.)

13602 *LC\_ALL* If set to a non-empty string value, override the values of all the other  
13603 internationalization variables.

13604 *LC\_CTYPE* Determine the locale for the interpretation of sequences of bytes of text data as  
13605 characters (for example, single-byte as opposed to multi-byte characters in  
13606 arguments).

**13607 *LC\_MESSAGES***

13608 Determine the locale that should be used to affect the format and contents of  
13609 diagnostic messages written to standard error.

13610 XSI *NLSPATH* Determine the location of message catalogs for the processing of *LC\_MESSAGES*.

13611        **PATH**        Determine the location of the *utility*, as described in the Base Definitions volume of  
13612                   IEEE Std 1003.1-2001, Chapter 8, Environment Variables. If *PATH* is specified as a  
13613                   *name=value* operand to *env*, the *value* given shall be used in the search for *utility*.

#### 13614 ASYNCHRONOUS EVENTS

13615                   Default.

#### 13616 STDOUT

13617                   If no *utility* operand is specified, each *name=value* pair in the resulting environment shall be  
13618                   written in the form:

13619                   "%s=%s\n", <*name*>, <*value*>

13620                   If the *utility* operand is specified, the *env* utility shall not write to standard output.

#### 13621 STDERR

13622                   The standard error shall be used only for diagnostic messages.

#### 13623 OUTPUT FILES

13624                   None.

#### 13625 EXTENDED DESCRIPTION

13626                   None.

#### 13627 EXIT STATUS

13628                   If *utility* is invoked, the exit status of *env* shall be the exit status of *utility*; otherwise, the *env*  
13629                   utility shall exit with one of the following values:

13630                   0      The *env* utility completed successfully.

13631                   1–125   An error occurred in the *env* utility.

13632                   126     The utility specified by *utility* was found but could not be invoked.

13633                   127     The utility specified by *utility* could not be found.

#### 13634 CONSEQUENCES OF ERRORS

13635                   Default.

#### 13636 APPLICATION USAGE

13637                   The *command*, *env*, *nice*, *nohup*, *time*, and *xargs* utilities have been specified to use exit code 127 if  
13638                   an error occurs so that applications can distinguish “failure to find a utility” from “invoked  
13639                   utility exited with an error indication”. The value 127 was chosen because it is not commonly  
13640                   used for other meanings; most utilities use small values for “normal error conditions” and the  
13641                   values above 128 can be confused with termination due to receipt of a signal. The value 126 was  
13642                   chosen in a similar manner to indicate that the utility could be found, but not invoked. Some  
13643                   scripts produce meaningful error messages differentiating the 126 and 127 cases. The distinction  
13644                   between exit codes 126 and 127 is based on KornShell practice that uses 127 when all attempts to  
13645                   *exec* the utility fail with [ENOENT], and uses 126 when any attempt to *exec* the utility fails for  
13646                   any other reason.

13647                   Historical implementations of the *env* utility use the *execvp()* or *execvp()* functions defined in the  
13648                   System Interfaces volume of IEEE Std 1003.1-2001 to invoke the specified utility; this provides  
13649                   better performance and keeps users from having to escape characters with special meaning to  
13650                   the shell. Therefore, shell functions, special built-ins, and built-ins that are only provided by the  
13651                   shell are not found.

**13652 EXAMPLES**

13653     The following command:

13654        `env -i PATH=/mybin mygrep xyz myfile`

13655        invokes the command *mygrep* with a new *PATH* value as the only entry in its environment. In  
13656        this case, *PATH* is used to locate *mygrep*, which then must reside in **/mybin**.

**13657 RATIONALE**

13658        As with all other utilities that invoke other utilities, this volume of IEEE Std 1003.1-2001 only  
13659        specifies what *env* does with standard input, standard output, standard error, input files, and  
13660        output files. If a utility is executed, it is not constrained by the specification of input and output  
13661        by *env*.

13662        The **-i** option was added to allow the functionality of the withdrawn – option in a manner  
13663        compatible with the Utility Syntax Guidelines.

13664        Some have suggested that *env* is redundant since the same effect is achieved by:

13665        `name=value ... utility [ argument ... ]`

13666        The example is equivalent to *env* when an environment variable is being added to the  
13667        environment of the command, but not when the environment is being set to the given value.  
13668        The *env* utility also writes out the current environment if invoked without arguments. There is  
13669        sufficient functionality beyond what the example provides to justify inclusion of *env*.

**13670 FUTURE DIRECTIONS**

13671        None.

**13672 SEE ALSO**

13673        Section 2.5 (on page 33), Section 2.14 (on page 64)

**13674 CHANGE HISTORY**

13675        First released in Issue 2.

## 13676 NAME

13677 ex — text editor

## 13678 SYNOPSIS

13679 UP ex [-rR][-s | -v][-c command][-t tagstring][-w size][file ...]

13680

## 13681 DESCRIPTION

13682 The *ex* utility is a line-oriented text editor. There are two other modes of the editor—open and  
13683 visual—in which screen-oriented editing is available. This is described more fully by the **ex open**  
13684 and **visual** commands and in *vi*.

13685 This section uses the term *edit buffer* to describe the current working text. No specific  
13686 implementation is implied by this term. All editing changes are performed on the edit buffer,  
13687 and no changes to it shall affect any file until an editor command writes the file.

13688 Certain terminals do not have all the capabilities necessary to support the complete *ex* definition,  
13689 such as the full-screen editing commands (*visual mode* or *open mode*). When these commands  
13690 cannot be supported on such terminals, this condition shall not produce an error message such  
13691 as “not an editor command” or report a syntax error. The implementation may either accept the  
13692 commands and produce results on the screen that are the result of an unsuccessful attempt to  
13693 meet the requirements of this volume of IEEE Std 1003.1-2001 or report an error describing the  
13694 terminal-related deficiency.

## 13695 OPTIONS

13696 The *ex* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section 12.2,  
13697 Utility Syntax Guidelines.

13698 The following options shall be supported:

13699 **-c command** Specify an initial command to be executed in the first edit buffer loaded from an  
13700 existing file (see the EXTENDED DESCRIPTION section). Implementations may  
13701 support more than a single **-c** option. In such implementations, the specified  
13702 commands shall be executed in the order specified on the command line.

13703 **-r** Recover the named files (see the EXTENDED DESCRIPTION section). Recovery  
13704 information for a file shall be saved during an editor or system crash (for example,  
13705 when the editor is terminated by a signal which the editor can catch), or after the  
13706 use of an **ex preserve** command.

13707 A *crash* in this context is an unexpected failure of the system or utility that requires  
13708 restarting the failed system or utility. A system crash implies that any utilities  
13709 running at the time also crash. In the case of an editor or system crash, the number  
13710 of changes to the edit buffer (since the most recent **preserve** command) that will be  
13711 recovered is unspecified.

13712 If no *file* operands are given and the **-t** option is not specified, all other options, the  
13713 *EXINIT* variable, and any *.exrc* files shall be ignored; a list of all recoverable files  
13714 available to the invoking user shall be written, and the editor shall exit normally  
13715 without further action.

13716 **-R** Set **readonly** edit option.

13717 **-s** Prepare *ex* for batch use by taking the following actions:

- 13718 • Suppress writing prompts and informational (but not diagnostic) messages.
- 13719 • Ignore the value of *TERM* and any implementation default terminal type and  
13720 assume the terminal is a type incapable of supporting open or visual modes;

- 13721 see the **visual** command and the description of *vi*.
- 13722 • Suppress the use of the *EXINIT* environment variable and the reading of any  
13723 *.exrc* file; see the EXTENDED DESCRIPTION section.
- 13724 • Suppress autoindentation, ignoring the value of the **autoindent** edit option.
- 13725 **-t tagstring** Edit the file containing the specified *tagstring*; see *ctags*. The tags feature  
13726 represented by **-t tagstring** and the **tag** command is optional. It shall be provided  
13727 on any system that also provides a conforming implementation of *ctags*; otherwise,  
13728 the use of **-t** produces undefined results. On any system, it shall be an error to  
13729 specify more than a single **-t** option.
- 13730 **-v** Begin in visual mode (see *vi*).
- 13731 **-w size** Set the value of the *window* editor option to *size*.

## 13732 OPERANDS

13733 The following operand shall be supported:

13734 *file* A pathname of a file to be edited.

## 13735 STDIN

13736 The standard input consists of a series of commands and input text, as described in the  
13737 EXTENDED DESCRIPTION section. The implementation may limit each line of standard input  
13738 to a length of {LINE\_MAX}.

13739 If the standard input is not a terminal device, it shall be as if the **-s** option had been specified.

13740 If a read from the standard input returns an error, or if the editor detects an end-of-file condition  
13741 from the standard input, it shall be equivalent to a SIGHUP asynchronous event.

## 13742 INPUT FILES

13743 Input files shall be text files or files that would be text files except for an incomplete last line that  
13744 is not longer than {LINE\_MAX}-1 bytes in length and contains no NUL characters. By default,  
13745 any incomplete last line shall be treated as if it had a trailing <newline>. The editing of other  
13746 forms of files may optionally be allowed by *ex* implementations.

13747 The *.exrc* files and source files shall be text files consisting of *ex* commands; see the EXTENDED  
13748 DESCRIPTION section.

13749 By default, the editor shall read lines from the files to be edited without interpreting any of those  
13750 lines as any form of editor command.

## 13751 ENVIRONMENT VARIABLES

13752 The following environment variables shall affect the execution of *ex*:

13753 **COLUMNS** Override the system-selected horizontal screen size. See the Base Definitions  
13754 volume of IEEE Std 1003.1-2001, Chapter 8, Environment Variables for valid values  
13755 and results when it is unset or null.

13756 **EXINIT** Determine a list of *ex* commands that are executed on editor start-up. See the  
13757 EXTENDED DESCRIPTION section for more details of the initialization phase.

13758 **HOME** Determine a pathname of a directory that shall be searched for an editor start-up  
13759 file named *.exrc*; see the EXTENDED DESCRIPTION section.

13760 **LANG** Provide a default value for the internationalization variables that are unset or null.  
13761 (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2,  
13762 Internationalization Variables for the precedence of internationalization variables  
13763 used to determine the values of locale categories.)

|           |                            |                                                                                                                                                                                                                                                                                                                                                                                                   |
|-----------|----------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 13764     | <i>LC_ALL</i>              | If set to a non-empty string value, override the values of all the other internationalization variables.                                                                                                                                                                                                                                                                                          |
| 13765     |                            |                                                                                                                                                                                                                                                                                                                                                                                                   |
| 13766     | <i>LC_COLLATE</i>          | Determine the locale for the behavior of ranges, equivalence classes, and multi-character collating elements within regular expressions.                                                                                                                                                                                                                                                          |
| 13767     |                            |                                                                                                                                                                                                                                                                                                                                                                                                   |
| 13768     |                            |                                                                                                                                                                                                                                                                                                                                                                                                   |
| 13769     | <i>LC_CTYPE</i>            | Determine the locale for the interpretation of sequences of bytes of text data as characters (for example, single-byte as opposed to multi-byte characters in arguments and input files), the behavior of character classes within regular expressions, the classification of characters as uppercase or lowercase letters, the case conversion of letters, and the detection of word boundaries. |
| 13770     |                            |                                                                                                                                                                                                                                                                                                                                                                                                   |
| 13771     |                            |                                                                                                                                                                                                                                                                                                                                                                                                   |
| 13772     |                            |                                                                                                                                                                                                                                                                                                                                                                                                   |
| 13773     |                            |                                                                                                                                                                                                                                                                                                                                                                                                   |
| 13774     | <i>LC_MESSAGES</i>         | Determine the locale that should be used to affect the format and contents of diagnostic messages written to standard error.                                                                                                                                                                                                                                                                      |
| 13775     |                            |                                                                                                                                                                                                                                                                                                                                                                                                   |
| 13776     |                            |                                                                                                                                                                                                                                                                                                                                                                                                   |
| 13777     | <i>LINES</i>               | Override the system-selected vertical screen size, used as the number of lines in a screenful and the vertical screen size in visual mode. See the Base Definitions volume of IEEE Std 1003.1-2001, Chapter 8, Environment Variables for valid values and results when it is unset or null.                                                                                                       |
| 13778     |                            |                                                                                                                                                                                                                                                                                                                                                                                                   |
| 13779     |                            |                                                                                                                                                                                                                                                                                                                                                                                                   |
| 13780     |                            |                                                                                                                                                                                                                                                                                                                                                                                                   |
| 13781 XSI | <i>NLSPATH</i>             | Determine the location of message catalogs for the processing of <i>LC_MESSAGES</i> .                                                                                                                                                                                                                                                                                                             |
| 13782     | <i>PATH</i>                | Determine the search path for the shell command specified in the <i>ex</i> editor commands <b>!</b> , <b>shell</b> , <b>read</b> , and <b>write</b> , and the open and visual mode command <b>!</b> ; see the description of command search and execution in Section 2.9.1.1 (on page 48).                                                                                                        |
| 13783     |                            |                                                                                                                                                                                                                                                                                                                                                                                                   |
| 13784     |                            |                                                                                                                                                                                                                                                                                                                                                                                                   |
| 13785     | <i>SHELL</i>               | Determine the preferred command line interpreter for use as the default value of the <b>shell</b> edit option.                                                                                                                                                                                                                                                                                    |
| 13786     |                            |                                                                                                                                                                                                                                                                                                                                                                                                   |
| 13787     | <i>TERM</i>                | Determine the name of the terminal type. If this variable is unset or null, an unspecified default terminal type shall be used.                                                                                                                                                                                                                                                                   |
| 13788     |                            |                                                                                                                                                                                                                                                                                                                                                                                                   |
| 13789     | <b>ASYNCHRONOUS EVENTS</b> |                                                                                                                                                                                                                                                                                                                                                                                                   |
| 13790     |                            | The following term is used in this and following sections to specify command and asynchronous event actions:                                                                                                                                                                                                                                                                                      |
| 13791     |                            |                                                                                                                                                                                                                                                                                                                                                                                                   |
| 13792     | <i>complete write</i>      | A complete write is a write of the entire contents of the edit buffer to a file of a type other than a terminal device, or the saving of the edit buffer caused by the user executing the <i>ex preserve</i> command. Writing the contents of the edit buffer to a temporary file that will be removed when the editor exits shall not be considered a complete write.                            |
| 13793     |                            |                                                                                                                                                                                                                                                                                                                                                                                                   |
| 13794     |                            |                                                                                                                                                                                                                                                                                                                                                                                                   |
| 13795     |                            |                                                                                                                                                                                                                                                                                                                                                                                                   |
| 13796     |                            |                                                                                                                                                                                                                                                                                                                                                                                                   |
| 13797     |                            |                                                                                                                                                                                                                                                                                                                                                                                                   |
| 13798     |                            | The following actions shall be taken upon receipt of signals:                                                                                                                                                                                                                                                                                                                                     |
| 13799     | <i>SIGINT</i>              | If the standard input is not a terminal device, <i>ex</i> shall not write the file or return to command or text input mode, and shall exit with a non-zero exit status.                                                                                                                                                                                                                           |
| 13800     |                            |                                                                                                                                                                                                                                                                                                                                                                                                   |
| 13801     |                            | Otherwise, if executing an open or visual text input mode command, <i>ex</i> in receipt of SIGINT shall behave identically to its receipt of the <ESC> character.                                                                                                                                                                                                                                 |
| 13802     |                            |                                                                                                                                                                                                                                                                                                                                                                                                   |
| 13803     |                            | Otherwise:                                                                                                                                                                                                                                                                                                                                                                                        |
| 13804     | 1.                         | If executing an <i>ex</i> text input mode command, all input lines that have been completely entered shall be resolved into the edit buffer, and any partially entered line shall be discarded.                                                                                                                                                                                                   |
| 13805     |                            |                                                                                                                                                                                                                                                                                                                                                                                                   |
| 13806     |                            |                                                                                                                                                                                                                                                                                                                                                                                                   |

- 13807            2. If there is a currently executing command, it shall be aborted and a message  
13808            displayed. Unless otherwise specified by the **ex** or **vi** command descriptions,  
13809            it is unspecified whether any lines modified by the executing command  
13810            appear modified, or as they were before being modified by the executing  
13811            command, in the buffer.
- 13812            If the currently executing command was a motion command, its associated  
13813            command shall be discarded.
- 13814            3. If in open or visual command mode, the terminal shall be alerted.
- 13815            4. The editor shall then return to command mode.
- 13816        **SIGCONT**    The screen shall be refreshed if in open or visual mode.
- 13817        **SIGHUP**     If the edit buffer has been modified since the last complete write, **ex** shall attempt  
13818            to save the edit buffer so that it can be recovered later using the **-r** option or the **ex**  
13819            **recover** command. The editor shall not write the file or return to command or text  
13820            input mode, and shall terminate with a non-zero exit status.
- 13821        **SIGTERM**    Refer to SIGHUP.
- 13822            The action taken for all other signals is unspecified.

### 13823 **STDOUT**

13824            The standard output shall be used only for writing prompts to the user, for informational  
13825            messages, and for writing lines from the file.

### 13826 **STDERR**

13827            The standard error shall be used only for diagnostic messages.

### 13828 **OUTPUT FILES**

13829            The output from **ex** shall be text files.

### 13830 **EXTENDED DESCRIPTION**

13831            Only the **ex** mode of the editor is described in this section. See **vi** for additional editing  
13832            capabilities available in **ex**.

13833            When an error occurs, **ex** shall write a message. If the terminal supports a standout mode (such  
13834            as inverse video), the message shall be written in standout mode. If the terminal does not  
13835            support a standout mode, and the edit option **errorbells** is set, an alert action shall precede the  
13836            error message.

13837            By default, **ex** shall start in command mode, which shall be indicated by a **:** prompt; see the  
13838            **prompt** command. Text input mode can be entered by the **append**, **insert**, or **change** commands;  
13839            it can be exited (and command mode re-entered) by typing a period ('.') alone at the beginning  
13840            of a line.

### 13841 **Initialization in ex and vi**

13842            The following symbols are used in this and following sections to specify locations in the edit  
13843            buffer:

#### 13844 *alternate and current pathnames*

13845            Two pathnames, named **current** and **alternate**, are maintained by the editor. Any **ex**  
13846            commands that take filenames as arguments shall set them as follows:

- 13847            1. If a *file* argument is specified to the **ex edit**, **ex**, or **recover** commands, or if an **ex tag**  
13848            command replaces the contents of the edit buffer.

- 13849        a. If the command replaces the contents of the edit buffer, the current pathname  
13850        shall be set to the *file* argument or the file indicated by the tag, and the alternate  
13851        pathname shall be set to the previous value of the current pathname.  
13852        b. Otherwise, the alternate pathname shall be set to the *file* argument.  
13853     2. If a *file* argument is specified to the **ex next** command:  
13854        a. If the command replaces the contents of the edit buffer, the current pathname  
13855        shall be set to the first *file* argument, and the alternate pathname shall be set to  
13856        the previous value of the current pathname.  
13857        3. If a *file* argument is specified to the **ex file** command, the current pathname shall be set  
13858        to the *file* argument, and the alternate pathname shall be set to the previous value of  
13859        the current pathname.  
13860        4. If a *file* argument is specified to the **ex read** and **write** commands (that is, when  
13861        reading or writing a file, and not to the program named by the **shell** edit option), or a  
13862        *file* argument is specified to the **ex exit** command:  
13863            a. If the current pathname has no value, the current pathname shall be set to the *file*  
13864            argument.  
13865            b. Otherwise, the alternate pathname shall be set to the *file* argument.

13866        If the alternate pathname is set to the previous value of the current pathname when the  
13867        current pathname had no previous value, then the alternate pathname shall have no value  
13868        as a result.

#### 13869        *current line*

13870        The line of the edit buffer referenced by the cursor. Each command description specifies the  
13871        current line after the command has been executed, as the *current line* value. When the edit  
13872        buffer contains no lines, the current line shall be zero; see **Addressing in ex** (on page 361).

#### 13873        *current column*

13874        The current display line column occupied by the cursor. (The columns shall be numbered  
13875        beginning at 1.) Each command description specifies the current column after the command  
13876        has been executed, as the *current column* value. This column is an *ideal* column that is  
13877        remembered over the lifetime of the editor. The actual display line column upon which the  
13878        cursor rests may be different from the current column; see the cursor positioning discussion  
13879        in **Command Descriptions in vi** (on page 989).

#### 13880        *set to non-<blank>*

13881        A description for a current column value, meaning that the current column shall be set to  
13882        the last display line column on which is displayed any part of the first non-<blank> of the  
13883        line. If the line has no non-<blank> non-<newline>s, the current column shall be set to the  
13884        last display line column on which is displayed any part of the last non-<newline> in the  
13885        line. If the line is empty, the current column shall be set to column position 1.

13886        The length of lines in the edit buffer may be limited to {LINE\_MAX} bytes. In open and visual  
13887        mode, the length of lines in the edit buffer may be limited to the number of characters that will  
13888        fit in the display. If either limit is exceeded during editing, an error message shall be written. If  
13889        either limit is exceeded by a line read in from a file, an error message shall be written and the  
13890        edit session may be terminated.

13891        If the editor stops running due to any reason other than a user command, and the edit buffer has  
13892        been modified since the last complete write, it shall be equivalent to a SIGHUP asynchronous  
13893        event. If the system crashes, it shall be equivalent to a SIGHUP asynchronous event.

13894 During initialization (before the first file is copied into the edit buffer or any user commands  
13895 from the terminal are processed) the following shall occur:

13896 1. If the environment variable *EXINIT* is set, the editor shall execute the *ex* commands  
13897 contained in that variable.

13898 2. If the *EXINIT* variable is not set, and all of the following are true:

13899 a. The *HOME* environment variable is not null and not empty.

13900 b. The file *.exrc* in the directory referred to by the *HOME* environment variable:

13901 1. Exists

13902 2. Is owned by the same user ID as the real user ID of the process or the process  
13903 has appropriate privileges

13904 3. Is not writable by anyone other than the owner

13905 the editor shall execute the *ex* commands contained in that file.

13906 3. If and only if all of the following are true:

13907 a. The current directory is not referred to by the *HOME* environment variable.

13908 b. A command in the *EXINIT* environment variable or a command in the *.exrc* file in the  
13909 directory referred to by the *HOME* environment variable sets the editor option *exrc*.

13910 c. The *.exrc* file in the current directory:

13911 1. Exists

13912 2. Is owned by the same user ID as the real user ID of the process, or by one of a  
13913 set of implementation-defined user IDs

13914 3. Is not writable by anyone other than the owner

13915 the editor shall attempt to execute the *ex* commands contained in that file.

13916 Lines in any *.exrc* file that are blank lines shall be ignored. If any *.exrc* file exists, but is not read  
13917 for ownership or permission reasons, it shall be an error.

13918 After the *EXINIT* variable and any *.exrc* files are processed, the first file specified by the user  
13919 shall be edited, as follows:

13920 1. If the user specified the *-t* option, the effect shall be as if the *ex tag* command was entered  
13921 with the specified argument, with the exception that if tag processing does not result in a  
13922 file to edit, the effect shall be as described in step 3. below.

13923 2. Otherwise, if the user specified any command line *file* arguments, the effect shall be as if  
13924 the *ex edit* command was entered with the first of those arguments as its *file* argument.

13925 3. Otherwise, the effect shall be as if the *ex edit* command was entered with a nonexistent  
13926 filename as its *file* argument. It is unspecified whether this action shall set the current  
13927 pathname. In an implementation where this action does not set the current pathname, any  
13928 editor command using the current pathname shall fail until an editor command sets the  
13929 current pathname.

13930 If the *-r* option was specified, the first time a file in the initial argument list or a file specified by  
13931 the *-t* option is edited, if recovery information has previously been saved about it, that  
13932 information shall be recovered and the editor shall behave as if the contents of the edit buffer  
13933 have already been modified. If there are multiple instances of the file to be recovered, the one  
13934 most recently saved shall be recovered, and an informational message that there are previous

13935 versions of the file that can be recovered shall be written. If no recovery information about a file  
13936 is available, an informational message to this effect shall be written, and the edit shall proceed as  
13937 usual.

13938 If the **-c** option was specified, the first time a file that already exists (including a file that might  
13939 not exist but for which recovery information is available, when the **-r** option is specified)  
13940 replaces or initializes the contents of the edit buffer, the current line shall be set to the last line of  
13941 the edit buffer, the current column shall be set to non-<blank>, and the **ex** commands specified  
13942 with the **-c** option shall be executed. In this case, the current line and current column shall not be  
13943 set as described for the command associated with the replacement or initialization of the edit  
13944 buffer contents. However, if the **-t** option or a **tag** command is associated with this action, the **-c**  
13945 option commands shall be executed and then the movement to the tag shall be performed.

13946 The current argument list shall initially be set to the filenames specified by the user on the command line. If no filenames are specified by the user, the current argument list shall be empty.  
13947 If the **-t** option was specified, it is unspecified whether any filename resulting from tag processing shall be prepended to the current argument list. In the case where the filename is added as a prefix to the current argument list, the current argument list reference shall be set to that filename. In the case where the filename is not added as a prefix to the current argument list, the current argument list reference shall logically be located before the first of the filenames specified on the command line (for example, a subsequent **ex next** command shall edit the first filename from the command line). If the **-t** option was not specified, the current argument list reference shall be to the first of the filenames on the command line.

## 13956 Addressing in ex

13957 Addressing in **ex** relates to the current line and the current column; the address of a line is its 1-based line number, the address of a column is its 1-based count from the beginning of the line.  
13958 Generally, the current line is the last line affected by a command. The current line number is the address of the current line. In each command description, the effect of the command on the current line number and the current column is described.

13962 Addresses are constructed as follows:

- 13963 1. The character '.' (period) shall address the current line.
- 13964 2. The character '\$' shall address the last line of the edit buffer.
- 13965 3. The positive decimal number *n* shall address the *n*th line of the edit buffer.
- 13966 4. The address "'x'" refers to the line marked with the mark name character 'x', which shall  
13967 be a lowercase letter from the portable character set or one of the characters ''`' or '''. It  
13968 shall be an error if the line that was marked is not currently present in the edit buffer or the  
13969 mark has not been set. Lines can be marked with the **ex mark** or **k** commands, or the **vi m**  
13970 command.
- 13971 5. A regular expression enclosed by slashes ('//') shall address the first line found by  
13972 searching forwards from the line following the current line toward the end of the edit  
13973 buffer and stopping at the first line for which the line excluding the terminating <newline>  
13974 matches the regular expression. As stated in **Regular Expressions in ex** (on page 391), an  
13975 address consisting of a null regular expression delimited by slashes " // " shall address the  
13976 next line for which the line excluding the terminating <newline> matches the last regular  
13977 expression encountered. In addition, the second slash can be omitted at the end of a  
13978 command line. If the **wrapscan** edit option is set, the search shall wrap around to the  
13979 beginning of the edit buffer and continue up to and including the current line, so that the  
13980 entire edit buffer is searched. Within the regular expression, the sequence "\/" shall  
13981 represent a literal slash instead of the regular expression delimiter.

- 13982        6. A regular expression enclosed in question marks ('?') shall address the first line found by  
13983        searching backwards from the line preceding the current line toward the beginning of the  
13984        edit buffer and stopping at the first line for which the line excluding the terminating  
13985        <newline> matches the regular expression. An address consisting of a null regular  
13986        expression delimited by question marks "???" shall address the previous line for which the  
13987        line excluding the terminating <newline> matches the last regular expression encountered.  
13988        In addition, the second question mark can be omitted at the end of a command line. If the  
13989        **wrapscan** edit option is set, the search shall wrap around from the beginning of the edit  
13990        buffer to the end of the edit buffer and continue up to and including the current line, so  
13991        that the entire edit buffer is searched. Within the regular expression, the sequence "\?"  
13992        shall represent a literal question mark instead of the RE delimiter.
- 13993        7. A plus sign ('+') or a minus sign ('-') followed by a decimal number shall address the  
13994        current line plus or minus the number. A '+' or '-' not followed by a decimal number  
13995        shall address the current line plus or minus 1.

13996        Addresses can be followed by zero or more address offsets, optionally <blank>-separated.  
13997        Address offsets are constructed as follows:

- 13998        1. A '+' or '-' immediately followed by a decimal number shall add (subtract) the  
13999        indicated number of lines to (from) the address. A '+' or '-' not followed by a decimal  
14000        number shall add (subtract) 1 to (from) the address.
- 14001        2. A decimal number shall add the indicated number of lines to the address.

14002        It shall not be an error for an intermediate address value to be less than zero or greater than the  
14003        last line in the edit buffer. It shall be an error for the final address value to be less than zero or  
14004        greater than the last line in the edit buffer.

14005        Commands take zero, one, or two addresses; see the descriptions of *1addr* and *2addr* in  
14006        **Command Descriptions in ex** (on page 368). If more than the required number of addresses are  
14007        provided to a command that requires zero addresses, it shall be an error. Otherwise, if more than  
14008        the required number of addresses are provided to a command, the addresses specified first shall  
14009        be evaluated and then discarded until the maximum number of valid addresses remain.

14010        Addresses shall be separated from each other by a comma (',') or a semicolon (';'). If no  
14011        address is specified before or after a comma or semicolon separator, it shall be as if the address  
14012        of the current line was specified before or after the separator. In the case of a semicolon  
14013        separator, the current line ('.') shall be set to the first address, and only then will the next  
14014        address be calculated. This feature can be used to determine the starting line for forwards and  
14015        backwards searches (see rules 5. and 6.).

14016        A percent sign ('%') shall be equivalent to entering the two addresses "1, \$".

14017        Any delimiting <blank>s between addresses, address separators, or address offsets shall be  
14018        discarded.

#### 14019        **Command Line Parsing in ex**

14020        The following symbol is used in this and following sections to describe parsing behavior:

14021        **escape**        If a character is referred to as "backslash-escaped" or "<control>-V-escaped," it  
14022        shall mean that the character acquired or lost a special meaning by virtue of being  
14023        preceded, respectively, by a backslash or <control>-V character. Unless otherwise  
14024        specified, the escaping character shall be discarded at that time and shall not be  
14025        further considered for any purpose.

14026 Command-line parsing shall be done in the following steps. For each step, characters already  
14027 evaluated shall be ignored; that is, the phrase ‘‘leading character’’ refers to the next character  
14028 that has not yet been evaluated.

- 14029 1. Leading colon characters shall be skipped.
- 14030 2. Leading <blank>s shall be skipped.
- 14031 3. If the leading character is a double-quote character, the characters up to and including the  
14032 next non-backslash-escaped <newline> shall be discarded, and any subsequent characters  
14033 shall be parsed as a separate command.
- 14034 4. Leading characters that can be interpreted as addresses shall be evaluated; see **Addressing**  
14035 **in ex** (on page 361).
- 14036 5. Leading <blank>s shall be skipped.
- 14037 6. If the next character is a vertical-line character or a <newline>:
  - 14038 a. If the next character is a <newline>:
    - 14039 1. If **ex** is in open or visual mode, the current line shall be set to the last address  
14040 specified, if any.
    - 14041 2. Otherwise, if the last command was terminated by a vertical-line character, no  
14042 action shall be taken; for example, the command " | |<newline>" shall  
14043 execute two implied commands, not three.
    - 14044 3. Otherwise, step 6.b. shall apply.
  - 14045 b. Otherwise, the implied command shall be the **print** command. The last #, p, and l  
14046 flags specified to any **ex** command shall be remembered and shall apply to this  
14047 implied command. Executing the **ex number, print, or list** command shall set the  
14048 remembered flags to #, nothing, and l, respectively, plus any other flags specified for  
14049 that execution of the **number, print, or list** command.
- 14050 If **ex** is not currently performing a **global** or v command, and no address or count is  
14051 specified, the current line shall be incremented by 1 before the command is executed.  
14052 If incrementing the current line would result in an address past the last line in the  
14053 edit buffer, the command shall fail, and the increment shall not happen.
- 14054 c. The <newline> or vertical-line character shall be discarded and any subsequent  
14055 characters shall be parsed as a separate command.
- 14056 7. The command name shall be comprised of the next character (if the character is not  
14057 alphabetic), or the next character and any subsequent alphabetic characters (if the  
14058 character is alphabetic), with the following exceptions:
  - 14059 a. Commands that consist of any prefix of the characters in the command name **delete**,  
14060 followed immediately by any of the characters '1', 'p', '+', '−', or '#' shall be  
14061 interpreted as a **delete** command, followed by a <blank>, followed by the characters  
14062 that were not part of the prefix of the **delete** command. The maximum number of  
14063 characters shall be matched to the command name **delete**; for example, "de1" shall  
14064 not be treated as "de" followed by the flag l.
  - 14065 b. Commands that consist of the character 'k', followed by a character that can be  
14066 used as the name of a mark, shall be equivalent to the mark command followed by a  
14067 <blank>, followed by the character that followed the 'k'.
  - 14068 c. Commands that consist of the character 's', followed by characters that could be  
14069 interpreted as valid options to the **s** command, shall be the equivalent of the **s**

14070 command, without any pattern or replacement values, followed by a <blank>,  
 14071 followed by the characters after the 's'.

- 14072 8. The command name shall be matched against the possible command names, and a  
 14073 command name that contains a prefix matching the characters specified by the user shall  
 14074 be the executed command. In the case of commands where the characters specified by the  
 14075 user could be ambiguous, the executed command shall be as follows:

|    |        |    |       |    |       |
|----|--------|----|-------|----|-------|
| a  | append | n  | next  | t  | t     |
| c  | change | p  | print | u  | undo  |
| ch | change | pr | print | un | undo  |
| e  | edit   | r  | read  | v  | v     |
| m  | move   | re | read  | w  | write |
| ma | mark   | s  | s     |    |       |

14082 Implementation extensions with names causing similar ambiguities shall not be checked  
 14083 for a match until all possible matches for commands specified by IEEE Std 1003.1-2001  
 14084 have been checked.

- 14085 9. If the command is a ! command, or if the command is a **read** command followed by zero  
 14086 or more <blank>s and a !, or if the command is a **write** command followed by one or more  
 14087 <blank>s and a !, the rest of the command shall include all characters up to a non-  
 14088 backslash-escaped <newline>. The <newline> shall be discarded and any subsequent  
 14089 characters shall be parsed as a separate ex command.
- 14090 10. Otherwise, if the command is an **edit**, **ex**, or **next** command, or a **visual** command while in  
 14091 open or visual mode, the next part of the command shall be parsed as follows:
- 14092 a. Any '!' character immediately following the command shall be skipped and be part  
 14093 of the command.
- 14094 b. Any leading <blank>s shall be skipped and be part of the command.
- 14095 c. If the next character is a '+', characters up to the first non-backslash-escaped  
 14096 <newline> or non-backslash-escaped <blank> shall be skipped and be part of the  
 14097 command.
- 14098 d. The rest of the command shall be determined by the steps specified in paragraph 12.
- 14099 11. Otherwise, if the command is a **global**, **open**, **s**, or **v** command, the next part of the  
 14100 command shall be parsed as follows:
- 14101 a. Any leading <blank>s shall be skipped and be part of the command.
- 14102 b. If the next character is not an alphanumeric, double-quote, <newline>, backslash, or  
 14103 vertical-line character:
- 14104 1. The next character shall be used as a command delimiter.
- 14105 2. If the command is a **global**, **open**, or **v** command, characters up to the first  
 14106 non-backslash-escaped <newline>, or first non-backslash-escaped delimiter  
 14107 character, shall be skipped and be part of the command.
- 14108 3. If the command is an **s** command, characters up to the first non-backslash-  
 14109 escaped <newline>, or second non-backslash-escaped delimiter character, shall  
 14110 be skipped and be part of the command.
- 14111 c. If the command is a **global** or **v** command, characters up to the first non-backslash-  
 14112 escaped <newline> shall be skipped and be part of the command.

14113                   d. Otherwise, the rest of the command shall be determined by the steps specified in  
14114                   paragraph 12.

14115                 12. Otherwise:

- 14116                 a. If the command was a **map**, **unmap**, **abbreviate**, or **unabbreviate** command,  
14117                 characters up to the first non-<control>-V-escaped <newline>, vertical-line, or  
14118                 double-quote character shall be skipped and be part of the command.
- 14119                 b. Otherwise, characters up to the first non-backslash-escaped <newline>, vertical-line,  
14120                 or double-quote character shall be skipped and be part of the command.
- 14121                 c. If the command was an **append**, **change**, or **insert** command, and the step 12.b.  
14122                 ended at a vertical-line character, any subsequent characters, up to the next non-  
14123                 backslash-escaped <newline> shall be used as input text to the command.
- 14124                 d. If the command was ended by a double-quote character, all subsequent characters,  
14125                 up to the next non-backslash-escaped <newline>, shall be discarded.
- 14126                 e. The terminating <newline> or vertical-line character shall be discarded and any  
14127                 subsequent characters shall be parsed as a separate ex command.

14128     Command arguments shall be parsed as described by the Synopsis and Description of each  
14129     individual ex command. This parsing shall not be <blank>-sensitive, except for the ! argument,  
14130     which must follow the command name without intervening <blank>s, and where it would  
14131     otherwise be ambiguous. For example, *count* and *flag* arguments need not be <blank>-separated  
14132     because "d22p" is not ambiguous, but *file* arguments to the ex **next** command must be  
14133     separated by one or more <blank>s. Any <blank> in command arguments for the **abbreviate**,  
14134     **unabbreviate**, **map**, and **unmap** commands can be <control>-V-escaped, in which case the  
14135     <blank> shall not be used as an argument delimiter. Any <blank> in the command argument for  
14136     any other command can be backslash-escaped, in which case that <blank> shall not be used as  
14137     an argument delimiter.

14138     Within command arguments for the **abbreviate**, **unabbreviate**, **map**, and **unmap** commands,  
14139     any character can be <control>-V-escaped. All such escaped characters shall be treated literally  
14140     and shall have no special meaning. Within command arguments for all other ex commands that  
14141     are not regular expressions or replacement strings, any character that would otherwise have a  
14142     special meaning can be backslash-escaped. Escaped characters shall be treated literally, without  
14143     special meaning as shell expansion characters or '!', '%', and '#' expansion characters. See  
14144     **Regular Expressions in ex** (on page 391) and **Replacement Strings in ex** (on page 391) for  
14145     descriptions of command arguments that are regular expressions or replacement strings.

14146     Non-backslash-escaped '%' characters appearing in *file* arguments to any ex command shall be  
14147     replaced by the current pathname; unescaped '#' characters shall be replaced by the alternate  
14148     pathname. It shall be an error if '%' or '#' characters appear unescaped in an argument and  
14149     their corresponding values are not set.

14150     Non-backslash-escaped '!' characters in the arguments to either the ex ! command or the open  
14151     and visual mode ! command, or in the arguments to the ex **read** command, where the first non-  
14152     <blank> after the command name is a '!' character, or in the arguments to the ex **write**  
14153     command where the command name is followed by one or more <blank>s and the first non-  
14154     <blank> after the command name is a '!' character, shall be replaced with the arguments to the  
14155     last of those three commands as they appeared after all unescaped '%', '#', and '!' characters  
14156     were replaced. It shall be an error if '!' characters appear unescaped in one of these commands  
14157     and there has been no previous execution of one of these commands.

14158     If an error occurs during the parsing or execution of an ex command:

- 14159 • An informational message to this effect shall be written. Execution of the **ex** command shall  
14160 stop, and the cursor (for example, the current line and column) shall not be further modified.
- 14161 • If the **ex** command resulted from a map expansion, all characters from that map expansion  
14162 shall be discarded, except as otherwise specified by the **map** command.
- 14163 • Otherwise, if the **ex** command resulted from the processing of an *EXINIT* environment  
14164 variable, a **.exrc** file, a **:source** command, a **-c** option, or a **+command** specified to an **ex edit**,  
14165 **ex**, **next**, or **visual** command, no further commands from the source of the commands shall  
14166 be executed.
- 14167 • Otherwise, if the **ex** command resulted from the execution of a buffer or a **global** or **v**  
14168 command, no further commands caused by the execution of the buffer or the **global** or **v**  
14169 command shall be executed.
- 14170 • Otherwise, if the **ex** command was not terminated by a <newline>, all characters up to and  
14171 including the next non-backslash-escaped <newline> shall be discarded.

## 14172 Input Editing in ex

14173 The following symbol is used in this and the following sections to specify command actions:

14174 *word* In the POSIX locale, a word consists of a maximal sequence of letters, digits, and  
14175 underscores, delimited at both ends by characters other than letters, digits, or  
14176 underscores, or by the beginning or end of a line or the edit buffer.

14177 When accepting input characters from the user, in either **ex** command mode or **ex** text input  
14178 mode, **ex** shall enable canonical mode input processing, as defined in the System Interfaces  
14179 volume of IEEE Std 1003.1-2001.

14180 If in **ex** text input mode:

1. If the **number** edit option is set, **ex** shall prompt for input using the line number that would  
be assigned to the line if it is entered, in the format specified for the **ex number** command.
2. If the **autoindent** edit option is set, **ex** shall prompt for input using **autoindent** characters,  
as described by the **autoindent** edit option. **autoindent** characters shall follow the line  
number, if any.

14186 If in **ex** command mode:

1. If the **prompt** edit option is set, input shall be prompted for using a single ' : ' character;  
otherwise, there shall be no prompt.

14189 The input characters in the following sections shall have the following effects on the input line.

## 14190 Scroll

14191 *Synopsis:* eof

14192 See the description of the *stty eof* character in *stty*.

14193 If in **ex** command mode:

14194 If the *eof* character is the first character entered on the line, the line shall be evaluated as if it  
14195 contained two characters: a <control>-D and a <newline>.

14196 Otherwise, the *eof* character shall have no special meaning.

- 14197      If in ex text input mode:
- 14198      If the cursor follows an **autoindent** character, the **autoindent** characters in the line shall be modified so that a part of the next text input character will be displayed on the first column in the line after the previous **shiftwidth** edit option column boundary, and the user shall be prompted again for input for the same line.
- 14202      Otherwise, if the cursor follows a '0', which follows an **autoindent** character, and the '0' was the previous text input character, the '0' and all **autoindent** characters in the line shall be discarded, and the user shall be prompted again for input for the same line.
- 14205      Otherwise, if the cursor follows a '^', which follows an **autoindent** character, and the '^' was the previous text input character, the '^' and all **autoindent** characters in the line shall be discarded, and the user shall be prompted again for input for the same line. In addition, the **autoindent** level for the next input line shall be derived from the same line from which the **autoindent** level for the current input line was derived.
- 14210      Otherwise, if there are no **autoindent** or text input characters in the line, the **eof** character shall be discarded.
- 14212      Otherwise, the **eof** character shall have no special meaning.
- 14213      **<newline>**
- 14214      *Synopsis:*    **<newline>**
- 14215                **<control>-J**
- 14216      If in ex command mode:
- 14217      Cause the command line to be parsed; **<control>-J** shall be mapped to the **<newline>** for this purpose.
- 14219      If in ex text input mode:
- 14220      Terminate the current line. If there are no characters other than **autoindent** characters on the line, all characters on the line shall be discarded.
- 14222      Prompt for text input on a new line after the current line. If the **autoindent** edit option is set, an appropriate number of **autoindent** characters shall be added as a prefix to the line as described by the **ex autoindent** edit option.
- 14225      **<backslash>**
- 14226      *Synopsis:*    **<backslash>**
- 14227      Allow the entry of a subsequent **<newline>** or **<control>-J** as a literal character, removing any special meaning that it may have to the editor during text input mode. The backslash character shall be retained and evaluated when the command line is parsed, or retained and included when the input text becomes part of the edit buffer.

14231 <control>-V

14232 *Synopsis:* <control>-V

14233 Allow the entry of any subsequent character as a literal character, removing any special meaning  
14234 that it may have to the editor during text input mode. The <control>-V character shall be  
14235 discarded before the command line is parsed or the input text becomes part of the edit buffer.

14236 If the “literal next” functionality is performed by the underlying system, it is implementation-  
14237 defined whether a character other than <control>-V performs this function.

14238 <control>-W

14239 *Synopsis:* <control>-W

14240 Discard the <control>-W, and the word previous to it in the input line, including any <blank>s  
14241 following the word and preceding the <control>-W. If the “word erase” functionality is  
14242 performed by the underlying system, it is implementation-defined whether a character other  
14243 than <control>-W performs this function.

#### 14244 Command Descriptions in ex

14245 The following symbols are used in this section to represent command modifiers. Some of these  
14246 modifiers can be omitted, in which case the specified defaults shall be used.

14247 *addr* A single line address, given in any of the forms described in **Addressing in ex** (on  
14248 page 361); the default shall be the current line ('.'), unless otherwise specified.

14249 If the line address is zero, it shall be an error, unless otherwise specified in the  
14250 following command descriptions.

14251 If the edit buffer is empty, and the address is specified with a command other than  
14252 '=' , **append**, **insert**, **open**, **put**, **read**, or **visual**, or the address is not zero, it shall be  
14253 an error.

14254 *2addr* Two addresses specifying an inclusive range of lines. If no addresses are specified,  
14255 the default for *2addr* shall be the current line only (".."), unless otherwise  
14256 specified in the following command descriptions. If one address is specified, *2addr*  
14257 shall specify that line only, unless otherwise specified in the following command  
14258 descriptions.

14259 It shall be an error if the first address is greater than the second address.

14260 If the edit buffer is empty, and the two addresses are specified with a command  
14261 other than the !, **write**, **wq**, or **xit** commands, or either address is not zero, it shall  
14262 be an error.

14263 *count* A positive decimal number. If *count* is specified, it shall be equivalent to specifying  
14264 an additional address to the command, unless otherwise specified by the following  
14265 command descriptions. The additional address shall be equal to the last address  
14266 specified to the command (either explicitly or by default) plus *count*-1.

14267 If this would result in an address greater than the last line of the edit buffer, it shall  
14268 be corrected to equal the last line of the edit buffer.

14269 *flags* One or more of the characters '+', '-', '#', 'p', or 'l' (ell). The flag characters  
14270 can be <blank>-separated, and in any order or combination. The characters '#' ,  
14271 'p' , and 'l' shall cause lines to be written in the format specified by the **print**  
14272 command with the specified *flags*.

14273

The lines to be written are as follows:

14274

1. All edit buffer lines written during the execution of the **ex &, ^, list, number, open, print, s, visual, and z** commands shall be written as specified by *flags*.

14276

2. After the completion of an **ex** command with a flag as an argument, the current line shall be written as specified by *flags*, unless the current line was the last line written by the command.

14279

The characters '+' and '-' cause the value of the current line after the execution of the **ex** command to be adjusted by the offset address as described in **Addressing in ex** (on page 361). This adjustment shall occur before the current line is written as described in 2. above.

14283

The default for *flags* shall be none.

14284

### buffer

14285

14286

14287

14288

14289

14290

14291

One of a number of named areas for holding text. The named buffers are specified by the alphanumeric characters of the POSIX locale. There shall also be one "unnamed" buffer. When no buffer is specified for editor commands that use a buffer, the unnamed buffer shall be used. Commands that store text into buffers shall store the text as it was before the command took effect, and shall store text occurring earlier in the file before text occurring later in the file, regardless of how the text region was specified. Commands that store text into buffers shall store the text into the unnamed buffer as well as any specified buffer.

14292

14293

In **ex** commands, buffer names are specified as the name by itself. In **open** or **visual** mode commands the name is preceded by a double quote (' ") character.

14294

14295

14296

14297

If the specified buffer name is an uppercase character, and the buffer contents are to be modified, the buffer shall be appended to rather than being overwritten. If the buffer is not being modified, specifying the buffer name in lowercase and uppercase shall have identical results.

14298

14299

14300

14301

14302

14303

14304

14305

14306

14307

14308

14309

There shall also be buffers named by the numbers 1 through 9. In **open** and **visual** mode, if a region of text including characters from more than a single line is being modified by the **vi c** or **d** commands, the motion character associated with the **c** or **d** commands specifies that the buffer text shall be in line mode, or the commands %, '^, /, ?, (,), N, n, {, or } are used to define a region of text for the **c** or **d** commands, the contents of buffers 1 through 8 shall be moved into the buffer named by the next numerically greater value, the contents of buffer 9 shall be discarded, and the region of text shall be copied into buffer 1. This shall be in addition to copying the text into a user-specified buffer or unnamed buffer, or both. Numeric buffers can be specified as a source buffer for **open** and **visual** mode commands; however, specifying a numeric buffer as the write target of an **open** or **visual** mode command shall have unspecified results.

14310

14311

14312

14313

14314

14315

14316

14317

14318

The text of each buffer shall have the characteristic of being in either line or character mode. Appending text to a non-empty buffer shall set the mode to match the characteristic of the text being appended. Appending text to a buffer shall cause the creation of at least one additional line in the buffer. All text stored into buffers by **ex** commands shall be in line mode. The **ex** commands that use buffers as the source of text specify individually how buffers of different modes are handled. Each **open** or **visual** mode command that uses buffers for any purpose specifies individually the mode of the text stored into the buffer and how buffers of different modes are handled.

14319        *file*      Command text used to derive a pathname. The default shall be the current  
14320        pathname, as defined previously, in which case, if no current pathname has yet  
14321        been established it shall be an error, except where specifically noted in the  
14322        individual command descriptions that follow. If the command text contains any of  
14323        the characters '~', '{', '[', '\*', '?', '\$', '^', ',', '"', and '\', it shall be  
14324        subjected to the process of "shell expansions", as described below; if more than a  
14325        single pathname results and the command expects only one, it shall be an error.

14326        The process of shell expansions in the editor shall be done as follows. The ex utility  
14327        shall pass two arguments to the program named by the shell edit option; the first  
14328        shall be -c, and the second shall be the string "echo" and the command text as a  
14329        single argument. The standard output and standard error of that command shall  
14330        replace the command text.

14331        !        A character that can be appended to the command name to modify its operation,  
14332        as detailed in the individual command descriptions. With the exception of the ex  
14333        **read**, **write**, and ! commands, the '!' character shall only act as a modifier if there  
14334        are no <blank>s between it and the command name.

14335        *remembered search direction*

14336        The vi commands N and n begin searching in a forwards or backwards direction in  
14337        the edit buffer based on a remembered search direction, which is initially unset,  
14338        and is set by the ex **global**, **v**, **s**, and **tag** commands, and the vi / and ? commands.

14339        **Abbreviate**

14340        *Synopsis:*    ab[*abbreviate*][*lhs rhs*]

14341        If *lhs* and *rhs* are not specified, write the current list of abbreviations and do nothing more.

14342        Implementations may restrict the set of characters accepted in *lhs* or *rhs*, except that printable    2  
14343        characters and <blank>s shall not be restricted. Additional restrictions shall be implementation-  
14344        defined.

14345        In both *lhs* and *rhs*, any character may be escaped with a <control>-V, in which case the  
14346        character shall not be used to delimit *lhs* from *rhs*, and the escaping <control>-V shall be  
14347        discarded.

14348        In open and visual text input mode, if a non-word or <ESC> character that is not escaped by a  
14349        <control>-V character is entered after a word character, a check shall be made for a set of  
14350        characters matching *lhs*, in the text input entered during this command. If it is found, the effect  
14351        shall be as if *rhs* was entered instead of *lhs*.

14352        The set of characters that are checked is defined as follows:

- 14353        1. If there are no characters inserted before the word and non-word or <ESC> characters that  
14354        triggered the check, the set of characters shall consist of the word character.
- 14355        2. If the character inserted before the word and non-word or <ESC> characters that triggered  
14356        the check is a word character, the set of characters shall consist of the characters inserted  
14357        immediately before the triggering characters that are word characters, plus the triggering  
14358        word character.
- 14359        3. If the character inserted before the word and non-word or <ESC> characters that triggered  
14360        the check is not a word character, the set of characters shall consist of the characters that  
14361        were inserted before the triggering characters that are neither <blank>s nor word  
14362        characters, plus the triggering word character.

14363 It is unspecified whether the *lhs* argument entered for the **ex abbreviate** and **unabbreviate**  
14364 commands is replaced in this fashion. Regardless of whether or not the replacement occurs, the  
14365 effect of the command shall be as if the replacement had not occurred.

14366 *Current line*: Unchanged.

14367 *Current column*: Unchanged.

## 14368 Append

14369 *Synopsis:* [1addr] a[ppend][!]

14370 Enter **ex** text input mode; the input text shall be placed after the specified line. If line zero is  
14371 specified, the text shall be placed at the beginning of the edit buffer.

14372 This command shall be affected by the **number** and **autoindent** edit options; following the  
14373 command name with ‘!’ shall cause the **autoindent** edit option setting to be toggled for the  
14374 duration of this command only.

14375 *Current line*: Set to the last input line; if no lines were input, set to the specified line, or to the  
14376 first line of the edit buffer if a line of zero was specified, or zero if the edit buffer is empty.

14377 *Current column*: Set to non-<blank>.

## 14378 Arguments

14379 *Synopsis:* ar[gs]

14380 Write the current argument list, with the current argument-list entry, if any, between ‘[’ and  
14381 ‘]’ characters.

14382 *Current line*: Unchanged.

14383 *Current column*: Unchanged.

## 14384 Change

14385 *Synopsis:* [2addr] c[hang]e[!][count]

14386 Enter **ex** text input mode; the input text shall replace the specified lines. The specified lines shall  
14387 be copied into the unnamed buffer, which shall become a line mode buffer.

14388 This command shall be affected by the **number** and **autoindent** edit options; following the  
14389 command name with ‘!’ shall cause the **autoindent** edit option setting to be toggled for the  
14390 duration of this command only.

14391 *Current line*: Set to the last input line; if no lines were input, set to the line before the first  
14392 address, or to the first line of the edit buffer if there are no lines preceding the first address, or to  
14393 zero if the edit buffer is empty.

14394 *Current column*: Set to non-<blank>.

14395       **Change Directory**

14396       *Synopsis:*    `chd[ir][!][directory]`  
14397                  `cd[!][directory]`

14398       Change the current working directory to *directory*.

14399       If no *directory* argument is specified, and the *HOME* environment variable is set to a non-null  
14400       and non-empty value, *directory* shall default to the value named in the *HOME* environment  
14401       variable. If the *HOME* environment variable is empty or is undefined, the default value of  
14402       *directory* is implementation-defined.

14403       If no '!' is appended to the command name, and the edit buffer has been modified since the  
14404       last complete write, and the current pathname does not begin with a '/', it shall be an error.

14405       *Current line:* Unchanged.

14406       *Current column:* Unchanged.

14407       **Copy**

14408       *Synopsis:*    `[2addr] co[py] 1addr [flags]`  
14409                  `[2addr] t 1addr [flags]`

14410       Copy the specified lines after the specified destination line; line zero specifies that the lines shall  
14411       be placed at the beginning of the edit buffer.

14412       *Current line:* Set to the last line copied.

14413       *Current column:* Set to non-<blank>.

14414       **Delete**

14415       *Synopsis:*    `[2addr] d[elete][buffer][count][flags]`

14416       Delete the specified lines into a buffer (defaulting to the unnamed buffer), which shall become a  
14417       line-mode buffer.

14418       Flags can immediately follow the command name; see **Command Line Parsing in ex** (on page  
14419       362).

14420       *Current line:* Set to the line following the deleted lines, or to the last line in the edit buffer if that  
14421       line is past the end of the edit buffer, or to zero if the edit buffer is empty.

14422       *Current column:* Set to non-<blank>.

14423       **Edit**

14424       *Synopsis:*    `e[dit][!][+command][file]`  
14425                  `ex[!][+command][file]`

14426       If no '!' is appended to the command name, and the edit buffer has been modified since the  
14427       last complete write, it shall be an error.

14428       If *file* is specified, replace the current contents of the edit buffer with the current contents of *file*,  
14429       and set the current pathname to *file*. If *file* is not specified, replace the current contents of the  
14430       edit buffer with the current contents of the file named by the current pathname. If for any reason  
14431       the current contents of the file cannot be accessed, the edit buffer shall be empty.

14432       The *+command* option shall be <blank>-delimited; <blank>s within *+command* can be escaped by  
14433       preceding them with a backslash character. The *+command* shall be interpreted as an *ex*  
14434       command immediately after the contents of the edit buffer have been replaced and the current

14435 line and column have been set.

14436 If the edit buffer is empty:

14437 *Current line*: Set to 0.

14438 *Current column*: Set to 1.

14439 Otherwise, if executed while in ex command mode or if the *+command* argument is specified:

14440 *Current line*: Set to the last line of the edit buffer.

14441 *Current column*: Set to non-<blank>.

14442 Otherwise, if *file* is omitted or results in the current pathname:

14443 *Current line*: Set to the first line of the edit buffer.

14444 *Current column*: Set to non-<blank>.

14445 Otherwise, if *file* is the same as the last file edited, the line and column shall be set as follows; if

14446 the file was previously edited, the line and column may be set as follows:

14447 *Current line*: Set to the last value held when that file was last edited. If this value is not a valid

14448 line in the new edit buffer, set to the first line of the edit buffer.

14449 *Current column*: If the current line was set to the last value held when the file was last edited, set

14450 to the last value held when the file was last edited. Otherwise, or if the last value is not a valid

14451 column in the new edit buffer, set to non-<blank>.

14452 Otherwise:

14453 *Current line*: Set to the first line of the edit buffer.

14454 *Current column*: Set to non-<blank>.

14455 **File**

14456 *Synopsis*: f[ile][file]

14457 If a *file* argument is specified, the alternate pathname shall be set to the current pathname, and

14458 the current pathname shall be set to *file*.

14459 Write an informational message. If the file has a current pathname, it shall be included in this

14460 message; otherwise, the message shall indicate that there is no current pathname. If the edit

14461 buffer contains lines, the current line number and the number of lines in the edit buffer shall be

14462 included in this message; otherwise, the message shall indicate that the edit buffer is empty. If

14463 the edit buffer has been modified since the last complete write, this fact shall be included in this

14464 message. If the **readonly** edit option is set, this fact shall be included in this message. The

14465 message may contain other unspecified information.

14466 *Current line*: Unchanged.

14467 *Current column*: Unchanged.

14468       **Global**

14469       *Synopsis:*    [*2addr*] **g[lobal]** /*pattern/ [commands]*  
14470                    [*2addr*] **v** /*pattern/ [commands]*

14471       The optional '!' character after the **global** command shall be the same as executing the **v** command.

14473       If *pattern* is empty (for example, " // ") or not specified, the last regular expression used in the  
14474       editor command shall be used as the *pattern*. The *pattern* can be delimited by slashes (shown in  
14475       the Synopsis), as well as any non-alphanumeric or non-<blank> other than backslash, vertical  
14476       line, double quote, or <newline>.

14477       If no lines are specified, the lines shall default to the entire file.

14478       The **global** and **v** commands are logically two-pass operations. First, mark the lines within the  
14479       specified lines for which the line excluding the terminating <newline> matches (**global**) or does  
14480       not match (**v** or **global!**) the specified pattern. Second, execute the **ex** commands given by  
14481       *commands*, with the current line ('.') set to each marked line. If an error occurs during this  
14482       process, or the contents of the edit buffer are replaced (for example, by the **ex :edit** command) an  
14483       error message shall be written and no more commands resulting from the execution of this  
14484       command shall be processed.

14485       Multiple **ex** commands can be specified by entering multiple commands on a single line using a  
14486       vertical line to delimit them, or one per line, by escaping each <newline> with a backslash.

14487       If no commands are specified:

- 14488       1. If in **ex** command mode, it shall be as if the **print** command were specified.
- 14489       2. Otherwise, no command shall be executed.

14490       For the **append**, **change**, and **insert** commands, the input text shall be included as part of the  
14491       command, and the terminating period can be omitted if the command ends the list of  
14492       commands. The **open** and **visual** commands can be specified as one of the commands, in which  
14493       case each marked line shall cause the editor to enter open or visual mode. If open or visual mode  
14494       is exited using the **vi Q** command, the current line shall be set to the next marked line, and open  
14495       or visual mode reentered, until the list of marked lines is exhausted.

14496       The **global**, **v**, and **undo** commands cannot be used in *commands*. Marked lines may be deleted  
14497       by commands executed for lines occurring earlier in the file than the marked lines. In this case,  
14498       no commands shall be executed for the deleted lines.

14499       If the remembered search direction is not set, the **global** and **v** commands shall set it to forward.

14500       The **autoprint** and **autoindent** edit options shall be inhibited for the duration of the **g** or **v**  
14501       command.

14502       *Current line:* If no commands executed, set to the last marked line. Otherwise, as specified for  
14503       the executed **ex** commands.

14504       *Current column:* If no commands are executed, set to non-<blank>; otherwise, as specified for the  
14505       individual **ex** commands.

14506       **Insert**

14507       *Synopsis:*    [*1addr*] *i[nsert][!]*

14508       Enter **ex** text input mode; the input text shall be placed before the specified line. If the line is zero  
14509       or 1, the text shall be placed at the beginning of the edit buffer.

14510       This command shall be affected by the **number** and **autoindent** edit options; following the  
14511       command name with '!' shall cause the **autoindent** edit option setting to be toggled for the  
14512       duration of this command only.

14513       *Current line:* Set to the last input line; if no lines were input, set to the line before the specified  
14514       line, or to the first line of the edit buffer if there are no lines preceding the specified line, or zero  
14515       if the edit buffer is empty.

14516       *Current column:* Set to non-<blank>.

14517       **Join**

14518       *Synopsis:*    [*2addr*] *j[oin][!][count][flags]*

14519       If *count* is specified:

14520       If no address was specified, the **join** command shall behave as if *2addr* were the current line  
14521       and the current line plus *count* (., . + *count*).

14522       If one address was specified, the **join** command shall behave as if *2addr* were the specified  
14523       address and the specified address plus *count* (*addr,addr+count*).

14524       If two addresses were specified, the **join** command shall behave as if an additional address,  
14525       equal to the last address plus *count* - 1 (*addr1,addr2,addr2+count-1*), was specified.

14526       If this would result in a second address greater than the last line of the edit buffer, it shall be  
14527       corrected to be equal to the last line of the edit buffer.

14528       If no *count* is specified:

14529       If no address was specified, the **join** command shall behave as if *2addr* were the current line  
14530       and the next line (., . + 1).

14531       If one address was specified, the **join** command shall behave as if *2addr* were the specified  
14532       address and the next line (*addr,addr+1*).

14533       Join the text from the specified lines together into a single line, which shall replace the specified  
14534       lines.

14535       If a '!' character is appended to the command name, the **join** shall be without modification of  
14536       any line, independent of the current locale.

14537       Otherwise, in the POSIX locale, set the current line to the first of the specified lines, and then, for  
14538       each subsequent line, proceed as follows:

- 14539       1. Discard leading <space>s from the line to be joined.
- 14540       2. If the line to be joined is now empty, delete it, and skip steps 3 through 5.
- 14541       3. If the current line ends in a <blank>, or the first character of the line to be joined is a ')' character,  
14542       join the lines without further modification.
- 14543       4. If the last character of the current line is a '.', join the lines with two <space>s between  
14544       them.

14545        5. Otherwise, join the lines with a single <space> between them.

14546        *Current line*: Set to the first line specified.

14547        *Current column*: Set to non-<blank>.

14548        **List**

14549        *Synopsis*:     [2addr] l[ist][count][flags]

14550        This command shall be equivalent to the **ex** command:

14551        [2addr] p[rint][count] l[flags]

14552        See **Print** (on page 380).

14553        **Map**

14554        *Synopsis*:     map[!][lhs rhs]

14555        If *lhs* and *rhs* are not specified:

14556        1. If '!' is specified, write the current list of text input mode maps.

14557        2. Otherwise, write the current list of command mode maps.

14558        3. Do nothing more.

14559        Implementations may restrict the set of characters accepted in *lhs* or *rhs*, except that printable  
14560        characters and <blank>s shall not be restricted. Additional restrictions shall be implementation-  
14561        defined. In both *lhs* and *rhs*, any character can be escaped with a <control>-V, in which case the  
14562        character shall not be used to delimit *lhs* from *rhs*, and the escaping <control>-V shall be  
14563        discarded.

14564        If the character '!' is appended to the **map** command name, the mapping shall be effective  
14565        during open or visual text input mode rather than **open** or **visual** command mode. This allows  
14566        *lhs* to have two different **map** definitions at the same time: one for command mode and one for  
14567        text input mode.

14568        For command mode mappings:

14569        When the *lhs* is entered as any part of a **vi** command in open or visual mode (but not as part  
14570        of the arguments to the command), the action shall be as if the corresponding *rhs* had been  
14571        entered.

14572        If any character in the command, other than the first, is escaped using a <control>-V  
14573        character, that character shall not be part of a match to an *lhs*.

14574        It is unspecified whether implementations shall support **map** commands where the *lhs* is  
14575        more than a single character in length, where the first character of the *lhs* is printable.

14576        If *lhs* contains more than one character and the first character is '#', followed by a sequence  
14577        of digits corresponding to a numbered function key, then when this function key is typed it  
14578        shall be mapped to *rhs*. Characters other than digits following a '#' character also represent  
14579        the function key named by the characters in the *lhs* following the '#' and may be mapped to  
14580        *rhs*. It is unspecified how function keys are named or what function keys are supported.

14581        For text input mode mappings:

14582 When the *lhs* is entered as any part of text entered in open or visual text input modes, the  
14583 action shall be as if the corresponding *rhs* had been entered.

14584 If any character in the input text is escaped using a <control>-V character, that character shall  
14585 not be part of a match to an *lhs*.

14586 It is unspecified whether the *lhs* text entered for subsequent **map** or **unmap** commands is  
14587 replaced with the *rhs* text for the purposes of the screen display; regardless of whether or not  
14588 the display appears as if the corresponding *rhs* text was entered, the effect of the command  
14589 shall be as if the *lhs* text was entered.

14590 If only part of the *lhs* is entered, it is unspecified how long the editor will wait for additional,  
14591 possibly matching characters before treating the already entered characters as not matching the  
14592 *lhs*.

14593 The *rhs* characters shall themselves be subject to remapping, unless otherwise specified by the  
14594 **remap** edit option, except that if the characters in *lhs* occur as prefix characters in *rhs*, those  
14595 characters shall not be remapped.

14596 On block-mode terminals, the mapping need not occur immediately (for example, it may occur  
14597 after the terminal transmits a group of characters to the system), but it shall achieve the same  
14598 results as if it occurred immediately.

14599 *Current line*: Unchanged.

14600 *Current column*: Unchanged.

## 14601 **Mark**

14602 *Synopsis:*    [ *1addr* ] *ma[rk]* *character*  
14603                [ *1addr* ] *k* *character*

14604 Implementations shall support *character* values of a single lowercase letter of the POSIX locale  
14605 and the characters ' ' and ' ' ; support of other characters is implementation-defined.

14606 If executing the **vi m** command, set the specified mark to the current line and 1-based numbered  
14607 character referenced by the current column, if any; otherwise, column position 1.

14608 Otherwise, set the specified mark to the specified line and 1-based numbered first non-<blank>  
14609 non-<newline> in the line, if any; otherwise, the last non-<newline> in the line, if any; otherwise,  
14610 column position 1.

14611 The mark shall remain associated with the line until the mark is reset or the line is deleted. If a  
14612 deleted line is restored by a subsequent **undo** command, any marks previously associated with  
14613 the line, which have not been reset, shall be restored as well. Any use of a mark not associated  
14614 with a current line in the edit buffer shall be an error.

14615 The marks ' and ' shall be set as described previously, immediately before the following events  
14616 occur in the editor:

- 14617 1. The use of '\$' as an **ex** address
- 14618 2. The use of a positive decimal number as an **ex** address
- 14619 3. The use of a search command as an **ex** address
- 14620 4. The use of a mark reference as an **ex** address
- 14621 5. The use of the following open and visual mode commands: <control>-], %, ( ), [ ], { }
- 14622 6. The use of the following open and visual mode commands: ', **G**, **H**, **L**, **M**, **z** if the current  
14623 line will change as a result of the command

14624     7. The use of the open and visual mode commands: **/**, **?**, **N**, **,** **n** if the current line or column  
14625     will change as a result of the command

14626     8. The use of the ex mode commands: **z**, **undo**, **global**, **v**

14627     For rules 1., 2., 3., and 4., the ‘ and ’ marks shall not be set if the **ex** command is parsed as  
14628     specified by rule 6.a. in **Command Line Parsing in ex** (on page 362).

14629     For rules 5., 6., and 7., the ‘ and ’ marks shall not be set if the commands are used as motion  
14630     commands in open and visual mode.

14631     For rules 1., 2., 3., 4., 5., 6., 7., and 8., the ‘ and ’ marks shall not be set if the command fails.

14632     The ‘ and ’ marks shall be set as described previously, each time the contents of the edit buffer  
14633     are replaced (including the editing of the initial buffer), if in open or visual mode, or if in **ex**  
14634     mode and the edit buffer is not empty, before any commands or movements (including  
14635     commands or movements specified by the **-c** or **-t** options or the **+command** argument) are  
14636     executed on the edit buffer. If in open or visual mode, the marks shall be set as if executing the **vi**  
14637     **m** command; otherwise, as if executing the **ex mark** command.

14638     When changing from **ex** mode to open or visual mode, if the ‘ and ’ marks are not already set,  
14639     the ‘ and ’ marks shall be set as described previously.

14640     *Current line*: Unchanged.

14641     *Current column*: Unchanged.

## 14642     Move

14643     *Synopsis:*    `[2addr] m[ove] 1addr [flags]`

14644     Move the specified lines after the specified destination line. A destination of line zero specifies  
14645     that the lines shall be placed at the beginning of the edit buffer. It shall be an error if the  
14646     destination line is within the range of lines to be moved.

14647     *Current line*: Set to the last of the moved lines.

14648     *Current column*: Set to non-<blank>.

## 14649     Next

14650     *Synopsis:*    `n[ext][!][+command][file ...]`

14651     If no ‘ ! ’ is appended to the command name, and the edit buffer has been modified since the  
14652     last complete write, it shall be an error, unless the file is successfully written as specified by the  
14653     **autowrite** option.

14654     If one or more files is specified:

- 14655       1. Set the argument list to the specified filenames.
- 14656       2. Set the current argument list reference to be the first entry in the argument list.
- 14657       3. Set the current pathname to the first filename specified.

14658     Otherwise:

- 14659       1. It shall be an error if there are no more filenames in the argument list after the filename  
14660         currently referenced.
- 14661       2. Set the current pathname and the current argument list reference to the filename after the  
14662         filename currently referenced in the argument list.

14663 Replace the contents of the edit buffer with the contents of the file named by the current  
14664 pathname. If for any reason the contents of the file cannot be accessed, the edit buffer shall be  
14665 empty.

14666 This command shall be affected by the **autowrite** and **writeany** edit options.

14667 The *+command* option shall be <blank>-delimited; <blank>s can be escaped by preceding them  
14668 with a backslash character. The *+command* shall be interpreted as an **ex** command immediately  
14669 after the contents of the edit buffer have been replaced and the current line and column have  
14670 been set.

14671 *Current line*: Set as described for the **edit** command.

14672 *Current column*: Set as described for the **edit** command.

### 14673 **Number**

14674 *Synopsis:*    [2addr] nu[mber][count][flags]  
14675                [2addr] #[count][flags]

14676 These commands shall be equivalent to the **ex** command:

14677    [2addr] p[rint][count] #[flags]

14678 See **Print** (on page 380).

### 14679 **Open**

14680 *Synopsis:*    [1addr] o[pen] /pattern/ [flags]

14681 This command need not be supported on block-mode terminals or terminals with insufficient  
14682 capabilities. If standard input, standard output, or standard error are not terminal devices, the  
14683 results are unspecified.

14684 Enter open mode.

14685 The trailing delimiter can be omitted from *pattern* at the end of the command line. If *pattern* is  
14686 empty (for example, " / ") or not specified, the last regular expression used in the editor shall be  
14687 used as the pattern. The pattern can be delimited by slashes (shown in the Synopsis), as well as  
14688 any alphanumeric, or non-<blank> other than backslash, vertical line, double quote, or  
14689 <newline>.

14690 *Current line*: Set to the specified line.

14691 *Current column*: Set to non-<blank>.

### 14692 **Preserve**

14693 *Synopsis:*    pre[serve]

14694 Save the edit buffer in a form that can later be recovered by using the **-r** option or by using the **ex**  
14695 **recover** command. After the file has been preserved, a mail message shall be sent to the user.  
14696 This message shall be readable by invoking the **mailx** utility. The message shall contain the name  
14697 of the file, the time of preservation, and an **ex** command that could be used to recover the file.  
14698 Additional information may be included in the mail message.

14699 *Current line*: Unchanged.

14700 *Current column*: Unchanged.

14701       **Print**

14702       *Synopsis:*    [*2addr*] *p[rint][count][flags]*

14703       Write the addressed lines. The behavior is unspecified if the number of columns on the display is  
14704       less than the number of columns required to write any single character in the lines being written.

14705       Non-printable characters, except for the <tab>, shall be written as implementation-defined  
14706       multi-character sequences.

14707       If the # flag is specified or the **number** edit option is set, each line shall be preceded by its line  
14708       number in the following format:

14709       "%"dΔΔ", <line number>

14710       If the **I** flag is specified or the **list** edit option is set:

- 14711       1. The characters listed in the Base Definitions volume of IEEE Std 1003.1-2001, Table 5-1,  
14712       Escape Sequences and Associated Actions shall be written as the corresponding escape  
14713       sequence.
- 14714       2. Non-printable characters not in the Base Definitions volume of IEEE Std 1003.1-2001, Table  
14715       5-1, Escape Sequences and Associated Actions shall be written as one three-digit octal  
14716       number (with a preceding backslash) for each byte in the character (most significant byte  
14717       first).
- 14718       3. The end of each line shall be marked with a '\$', and literal '\$' characters within the line  
14719       shall be written with a preceding backslash.

2

14720       Long lines shall be folded; the length at which folding occurs is unspecified, but should be  
14721       appropriate for the output terminal, considering the number of columns of the terminal.

14722       If a line is folded, and the **I** flag is not specified and the **list** edit option is not set, it is unspecified  
14723       whether a multi-column character at the folding position is separated; it shall not be discarded.

14724       *Current line:* Set to the last written line.

14725       *Current column:* Unchanged if the current line is unchanged; otherwise, set to non-<blank>.

14726       **Put**

14727       *Synopsis:*    [*1addr*] *pu[t][buffer]*

14728       Append text from the specified buffer (by default, the unnamed buffer) to the specified line; line  
14729       zero specifies that the text shall be placed at the beginning of the edit buffer. Each portion of a  
14730       line in the buffer shall become a new line in the edit buffer, regardless of the mode of the buffer.

14731       *Current line:* Set to the last line entered into the edit buffer.

14732       *Current column:* Set to non-<blank>.

14733       **Quit**

14734       *Synopsis:*    *q[uit][!]*

14735       If no '!' is appended to the command name:

- 14736       1. If the edit buffer has been modified since the last complete write, it shall be an error.
- 14737       2. If there are filenames in the argument list after the filename currently referenced, and the  
14738       last command was not a **quit**, **wq**, **xit**, or **ZZ** (see **Exit** (on page 1023)) command, it shall be  
14739       an error.

14740 Otherwise, terminate the editing session.

### 14741 **Read**

14742 *Synopsis:* [ *laddr*] r[ead][!][*file*]

14743 If '!' is not the first non-<blank> to follow the command name, a copy of the specified file shall  
14744 be appended into the edit buffer after the specified line; line zero specifies that the copy shall be  
14745 placed at the beginning of the edit buffer. The number of lines and bytes read shall be written. If  
14746 no *file* is named, the current pathname shall be the default. If there is no current pathname, then  
14747 *file* shall become the current pathname. If there is no current pathname or *file* operand, it shall be  
14748 an error. Specifying a *file* that is not of type regular shall have unspecified results.

14749 Otherwise, if *file* is preceded by '!', the rest of the line after the '!' shall have '%', '#', and  
14750 '!' characters expanded as described in **Command Line Parsing in ex** (on page 362).

14751 The **ex** utility shall then pass two arguments to the program named by the shell edit option; the  
14752 first shall be -c and the second shall be the expanded arguments to the **read** command as a  
14753 single argument. The standard input of the program shall be set to the standard input of the **ex**  
14754 program when it was invoked. The standard error and standard output of the program shall be  
14755 appended into the edit buffer after the specified line.

14756 Each line in the copied file or program output (as delimited by <newline>s or the end of the file  
14757 or output if it is not immediately preceded by a <newline>), shall be a separate line in the edit  
14758 buffer. Any occurrences of <carriage-return> and <newline> pairs in the output shall be treated  
14759 as single <newline>s.

14760 The special meaning of the '!' following the **read** command can be overridden by escaping it  
14761 with a backslash character.

14762 *Current line:* If no lines are added to the edit buffer, unchanged. Otherwise, if in open or visual  
14763 mode, set to the first line entered into the edit buffer. Otherwise, set to the last line entered into  
14764 the edit buffer.

14765 *Current column:* Set to non-<blank>.

### 14766 **Recover**

14767 *Synopsis:* rec[over][!][*file*]

14768 If no '!' is appended to the command name, and the edit buffer has been modified since the  
14769 last complete write, it shall be an error.

14770 If no *file* operand is specified, then the current pathname shall be used. If there is no current  
14771 pathname or *file* operand, it shall be an error.

14772 If no recovery information has previously been saved about *file*, the **recover** command shall  
14773 behave identically to the **edit** command, and an informational message to this effect shall be  
14774 written.

14775 Otherwise, set the current pathname to *file*, and replace the current contents of the edit buffer  
14776 with the recovered contents of *file*. If there are multiple instances of the file to be recovered, the  
14777 one most recently saved shall be recovered, and an informational message that there are  
14778 previous versions of the file that can be recovered shall be written. The editor shall behave as if  
14779 the contents of the edit buffer have already been modified.

14780 *Current file:* Set as described for the **edit** command.

14781 *Current column:* Set as described for the **edit** command.

14782       **Rewind**

14783       *Synopsis:*    `rew[nd][!]`

14784       If no '!' is appended to the command name, and the edit buffer has been modified since the  
14785       last complete write, it shall be an error, unless the file is successfully written as specified by the  
14786       **autowrite** option.

14787       If the argument list is empty, it shall be an error.

14788       The current argument list reference and the current pathname shall be set to the first filename in  
14789       the argument list.

14790       Replace the contents of the edit buffer with the contents of the file named by the current  
14791       pathname. If for any reason the contents of the file cannot be accessed, the edit buffer shall be  
14792       empty.

14793       This command shall be affected by the **autowrite** and **writeany** edit options.

14794       *Current line:* Set as described for the **edit** command.

14795       *Current column:* Set as described for the **edit** command.

14796       **Set**

14797       *Synopsis:*    `se[t][option[=value]] ...[nooption ...][option? ...][all]`

14798       When no arguments are specified, write the value of the **term** edit option and those options  
14799       whose values have been changed from the default settings; when the argument *all* is specified,  
14800       write all of the option values.

14801       Giving an option name followed by the character '?' shall cause the current value of that  
14802       option to be written. The '?' can be separated from the option name by zero or more <blank>s.  
14803       The '?' shall be necessary only for Boolean valued options. Boolean options can be given values  
14804       by the form **set** *option* to turn them on or **set** **nooption** to turn them off; string and numeric  
14805       options can be assigned by the form **set** *option*=*value*. Any <blank>s in strings can be included  
14806       as is by preceding each <blank> with an escaping backslash. More than one option can be set or  
14807       listed by a single **set** command by specifying multiple arguments, each separated from the next  
14808       by one or more <blank>s.

14809       See **Edit Options in ex** (on page 392) for details about specific options.

14810       *Current line:* Unchanged.

14811       *Current column:* Unchanged.

14812       **Shell**

14813       *Synopsis:*    `sh[ell]`

14814       Invoke the program named in the **shell** edit option with the single argument **-i** (interactive  
14815       mode). Editing shall be resumed when the program exits.

14816       *Current line:* Unchanged.

14817       *Current column:* Unchanged.

14818       **Source**

14819       *Synopsis:*    `so[urce] file`

14820       Read and execute *ex* commands from *file*. Lines in the file that are blank lines shall be ignored.

14821       *Current line:* As specified for the individual *ex* commands.

14822       *Current column:* As specified for the individual *ex* commands.

14823       **Substitute**

14824       *Synopsis:*    `[2addr] s[ubstitute][/pattern/repl/[options][count][flags]]`  
                   `[2addr] &[options][count][flags]`  
                   `[2addr] ~[options][count][flags]`

14827       Replace the first instance of the pattern *pattern* by the string *repl* on each specified line. (See  
 14828       **Regular Expressions in ex** (on page 391) and **Replacement Strings in ex** (on page 391).) Any  
 14829       non-alphabetic, non-<blank> delimiter other than '\', '|', double quote, or <newline> can be  
 14830       used instead of '/'. Backslash characters can be used to escape delimiters, backslash  
 14831       characters, and other special characters.

14832       The trailing delimiter can be omitted from *pattern* or from *repl* at the end of the command line. If  
 14833       both *pattern* and *repl* are not specified or are empty (for example, " // "), the last **s** command  
 14834       shall be repeated. If only *pattern* is not specified or is empty, the last regular expression used in  
 14835       the editor shall be used as the pattern. If only *repl* is not specified or is empty, the pattern shall be  
 14836       replaced by nothing. If the entire replacement pattern is '%', the last replacement pattern to an  
 14837       **s** command shall be used.

14838       Entering a <carriage-return> in *repl* (which requires an escaping backslash in *ex* mode and an  
 14839       escaping <control>-V in open or *vi* mode) shall split the line at that point, creating a new line in  
 14840       the edit buffer. The <carriage-return> shall be discarded.

14841       If *options* includes the letter 'g' (**global**), all non-overlapping instances of the pattern in the line  
 14842       shall be replaced.

14843       If *options* includes the letter 'c' (**confirm**), then before each substitution the line shall be written;  
 14844       the written line shall reflect all previous substitutions. On the following line, <space>s shall be  
 14845       written beneath the characters from the line that are before the *pattern* to be replaced, and '^'  
 14846       characters written beneath the characters included in the *pattern* to be replaced. The *ex* utility  
 14847       shall then wait for a response from the user. An affirmative response shall cause the substitution  
 14848       to be done, while any other input shall not make the substitution. An affirmative response shall  
 14849       consist of a line with the affirmative response (as defined by the current locale) at the beginning  
 14850       of the line. This line shall be subject to editing in the same way as the *ex* command line.

14851       If interrupted (see the ASYNCHRONOUS EVENTS section), any modifications confirmed by the  
 14852       user shall be preserved in the edit buffer after the interrupt.

14853       If the remembered search direction is not set, the **s** command shall set it to forward.

14854       In the second Synopsis, the **&** command shall repeat the previous substitution, as if the **&**  
 14855       command were replaced by:

14856       `s/pattern/repl/`

14857       where *pattern* and *repl* are as specified in the previous **s**, **&**, or **~** command.

14858       In the third Synopsis, the **~** command shall repeat the previous substitution, as if the '~~' were  
 14859       replaced by:

14860        *s/pattern/repl/*  
14861        where *pattern* shall be the last regular expression specified to the editor, and *repl* shall be from  
14862        the previous substitution (including & and ~) command.

14863        These commands shall be affected by the *LC\_MESSAGES* environment variable.

14864        *Current line*: Set to the last line in which a substitution occurred, or, unchanged if no  
14865        substitution occurred.

14866        *Current column*: Set to non-<blank>.

### 14867        **Suspend**

14868        *Synopsis*:     *su[spend][!]*  
14869              *st[op][!]*

14870        Allow control to return to the invoking process; *ex* shall suspend itself as if it had received the  
14871        SIGTSTP signal. The suspension shall occur only if job control is enabled in the invoking shell  
14872        (see the description of *set -m*).

14873        These commands shall be affected by the **autowrite** and **writeany** edit options.

14874        The current **susp** character (see *stty*) shall be equivalent to the **suspend** command.

### 14875        **Tag**

14876        *Synopsis*:     *ta[g][!]* *tagstring*

14877        The results are unspecified if the format of a tags file is not as specified by the *ctags* utility (see  
14878        *ctags*) description.

14879        The **tag** command shall search for *tagstring* in the tag files referred to by the **tag** edit option, in  
14880        the order they are specified, until a reference to *tagstring* is found. Files shall be searched from  
14881        beginning to end. If no reference is found, it shall be an error and an error message to this effect  
14882        shall be written. If the reference is not found, or if an error occurs while processing a file referred  
14883        to in the **tag** edit option, it shall be an error, and an error message shall be written at the first  
14884        occurrence of such an error.

14885        Otherwise, if the tags file contained a pattern, the pattern shall be treated as a regular expression  
14886        used in the editor; for example, for the purposes of the **s** command.

14887        If the *tagstring* is in a file with a different name than the current pathname, set the current  
14888        pathname to the name of that file, and replace the contents of the edit buffer with the contents of  
14889        that file. In this case, if no '!' is appended to the command name, and the edit buffer has been  
14890        modified since the last complete write, it shall be an error, unless the file is successfully written  
14891        as specified by the **autowrite** option.

14892        This command shall be affected by the **autowrite**, **tag**, **taglength**, and **writeany** edit options.

14893        *Current line*: If the tags file contained a line number, set to that line number. If the line number is  
14894        larger than the last line in the edit buffer, an error message shall be written and the current line  
14895        shall be set as specified for the **edit** command.

14896        If the tags file contained a pattern, set to the first occurrence of the pattern in the file. If no  
14897        matching pattern is found, an error message shall be written and the current line shall be set as  
14898        specified for the **edit** command.

14899        *Current column*: If the tags file contained a line-number reference and that line-number was not  
14900        larger than the last line in the edit buffer, or if the tags file contained a pattern and that pattern  
14901        was found, set to non-<blank>. Otherwise, set as specified for the **edit** command.

14902       **Unabbreviate**

14903     *Synopsis:*    una[bbrev] *lhs*

14904     If *lhs* is not an entry in the current list of abbreviations (see **Abbreviate** (on page 370)), it shall be  
14905     an error. Otherwise, delete *lhs* from the list of abbreviations.

14906     *Current line:* Unchanged.

14907     *Current column:* Unchanged.

14908       **Undo**

14909     *Synopsis:*    u[ndo]

14910     Reverse the changes made by the last command that modified the contents of the edit buffer,  
14911     including **undo**. For this purpose, the **global**, **v**, **open**, and **visual** commands, and commands  
14912     resulting from buffer executions and mapped character expansions, are considered single  
14913     commands.

14914     If no action that can be undone preceded the **undo** command, it shall be an error.

14915     If the **undo** command restores lines that were marked, the mark shall also be restored unless it  
14916     was reset subsequent to the deletion of the lines.

14917     *Current line:*

- 14918     1. If lines are added or changed in the file, set to the first line added or changed.
- 14919     2. Set to the line before the first line deleted, if it exists.
- 14920     3. Set to 1 if the edit buffer is not empty.
- 14921     4. Set to zero.

14922     *Current column:* Set to non-<blank>.

14923       **Unmap**

14924     *Synopsis:*    unm[ap][!] *lhs*

14925     If '!' is appended to the command name, and if *lhs* is not an entry in the list of text input mode  
14926     map definitions, it shall be an error. Otherwise, delete *lhs* from the list of text input mode map  
14927     definitions.

14928     If no '!' is appended to the command name, and if *lhs* is not an entry in the list of command  
14929     mode map definitions, it shall be an error. Otherwise, delete *lhs* from the list of command mode  
14930     map definitions.

14931     *Current line:* Unchanged.

14932     *Current column:* Unchanged.

14933       **Version**

14934     *Synopsis:*    ve[rSION]

14935     Write a message containing version information for the editor. The format of the message is  
14936     unspecified.

14937     *Current line:* Unchanged.

14938     *Current column:* Unchanged.

14939      **Visual**

14940      *Synopsis:*    [1addr] vi[ual][type][count][flags]

14941      If ex is currently in open or visual mode, the Synopsis and behavior of the visual command shall  
14942      be the same as the **edit** command, as specified by **Edit** (on page 372).

14943      Otherwise, this command need not be supported on block-mode terminals or terminals with  
14944      insufficient capabilities. If standard input, standard output, or standard error are not terminal  
14945      devices, the results are unspecified.

14946      If *count* is specified, the value of the **window** edit option shall be set to *count* (as described in  
14947      **window** (on page 398)). If the '^' type character was also specified, the **window** edit option  
14948      shall be set before being used by the type character.

14949      Enter visual mode. If *type* is not specified, it shall be as if a *type* of '+' was specified. The *type*  
14950      shall cause the following effects:

- + Place the beginning of the specified line at the top of the display.
- Place the end of the specified line at the bottom of the display.
- . Place the beginning of the specified line in the middle of the display.
- ^ If the specified line is less than or equal to the value of the **window** edit option, set the line  
to 1; otherwise, decrement the line by the value of the **window** edit option minus 1. Place  
the beginning of this line as close to the bottom of the displayed lines as possible, while still  
displaying the value of the **window** edit option number of lines.

14958      *Current line*: Set to the specified line.

14959      *Current column*: Set to non-<blank>.

14960      **Write**

14961      *Synopsis:*    [2addr] w[rite][!][>>][file]  
14962                [2addr] w[rite][!][file]  
14963                [2addr] wq[!][>>][file]

14964      If no lines are specified, the lines shall default to the entire file.

14965      The command **wq** shall be equivalent to a **write** command followed by a **quit** command; **wq!**  
14966      shall be equivalent to **write!** followed by **quit**. In both cases, if the **write** command fails, the  
14967      **quit** shall not be attempted.

14968      If the command name is not followed by one or more <blank>s, or *file* is not preceded by a '!'  
14969      character, the **write** shall be to a file.

1. If the >> argument is specified, and the file already exists, the lines shall be appended to  
the file instead of replacing its contents. If the >> argument is specified, and the file does  
not already exist, it is unspecified whether the write shall proceed as if the >> argument  
had not been specified or if the write shall fail.
2. If the **readonly** edit option is set (see **readonly** (on page 395)), the **write** shall fail.
3. If *file* is specified, and is not the current pathname, and the file exists, the **write** shall fail.
4. If *file* is not specified, the current pathname shall be used. If there is no current pathname,  
the **write** command shall fail.
5. If the current pathname is used, and the current pathname has been changed by the **file** or  
**read** commands, and the file exists, the **write** shall fail. If the **write** is successful,

14980 subsequent **writes** shall not fail for this reason (unless the current pathname is changed  
14981 again).

14982 6. If the whole edit buffer is not being written, and the file to be written exists, the **write** shall  
14983 fail.

14984 For rules 1., 2., 4., and 5., the **write** can be forced by appending the character '!' to the  
14985 command name.

14986 For rules 2., 4., and 5., the **write** can be forced by setting the **writeany** edit option.

14987 Additional, implementation-defined tests may cause the **write** to fail.

14988 If the edit buffer is empty, a file without any contents shall be written.

14989 An informational message shall be written noting the number of lines and bytes written.

14990 Otherwise, if the command is followed by one or more <blank>s, and the file is preceded by  
14991 '!', the rest of the line after the '!' shall have '%', '#', and '!' characters expanded as  
14992 described in **Command Line Parsing in ex** (on page 362).

14993 The **ex** utility shall then pass two arguments to the program named by the **shell** edit option; the  
14994 first shall be **-c** and the second shall be the expanded arguments to the **write** command as a  
14995 single argument. The specified lines shall be written to the standard input of the command. The  
14996 standard error and standard output of the program, if any, shall be written as described for the  
14997 **print** command. If the last character in that output is not a <newline>, a <newline> shall be  
14998 written at the end of the output.

14999 The special meaning of the '!' following the **write** command can be overridden by escaping it  
15000 with a backslash character.

15001 *Current line:* Unchanged.

15002 *Current column:* Unchanged.

### 15003 **Write and Exit**

15004 *Synopsis:* [2addr] x[it][!][file]

15005 If the edit buffer has not been modified since the last complete **write**, **xit** shall be equivalent to  
15006 the **quit** command, or if a '!' is appended to the command name, to **quit!**.

15007 Otherwise, **xit** shall be equivalent to the **wq** command, or if a '!' is appended to the command  
15008 name, to **wq!**.

15009 *Current line:* Unchanged.

15010 *Current column:* Unchanged.

### 15011 **Yank**

15012 *Synopsis:* [2addr] ya[nk][buffer][count]

15013 Copy the specified lines to the specified buffer (by default, the unnamed buffer), which shall  
15014 become a line-mode buffer.

15015 *Current line:* Unchanged.

15016 *Current column:* Unchanged.

15017       **Adjust Window**

15018       *Synopsis:*    [*laddr*] [*z*!] [*type* ...] [*count*] [*flags*]

15019       If no line is specified, the current line shall be the default; if *type* is omitted as well, the current  
15020       line value shall first be incremented by 1. If incrementing the current line would cause it to be greater  
15021       than the last line in the edit buffer, it shall be an error.

15022       If there are <blank>s between the *type* argument and the preceding **z** command name or optional  
15023       '!' character, it shall be an error.

15024       If *count* is specified, the value of the **window** edit option shall be set to *count* (as described in  
15025       **window** (on page 398)). If *count* is omitted, it shall default to 2 times the value of the **scroll** edit  
15026       option, or if ! was specified, the number of lines in the display minus 1.

15027       If *type* is omitted, then *count* lines starting with the specified line shall be written. Otherwise,  
15028       *count* lines starting with the line specified by the *type* argument shall be written.

15029       The *type* argument shall change the lines to be written. The possible values of *type* are as follows:

15030       – The specified line shall be decremented by the following value:

15031              ((number of '--' characters) x *count*) -1)

15032       If the calculation would result in a number less than 1, it shall be an error. Write lines from  
15033       the edit buffer, starting at the new value of line, until *count* lines or the last line in the edit  
15034       buffer has been written.

15035       + The specified line shall be incremented by the following value:

15036              (((number of '+-' characters) -1) x *count*) +1

15037       If the calculation would result in a number greater than the last line in the edit buffer, it  
15038       shall be an error. Write lines from the edit buffer, starting at the new value of line, until  
15039       *count* lines or the last line in the edit buffer has been written.

15040       =,. If more than a single '.' or '=' is specified, it shall be an error. The following steps shall be  
15041       taken:

15042              1. If *count* is zero, nothing shall be written.

15043              2. Write as many of the *N* lines before the current line in the edit buffer as exist. If *count*  
15044       or '!' was specified, *N* shall be:

15045                  (*count* -1) /2

15046              Otherwise, *N* shall be:

15047                  (*count* -3) /2

15048              If *N* is a number less than 3, no lines shall be written.

15049              3. If '=' was specified as the type character, write a line consisting of the smaller of the  
15050       number of columns in the display divided by two, or 40 '-' characters.

15051              4. Write the current line.

15052              5. Repeat step 3.

15053              6. Write as many of the *N* lines after the current line in the edit buffer as exist. *N* shall be  
15054       defined as in step 2. If *N* is a number less than 3, no lines shall be written. If *count* is  
15055       less than 3, no lines shall be written.

- 15056        ^ The specified line shall be decremented by the following value:  
15057              (((number of ``^'' characters) +1) x count) -1  
15058        If the calculation would result in a number less than 1, it shall be an error. Write lines from  
15059        the edit buffer, starting at the new value of line, until *count* lines or the last line in the edit  
15060        buffer has been written.  
15061        *Current line*: Set to the last line written, unless the type is =, in which case, set to the specified  
15062        line.  
15063        *Current column*: Set to non-<blank>.  
15064        **Escape**  
15065        *Synopsis:*     ! *command*  
15066              [*addr*]! *command*  
15067        The contents of the line after the '!' shall have '%', '#', and '!' characters expanded as  
15068        described in **Command Line Parsing in ex** (on page 362). If the expansion causes the text of the  
15069        line to change, it shall be redisplayed, preceded by a single '!' character.  
15070        The *ex* utility shall execute the program named by the **shell** edit option. It shall pass two  
15071        arguments to the program; the first shall be **-c**, and the second shall be the expanded arguments  
15072        to the ! command as a single argument.  
15073        If no lines are specified, the standard input, standard output, and standard error of the program  
15074        shall be set to the standard input, standard output, and standard error of the *ex* program when it  
15075        was invoked. In addition, a warning message shall be written if the edit buffer has been  
15076        modified since the last complete write, and the **warn** edit option is set.  
15077        If lines are specified, they shall be passed to the program as standard input, and the standard  
15078        output and standard error of the program shall replace those lines in the edit buffer. Each line in  
15079        the program output (as delimited by <newline>s or the end of the output if it is not immediately  
15080        preceded by a <newline>), shall be a separate line in the edit buffer. Any occurrences of  
15081        <carriage-return> and <newline> pairs in the output shall be treated as single <newline>s. The  
15082        specified lines shall be copied into the unnamed buffer before they are replaced, and the  
15083        unnamed buffer shall become a line-mode buffer.  
15084        If in *ex* mode, a single '!' character shall be written when the program completes.  
15085        This command shall be affected by the **shell** and **warn** edit options. If no lines are specified, this  
15086        command shall be affected by the **autowrite** and **writeany** edit options. If lines are specified, this  
15087        command shall be affected by the **autoprint** edit option.  
15088        *Current line*:  
15089              1. If no lines are specified, unchanged.  
15090              2. Otherwise, set to the last line read in, if any lines are read in.  
15091              3. Otherwise, set to the line before the first line of the lines specified, if that line exists.  
15092              4. Otherwise, set to the first line of the edit buffer if the edit buffer is not empty.  
15093              5. Otherwise, set to zero.  
15094        *Current column*: If no lines are specified, unchanged. Otherwise, set to non-<blank>.

15095       **Shift Left**

15096       *Synopsis:*    [*2addr*] <[< . . . ] [*count*] [*flags*]

15097       Shift the specified lines to the start of the line; the number of column positions to be shifted shall  
15098       be the number of command characters times the value of the **shiftwidth** edit option. Only  
15099       leading <blank>s shall be deleted or changed into other <blank>s in shifting; other characters  
15100       shall not be affected.

15101       Lines to be shifted shall be copied into the unnamed buffer, which shall become a line-mode  
15102       buffer.

15103       This command shall be affected by the **autoprint** edit option.

15104       *Current line:* Set to the last line in the lines specified.

15105       *Current column:* Set to non-<blank>.

15106       **Shift Right**

15107       *Synopsis:*    [*2addr*] >[> . . . ] [*count*] [*flags*]

15108       Shift the specified lines away from the start of the line; the number of column positions to be  
15109       shifted shall be the number of command characters times the value of the **shiftwidth** edit option.  
15110       The shift shall be accomplished by adding <blank>s as a prefix to the line or changing leading  
15111       <blank>s into other <blank>s. Empty lines shall not be changed.

15112       Lines to be shifted shall be copied into the unnamed buffer, which shall become a line-mode  
15113       buffer.

15114       This command shall be affected by the **autoprint** edit option.

15115       *Current line:* Set to the last line in the lines specified.

15116       *Current column:* Set to non-<blank>.

15117       **<control>-D**

15118       *Synopsis:*    <control>-D

15119       Write the next *n* lines, where *n* is the minimum of the values of the **scroll** edit option and the  
15120       number of lines after the current line in the edit buffer. If the current line is the last line of the  
15121       edit buffer it shall be an error.

15122       *Current line:* Set to the last line written.

15123       *Current column:* Set to non-<blank>.

15124       **Write Line Number**

15125       *Synopsis:*    [*1addr*] = [*flags*]

15126       If *line* is not specified, it shall default to the last line in the edit buffer. Write the line number of  
15127       the specified line.

15128       *Current line:* Unchanged.

15129       *Current column:* Unchanged.

15130       **Execute**

15131       *Synopsis:*    [*2addr*] @ *buffer*  
 15132                    [*2addr*] \* *buffer*

15133       If no buffer is specified or is specified as '@' or '\*', the last buffer executed shall be used. If no  
 15134       previous buffer has been executed, it shall be an error.

15135       For each line specified by the addresses, set the current line ('.') to the specified line, and  
 15136       execute the contents of the named *buffer* (as they were at the time the @ command was executed)  
 15137       as ex commands. For each line of a line-mode buffer, and all but the last line of a character-mode  
 15138       buffer, the ex command parser shall behave as if the line was terminated by a <newline>.

15139       If an error occurs during this process, or a line specified by the addresses does not exist when the  
 15140       current line would be set to it, or more than a single line was specified by the addresses, and the  
 15141       contents of the edit buffer are replaced (for example, by the ex :edit command) an error message  
 15142       shall be written, and no more commands resulting from the execution of this command shall be  
 15143       processed.

15144       *Current line:* As specified for the individual ex commands.

15145       *Current column:* As specified for the individual ex commands.

15146       **Regular Expressions in ex**

15147       The ex utility shall support regular expressions that are a superset of the basic regular  
 15148       expressions described in the Base Definitions volume of IEEE Std 1003.1-2001, Section 9.3, Basic  
 15149       Regular Expressions. A null regular expression ("//") shall be equivalent to the last regular  
 15150       expression encountered.

15151       Regular expressions can be used in addresses to specify lines and, in some commands (for  
 15152       example, the **substitute** command), to specify portions of a line to be substituted.

15153       The following constructs can be used to enhance the basic regular expressions:

15154       \< Match the beginning of a *word*. (See the definition of *word* at the beginning of **Command**  
 15155       **Descriptions in ex** (on page 368).)

15156       \> Match the end of a *word*.

15157       ~ Match the replacement part of the last **substitute** command. The tilde ('~') character can  
 15158       be escaped in a regular expression to become a normal character with no special meaning.  
 15159       The backslash shall be discarded.

15160       When the editor option **magic** is not set, the only characters with special meanings shall be '^'  
 15161       at the beginning of a pattern, '\$' at the end of a pattern, and '\'. The characters '.', '\*',  
 15162       '[', and '^' shall be treated as ordinary characters unless preceded by a '\'; when preceded  
 15163       by a '\' they shall regain their special meaning, or in the case of backslash, be handled as a  
 15164       single backslash. Backslashes used to escape other characters shall be discarded.

15165       **Replacement Strings in ex**

15166       The character '&' ('&' if the editor option **magic** is not set) in the replacement string shall  
 15167       stand for the text matched by the pattern to be replaced. The character '~' ('\~' if **magic** is not  
 15168       set) shall be replaced by the replacement part of the previous **substitute** command. The  
 15169       sequence '\n', where *n* is an integer, shall be replaced by the text matched by the pattern  
 15170       enclosed in the *n*th set of parentheses '\(' and '\)'.

15171       The strings '\l', '\u', '\L', and '\U' can be used to modify the case of elements in the  
 15172       replacement string (using the '\&' or "\digit) notation. The string '\l' ('\u') shall cause

15173 the character that follows to be converted to lowercase (uppercase). The string '\L' ('\U')  
15174 shall cause all characters subsequent to it to be converted to lowercase (uppercase) as they are  
15175 inserted by the substitution until the string '\e' or '\E', or the end of the replacement string,  
15176 is encountered.

15177 Otherwise, any character following a backslash shall be treated as that literal character, and the  
15178 escaping backslash shall be discarded.

15179 An example of case conversion with the **s** command is as follows:

```
15180 :p
15181 The cat sat on the mat.
15182 :s/\<.at\>/\u&/gp
15183 The Cat Sat on the Mat.
15184 :s/S\(.*\)M/S\U\1\em/p
15185 The Cat SAT ON THE Mat.
```

## 15186 Edit Options in ex

15187 The **ex** utility has a number of options that modify its behavior. These options have default  
15188 settings, which can be changed using the **set** command.

15189 Options are Boolean unless otherwise specified.

### 15190 **autoindent, ai**

15191 [Default *unset*]

15192 If **autoindent** is set, each line in input mode shall be indented (using first as many **<tab>**s as  
15193 possible, as determined by the editor option **tabstop**, and then using **<space>**s) to align with  
15194 another line, as follows:

- 15195 1. If in open or visual mode and the text input is part of a line-oriented command (see the  
15196 EXTENDED DESCRIPTION in **vi**), align to the first column.
- 15197 2. Otherwise, if in open or visual mode, indentation for each line shall be set as follows:
  - 15198 a. If a line was previously inserted as part of this command, it shall be set to the  
15199 indentation of the last inserted line by default, or as otherwise specified for the  
15200 **<control>-D** character in **Input Mode Commands in vi** (on page 1023).
  - 15201 b. Otherwise, it shall be set to the indentation of the previous current line, if any;  
15202 otherwise, to the first column.
- 15203 3. For the **ex a, i, and c** commands, indentation for each line shall be set as follows:
  - 15204 a. If a line was previously inserted as part of this command, it shall be set to the  
15205 indentation of the last inserted line by default, or as otherwise specified for the **eof**  
15206 character in **Scroll** (on page 366).
  - 15207 b. Otherwise, if the command is the **ex a** command, it shall be set to the line appended  
15208 after, if any; otherwise to the first column.
  - 15209 c. Otherwise, if the command is the **ex i** command, it shall be set to the line inserted  
15210 before, if any; otherwise to the first column.
  - 15211 d. Otherwise, if the command is the **ex c** command, it shall be set to the indentation of  
15212 the line replaced.

15213       **autoprint, ap**

15214       [Default set]

15215       If **autoprint** is set, the current line shall be written after each **ex** command that modifies the  
15216       contents of the current edit buffer, and after each **tag** command for which the tag search pattern  
15217       was found or tag line number was valid, unless:

- 15218       1. The command was executed while in open or visual mode.
- 15219       2. The command was executed as part of a **global** or **v** command or @ buffer execution.
- 15220       3. The command was the form of the **read** command that reads a file into the edit buffer.
- 15221       4. The command was the **append**, **change**, or **insert** command.
- 15222       5. The command was not terminated by a <newline>.
- 15223       6. The current line shall be written by a flag specified to the command; for example, **delete #**  
15224       shall write the current line as specified for the flag modifier to the **delete** command, and  
15225       not as specified by the **autoprint** edit option.

15226       **autowrite, aw**

15227       [Default unset]

15228       If **autowrite** is set, and the edit buffer has been modified since it was last completely written to  
15229       any file, the contents of the edit buffer shall be written as if the **ex write** command had been  
15230       specified without arguments, before each command affected by the **autowrite** edit option is  
15231       executed. Appending the character '!' to the command name of any of the **ex** commands  
15232       except '!' shall prevent the write. If the write fails, it shall be an error and the command shall  
15233       not be executed.

15234       **beautify, bf**

15235 XSI     [Default unset]

15236       If **beautify** is set, all non-printable characters, other than <tab>s, <newline>s, and <form-feed>s,  
15237       shall be discarded from text read in from files.

15238       **directory, dir**

15239       [Default *implementation-defined*]

15240       The value of this option specifies the directory in which the editor buffer is to be placed. If this  
15241       directory is not writable by the user, the editor shall quit.

15242       **edcompatible, ed**

15243       [Default unset]

15244       Causes the presence of **g** and **c** suffixes on substitute commands to be remembered, and toggled  
15245       by repeating the suffixes.

15246       **errorbells, eb**

15247       [Default *unset*]

15248       If the editor is in **ex** mode, and the terminal does not support a standout mode (such as inverse video), and **errorbells** is set, error messages shall be preceded by alerting the terminal.

15250       **exrc**

15251       [Default *unset*]

15252       If **exrc** is set, **ex** shall access any **.exrc** file in the current directory, as described in **Initialization in ex and vi** (on page 358). If **exrc** is not set, **ex** shall ignore any **.exrc** file in the current directory during initialization, unless the current directory is that named by the **HOME** environment variable.

15256       **ignorecase, ic**

15257       [Default *unset*]

15258       If **ignorecase** is set, characters that have uppercase and lowercase representations shall have those representations considered as equivalent for purposes of regular expression comparison.

15260       The **ignorecase** edit option shall affect all remembered regular expressions; for example, unsetting the **ignorecase** edit option shall cause a subsequent **vi n** command to search for the last basic regular expression in a case-sensitive fashion.

15263       **list**

15264       [Default *unset*]

15265       If **list** is set, edit buffer lines written while in **ex** command mode shall be written as specified for the **print** command with the **I** flag specified. In open or visual mode, each edit buffer line shall be displayed as specified for the **ex print** command with the **I** flag specified. In open or visual text input mode, when the cursor does not rest on any character in the line, it shall rest on the '**\$**' marking the end of the line.

15270       **magic**

15271       [Default *set*]

15272       If **magic** is set, modify the interpretation of characters in regular expressions and substitution replacement strings (see **Regular Expressions in ex** (on page 391) and **Replacement Strings in ex** (on page 391)).

15275       **mesg**

15276       [Default *set*]

15277       If **mesg** is set, the permission for others to use the **write** or **talk** commands to write to the terminal shall be turned on while in open or visual mode. The shell-level command **mesg n** shall take precedence over any setting of the **ex mesg** option; that is, if **mesg y** was issued before the editor started (or in a shell escape), such as:

15281       `:!mesg y`

15282       the **mesg** option in **ex** shall suppress incoming messages, but the **mesg** option shall not enable incoming messages if **mesg n** was issued.

15284       **number, nu**

15285       [Default unset]

15286       If **number** is set, edit buffer lines written while in **ex** command mode shall be written with line  
15287       numbers, in the format specified by the **print** command with the # flag specified. In **ex** text input  
15288       mode, each line shall be preceded by the line number it will have in the file.

15289       In open or visual mode, each edit buffer line shall be displayed with a preceding line number, in  
15290       the format specified by the **ex print** command with the # flag specified. This line number shall  
15291       not be considered part of the line for the purposes of evaluating the current column; that is,  
15292       column position 1 shall be the first column position after the format specified by the **print**  
15293       command.

15294       **paragraphs, para**

15295       [Default in the POSIX locale `IPLPPPQPP LIpplpipbp`]

15296       The **paragraphs** edit option shall define additional paragraph boundaries for the open and visual  
15297       mode commands. The **paragraphs** edit option can be set to a character string consisting of zero  
15298       or more character pairs. It shall be an error to set it to an odd number of characters.

15299       **prompt**

15300       [Default set]

15301       If **prompt** is set, **ex** command mode input shall be prompted for with a colon (':'); when unset,  
15302       no prompt shall be written.

15303       **readonly**

15304       [Default see text]

15305       If the **readonly** edit option is set, read-only mode shall be enabled (see **Write** (on page 386)). The  
15306       **readonly** edit option shall be initialized to set if either of the following conditions are true:

- 15307       • The command-line option **-R** was specified.
- 15308       • Performing actions equivalent to the **access()** function called with the following arguments  
15309        indicates that the file lacks write permission:
  - 15310           1. The current pathname is used as the *path* argument.
  - 15311           2. The constant **W\_OK** is used as the *amode* argument.

15312       The **readonly** edit option may be initialized to set for other, implementation-defined reasons.  
15313       The **readonly** edit option shall not be initialized to unset based on any special privileges of the  
15314       user or process. The **readonly** edit option shall be reinitialized each time that the contents of the  
15315       edit buffer are replaced (for example, by an **edit** or **next** command) unless the user has explicitly  
15316       set it, in which case it shall remain set until the user explicitly unsets it. Once unset, it shall again  
15317       be reinitialized each time that the contents of the edit buffer are replaced.

- 15318       **redraw**
- 15319       [Default *unset*]
- 15320       The editor simulates an intelligent terminal on a dumb terminal. (Since this is likely to require a  
15321       large amount of output to the terminal, it is useful only at high transmission speeds.)
- 15322       **remap**
- 15323       [Default *set*]
- 15324       If **remap** is set, map translation shall allow for maps defined in terms of other maps; translation  
15325       shall continue until a final product is obtained. If unset, only a one-step translation shall be done.
- 15326       **report**
- 15327       [Default 5]
- 15328       The value of this **report** edit option specifies what number of lines being added, copied, deleted,  
15329       or modified in the edit buffer will cause an informational message to be written to the user. The  
15330       following conditions shall cause an informational message. The message shall contain the  
15331       number of lines added, copied, deleted, or modified, but is otherwise unspecified.
- 15332       • An *ex* or *vi* editor command, other than **open**, **undo**, or **visual**, that modifies at least the value  
15333       of the **report** edit option number of lines, and which is not part of an *ex global* or **v**  
15334       command, or *ex* or *vi* buffer execution, shall cause an informational message to be written.
- 15335       • An *ex yank* or *vi y* or **Y** command, that copies at least the value of the **report** edit option plus  
15336       1 number of lines, and which is not part of an *ex global* or **v** command, or *ex* or *vi* buffer  
15337       execution, shall cause an informational message to be written.
- 15338       • An *ex global*, **v**, **open**, **undo**, or **visual** command or *ex* or *vi* buffer execution, that adds or  
15339       deletes a total of at least the value of the **report** edit option number of lines, and which is not  
15340       part of an *ex global* or **v** command, or *ex* or *vi* buffer execution, shall cause an informational  
15341       message to be written. (For example, if 3 lines were added and 8 lines deleted during an *ex*  
15342       **visual** command, 5 would be the number compared against the **report** edit option after the  
15343       command completed.)
- 15344       **scroll**, **scr**
- 15345       [Default (number of lines in the display -1)/2]
- 15346       The value of the **scroll** edit option shall determine the number of lines scrolled by the *ex*  
15347       <control>-D and **z** commands. For the *vi* <control>-D and <control>-U commands, it shall be the  
15348       initial number of lines to scroll when no previous <control>-D or <control>-U command has  
15349       been executed.
- 15350       **sections**
- 15351       [Default in the POSIX locale `NHSHH HUnhsh`]
- 15352       The **sections** edit option shall define additional section boundaries for the open and visual mode  
15353       commands. The **sections** edit option can be set to a character string consisting of zero or more  
15354       character pairs; it shall be an error to set it to an odd number of characters.

15355       **shell, sh**

15356       [Default from the environment variable *SHELL*]

15357       The value of this option shall be a string. The default shall be taken from the *SHELL*  
15358       environment variable. If the *SHELL* environment variable is null or empty, the *sh* (see *sh*) utility  
15359       shall be the default.

15360       **shiftwidth, sw**

15361       [Default 8]

15362       The value of this option shall give the width in columns of an indentation level used during  
15363       autoindentation and by the shift commands (< and >).

15364       **showmatch, sm**

15365       [Default *unset*]

15366       The functionality described for the **showmatch** edit option need not be supported on block-  
15367       mode terminals or terminals with insufficient capabilities.

15368       If **showmatch** is set, in open or visual mode, when a ')' or ')' is typed, if the matching '(' or  
15369       '{' is currently visible on the display, the matching '(' or '{' shall be flagged moving the  
15370       cursor to its location for an unspecified amount of time.

15371       **showmode**

15372       [Default *unset*]

15373       If **showmode** is set, in open or visual mode, the current mode that the editor is in shall be  
15374       displayed on the last line of the display. Command mode and text input mode shall be  
15375       differentiated; other unspecified modes and implementation-defined information may be  
15376       displayed.

15377       **slowopen**

15378       [Default *unset*]

15379       If **slowopen** is set during open and visual text input modes, the editor shall not update portions  
15380       of the display other than those display line columns that display the characters entered by the  
15381       user (see **Input Mode Commands in vi** (on page 1023)).

15382       **tabstop, ts**

15383       [Default 8]

15384       The value of this edit option shall specify the column boundary used by a <tab> in the display  
15385       (see **autoprint, ap** (on page 393) and **Input Mode Commands in vi** (on page 1023)).

15386       **taglength, tl**

15387       [Default zero]

15388       The value of this edit option shall specify the maximum number of characters that are  
15389       considered significant in the user-specified tag name and in the tag name from the tags file. If the  
15390       value is zero, all characters in both tag names shall be significant.

15391       **tags**

15392       [Default *see text*]

15393       The value of this edit option shall be a string of <blank>-delimited pathnames of files used by

15394       the **tag** command. The default value is unspecified.

15395       **term**

15396       [Default from the environment variable *TERM*]

15397       The value of this edit option shall be a string. The default shall be taken from the *TERM* variable

15398       in the environment. If the *TERM* environment variable is empty or null, the default is

15399       unspecified. The editor shall use the value of this edit option to determine the type of the display

15400       device.

15401       The results are unspecified if the user changes the value of the term edit option after editor

15402       initialization.

15403       **terse**

15404       [Default *unset*]

15405       If **terse** is set, error messages may be less verbose. However, except for this caveat, error

15406       messages are unspecified. Furthermore, not all error messages need change for different settings

15407       of this option.

15408       **warn**

15409       [Default *set*]

15410       If **warn** is set, and the contents of the edit buffer have been modified since they were last

15411       completely written, the editor shall write a warning message before certain ! commands (see

15412       **Escape** (on page 389)).

15413       **window**

15414       [Default *see text*]

15415       A value used in open and visual mode, by the <control>-B and <control>-F commands, and, in

15416       visual mode, to specify the number of lines displayed when the screen is repainted.

15417       If the **-w** command-line option is not specified, the default value shall be set to the value of the

15418       *LINES* environment variable. If the *LINES* environment variable is empty or null, the default

15419       shall be the number of lines in the display minus 1.

15420       Setting the **window** edit option to zero or to a value greater than the number of lines in the

15421       display minus 1 (either explicitly or based on the **-w** option or the *LINES* environment variable)

15422       shall cause the **window** edit option to be set to the number of lines in the display minus 1.

15423       The baud rate of the terminal line may change the default in an implementation-defined manner.

15424       **wrapmargin, wm**  
15425       [Default 0]  
15426       If the value of this edit option is zero, it shall have no effect.  
15427       If not in the POSIX locale, the effect of this edit option is implementation-defined.  
15428       Otherwise, it shall specify a number of columns from the ending margin of the terminal.  
15429       During open and visual text input modes, for each character for which any part of the character  
15430       is displayed in a column that is less than **wrapmargin** columns from the ending margin of the  
15431       display line, the editor shall behave as follows:  
15432       1. If the character triggering this event is a <blank>, it, and all immediately preceding  
15433       <blank>s on the current line entered during the execution of the current text input  
15434       command, shall be discarded, and the editor shall behave as if the user had entered a single  
15435       <newline> instead. In addition, if the next user-entered character is a <space>, it shall be  
15436       discarded as well.  
15437       2. Otherwise, if there are one or more <blank>s on the current line immediately preceding the  
15438       last group of inserted non-<blank>s which was entered during the execution of the current  
15439       text input command, the <blank>s shall be replaced as if the user had entered a single  
15440       <newline> instead.  
15441       If the **autoindent** edit option is set, and the events described in 1. or 2. are performed, any  
15442       <blank>s at or after the cursor in the current line shall be discarded.  
15443       The ending margin shall be determined by the system or overridden by the user, as described for  
15444       **COLUMNS** in the ENVIRONMENT VARIABLES section and the Base Definitions volume of  
15445       IEEE Std 1003.1-2001, Chapter 8, Environment Variables.  
15446       **wrapscan, ws**  
15447       [Default *set*]  
15448       If **wrapscan** is set, searches (the **ex** / or ? addresses, or open and visual mode /, ?, N, and n  
15449       commands) shall wrap around the beginning or end of the edit buffer; when unset, searches  
15450       shall stop at the beginning or end of the edit buffer.  
15451       **writeany, wa**  
15452       [Default *unset*]  
15453       If **writeany** is set, some of the checks performed when executing the **ex write** commands shall be  
15454       inhibited, as described in editor option **autowrite**.  
15455 **EXIT STATUS**  
15456       The following exit values shall be returned:  
15457       0   Successful completion.  
15458       >0   An error occurred.  
15459 **CONSEQUENCES OF ERRORS**  
15460       When any error is encountered and the standard input is not a terminal device file, **ex** shall not  
15461       write the file or return to command or text input mode, and shall terminate with a non-zero exit  
15462       status.  
15463       Otherwise, when an unrecoverable error is encountered, it shall be equivalent to a SIGHUP  
15464       asynchronous event.

15465       Otherwise, when an error is encountered, the editor shall behave as specified in **Command Line**  
15466       **Parsing in ex** (on page 362).

## 15467 APPLICATION USAGE

15468       If a SIGSEGV signal is received while **ex** is saving a file, the file might not be successfully saved.

15469       The **next** command can accept more than one file, so usage such as:

15470       **next `ls [abc]\*`**

15471       is valid; it would not be valid for the **edit** or **read** commands, for example, because they expect  
15472       only one file and unspecified results occur.

## 15473 EXAMPLES

15474       None.

## 15475 RATIONALE

15476       The **ex/vi** specification is based on the historical practice found in the 4 BSD and System V  
15477       implementations of **ex** and **vi**. A freely redistributable implementation of **ex/vi**, which is  
15478       tracking IEEE Std 1003.1-2001 fairly closely, and demonstrates the intended changes between  
15479       historical implementations and IEEE Std 1003.1-2001, may be obtained by anonymous FTP from:

15480       `ftp://ftp.rdg.opengroup.org/pub/mirrors/nvi` 1

15481       A *restricted editor* (both the historical **red** utility and modifications to **ex**) were considered and  
15482       rejected for inclusion. Neither option provided the level of security that users might expect.

15483       It is recognized that **ex** visual mode and related features would be difficult, if not impossible, to  
15484       implement satisfactorily on a block-mode terminal, or a terminal without any form of cursor  
15485       addressing; thus, it is not a mandatory requirement that such features should work on all  
15486       terminals. It is the intention, however, that an **ex** implementation should provide the full set of  
15487       capabilities on all terminals capable of supporting them.

## 15488 Options

15489       The **-c** replacement for **+command** was inspired by the **-e** option of **sed**. Historically, all such  
15490       commands (see **edit** and **next** as well) were executed from the last line of the edit buffer. This  
15491       meant, for example, that "**+/pattern**" would fail unless the **wrapscan** option was set.  
15492       IEEE Std 1003.1-2001 requires conformance to historical practice. Historically, some  
15493       implementations restricted the **ex** commands that could be listed as part of the command line  
15494       arguments. For consistency, IEEE Std 1003.1-2001 does not permit these restrictions.

15495       In historical implementations of the editor, the **-R** option (and the **readonly** edit option) only  
15496       prevented overwriting of files; appending to files was still permitted, mapping loosely into the  
15497       **csh noclobber** variable. Some implementations, however, have not followed this semantic, and  
15498       **readonly** does not permit appending either. IEEE Std 1003.1-2001 follows the latter practice,  
15499       believing that it is a more obvious and intuitive meaning of **readonly**.

15500       The **-s** option suppresses all interactive user feedback and is useful for editing scripts in batch  
15501       jobs. The list of specific effects is historical practice. The terminal type "incapable of supporting  
15502       open and visual modes" has historically been named "dumb".

15503       The **-t** option was required because the **ctags** utility appears in IEEE Std 1003.1-2001 and the  
15504       option is available in all historical implementations of **ex**.

15505       Historically, the **ex** and **vi** utilities accepted a **-x** option, which did encryption based on the  
15506       algorithm found in the historical **crypt** utility. The **-x** option for encryption, and the associated  
15507       **crypt** utility, were omitted because the algorithm used was not specifiable and the export control  
15508       laws of some nations make it difficult to export cryptographic technology. In addition, it did not

15509 historically provide the level of security that users might expect.

### 15510 Standard Input

15511 An end-of-file condition is not equivalent to an end-of-file character. A common end-of-file  
15512 character, <control>-D, is historically an ex command.

15513 There was no maximum line length in historical implementations of ex. Specifically, as it was  
15514 parsed in chunks, the addresses had a different maximum length than the filenames. Further, the  
15515 maximum line buffer size was declared as BUFSIZ, which was different lengths on different  
15516 systems. This version selected the value of {LINE\_MAX} to impose a reasonable restriction on  
15517 portable usage of ex and to aid test suite writers in their development of realistic tests that  
15518 exercise this limit.

### 15519 Input Files

15520 It was an explicit decision by the standard developers that a <newline> be added to any file  
15521 lacking one. It was believed that this feature of ex and vi was relied on by users in order to make  
15522 text files lacking a trailing <newline> more portable. It is recognized that this will require a  
15523 user-specified option or extension for implementations that permit ex and vi to edit files of type  
15524 other than text if such files are not otherwise identified by the system. It was agreed that the  
15525 ability to edit files of arbitrary type can be useful, but it was not considered necessary to  
15526 mandate that an ex or vi implementation be required to handle files other than text files.

15527 The paragraph in the INPUT FILES section, “By default, …”, is intended to close a long-standing  
15528 security problem in ex and vi; that of the “modeline” or “modelines” edit option. This feature  
15529 allows any line in the first or last five lines of the file containing the strings “ex:” or “vi:”  
15530 (and, apparently, “ei:” or “vx:”) to be a line containing editor commands, and ex interprets all  
15531 the text up to the next ‘:’ or <newline> as a command. Consider the consequences, for  
15532 example, of an unsuspecting user using ex or vi as the editor when replying to a mail message in  
15533 which a line such as:

15534 ex: ! rm -rf :

15535 appeared in the signature lines. The standard developers believed strongly that an editor should  
15536 not by default interpret any lines of a file. Vendors are strongly urged to delete this feature from  
15537 their implementations of ex and vi.

### 15538 Asynchronous Events

15539 The intention of the phrase “complete write” is that the entire edit buffer be written to stable  
15540 storage. The note regarding temporary files is intended for implementations that use temporary  
15541 files to back edit buffers unnamed by the user.

15542 Historically, SIGQUIT was ignored by ex, but was the equivalent of the Q command in visual  
15543 mode; that is, it exited visual mode and entered ex mode. IEEE Std 1003.1-2001 permits, but does  
15544 not require, this behavior. Historically, SIGINT was often used by vi users to terminate text  
15545 input mode (<control>-C is often easier to enter than <ESC>). Some implementations of vi  
15546 alerted the terminal on this event, and some did not. IEEE Std 1003.1-2001 requires that SIGINT  
15547 behave identically to <ESC>, and that the terminal not be alerted.

15548 Historically, suspending the ex editor during text input mode was similar to SIGINT, as  
15549 completed lines were retained, but any partial line discarded, and the editor returned to  
15550 command mode. IEEE Std 1003.1-2001 is silent on this issue; implementations are encouraged to  
15551 follow historical practice, where possible.

15552 Historically, the *vi* editor did not treat SIGTSTP as an asynchronous event, and it was therefore  
15553 impossible to suspend the editor in visual text input mode. There are two major reasons for this.  
15554 The first is that SIGTSTP is a broadcast signal on UNIX systems, and the chain of events where  
15555 the shell execs an application that then execs *vi* usually caused confusion for the terminal state if  
15556 SIGTSTP was delivered to the process group in the default manner. The second was that most  
15557 implementations of the UNIX *curses* package are not reentrant, and the receipt of SIGTSTP at the  
15558 wrong time will cause them to crash. IEEE Std 1003.1-2001 is silent on this issue; implementations  
15559 are encouraged to treat suspension as an asynchronous event if possible.

15560 Historically, modifications to the edit buffer made before SIGINT interrupted an operation were  
15561 retained; that is, anywhere from zero to all of the lines to be modified might have been modified  
15562 by the time the SIGINT arrived. These changes were not discarded by the arrival of SIGINT.  
15563 IEEE Std 1003.1-2001 permits this behavior, noting that the **undo** command is required to be able  
15564 to undo these partially completed commands.

15565 The action taken for signals other than SIGINT, SIGCONT, SIGHUP, and SIGTERM is  
15566 unspecified because some implementations attempt to save the edit buffer in a useful state when  
15567 other signals are received.

#### 15568 Standard Error

15569 For *ex/vi*, diagnostic messages are those messages reported as a result of a failed attempt to  
15570 invoke *ex* or *vi*, such as invalid options or insufficient resources, or an abnormal termination  
15571 condition. Diagnostic messages should not be confused with the error messages generated by  
15572 inappropriate or illegal user commands.

#### 15573 Initialization in ex and vi

15574 If an *ex* command (other than **cd**, **chdir**, or **source**) has a filename argument, one or both of the  
15575 alternate and current pathnames will be set. Informally, they are set as follows:

- 15576 1. If the *ex* command is one that replaces the contents of the edit buffer, and it succeeds, the  
15577 current pathname will be set to the filename argument (the first filename argument in the  
15578 case of the **next** command) and the alternate pathname will be set to the previous current  
15579 pathname, if there was one.
- 15580 2. In the case of the file read/write forms of the **read** and **write** commands, if there is no  
15581 current pathname, the current pathname will be set to the filename argument.
- 15582 3. Otherwise, the alternate pathname will be set to the filename argument.

15583 For example, **:edit foo** and **:recover foo**, when successful, set the current pathname, and, if there  
15584 was a previous current pathname, the alternate pathname. The commands **:write**, **!command**,  
15585 and **:edit** set neither the current or alternate pathnames. If the **:edit foo** command were to fail for  
15586 some reason, the alternate pathname would be set. The **read** and **write** commands set the  
15587 alternate pathname to their *file* argument, unless the current pathname is not set, in which case  
15588 they set the current pathname to their *file* arguments. The alternate pathname was not  
15589 historically set by the **:source** command. IEEE Std 1003.1-2001 requires conformance to historical  
15590 practice. Implementations adding commands that take filenames as arguments are encouraged  
15591 to set the alternate pathname as described here.

15592 Historically, *ex* and *vi* read the **.exrc** file in the **\$HOME** directory twice, if the editor was executed  
15593 in the **\$HOME** directory. IEEE Std 1003.1-2001 prohibits this behavior.

15594 Historically, the 4 BSD *ex* and *vi* read the **\$HOME** and local **.exrc** files if they were owned by the  
15595 real ID of the user, or the **sourceany** option was set, regardless of other considerations. This was  
15596 a security problem because it is possible to put normal UNIX system commands inside a **.exrc**

15597 file. IEEE Std 1003.1-2001 does not specify the **sourceany** option, and historical implementations  
15598 are encouraged to delete it.

15599 The **.exrc** files must be owned by the real ID of the user, and not writable by anyone other than  
15600 the owner. The appropriate privileges exception is intended to permit users to acquire special  
15601 privileges, but continue to use the **.exrc** files in their home directories.

15602 System V Release 3.2 and later **vi** implementations added the option [**no**]**exrc**. The behavior is  
15603 that local **.exrc** files are read-only if the **exrc** option is set. The default for the **exrc** option was off,  
15604 so by default, local **.exrc** files were not read. The problem this was intended to solve was that  
15605 System V permitted users to give away files, so there is no possible ownership or writeability  
15606 test to ensure that the file is safe. This is still a security problem on systems where users can give  
15607 away files, but there is nothing additional that IEEE Std 1003.1-2001 can do. The  
15608 implementation-defined exception is intended to permit groups to have local **.exrc** files that are  
15609 shared by users, by creating pseudo-users to own the shared files.

15610 IEEE Std 1003.1-2001 does not mention system-wide **ex** and **vi** start-up files. While they exist in  
15611 several implementations of **ex** and **vi**, they are not present in any implementations considered  
15612 historical practice by IEEE Std 1003.1-2001. Implementations that have such files should use  
15613 them only if they are owned by the real user ID or an appropriate user (for example, root on  
15614 UNIX systems) and if they are not writable by any user other than their owner. System-wide  
15615 start-up files should be read before the **EXINIT** variable, **\$HOME/.exrc**, or local **.exrc** files are  
15616 evaluated.

15617 Historically, any **ex** command could be entered in the **EXINIT** variable or the **.exrc** file, although  
15618 ones requiring that the edit buffer already contain lines of text generally caused historical  
15619 implementations of the editor to drop **core**. IEEE Std 1003.1-2001 requires that any **ex** command  
15620 be permitted in the **EXINIT** variable and **.exrc** files, for simplicity of specification and  
15621 consistency, although many of them will obviously fail under many circumstances.

15622 The initialization of the contents of the edit buffer uses the phrase “the effect shall be” with  
15623 regard to various **ex** commands. The intent of this phrase is that edit buffer contents loaded  
15624 during the initialization phase not be lost; that is, loading the edit buffer should fail if the **.exrc**  
15625 file read in the contents of a file and did not subsequently write the edit buffer. An additional  
15626 intent of this phrase is to specify that the initial current line and column is set as specified for the  
15627 individual **ex** commands.

15628 Historically, the **-t** option behaved as if the tag search were a **+command**; that is, it was executed  
15629 from the last line of the file specified by the tag. This resulted in the search failing if the pattern  
15630 was a forward search pattern and the **wrapscan** edit option was not set. IEEE Std 1003.1-2001  
15631 does not permit this behavior, requiring that the search for the tag pattern be performed on the  
15632 entire file, and, if not found, that the current line be set to a more reasonable location in the file.

15633 Historically, the empty edit buffer presented for editing when a file was not specified by the user  
15634 was unnamed. This is permitted by IEEE Std 1003.1-2001; however, implementations are  
15635 encouraged to provide users a temporary filename for this buffer because it permits them the  
15636 use of **ex** commands that use the current pathname during temporary edit sessions.

15637 Historically, the file specified using the **-t** option was not part of the current argument list. This  
15638 practice is permitted by IEEE Std 1003.1-2001; however, implementations are encouraged to  
15639 include its name in the current argument list for consistency.

15640 Historically, the **-c** command was generally not executed until a file that already exists was  
15641 edited. IEEE Std 1003.1-2001 requires conformance to this historical practice. Commands that  
15642 could cause the **-c** command to be executed include the **ex** commands **edit**, **next**, **recover**,  
15643 **rewind**, and **tag**, and the **vi** commands **<control>-^** and **<control>-J**. Historically, reading a file  
15644 into an edit buffer did not cause the **-c** command to be executed (even though it might set the

15645 current pathname) with the exception that it did cause the **-c** command to be executed if: the  
15646 editor was in **ex** mode, the edit buffer had no current pathname, the edit buffer was empty, and  
15647 no read commands had yet been attempted. For consistency and simplicity of specification,  
15648 IEEE Std 1003.1-2001 does not permit this behavior.

15649 Historically, the **-r** option was the same as a normal edit session if there was no recovery  
15650 information available for the file. This allowed users to enter:

15651 `vi -r *.c`

15652 and recover whatever files were recoverable. In some implementations, recovery was attempted  
15653 only on the first file named, and the file was not entered into the argument list; in others,  
15654 recovery was attempted for each file named. In addition, some historical implementations  
15655 ignored **-r** if **-t** was specified or did not support command line *file* arguments with the **-t** option.  
15656 For consistency and simplicity of specification, IEEE Std 1003.1-2001 disallows these special  
15657 cases, and requires that recovery be attempted the first time each file is edited.

15658 Historically, **vi** initialized the ‘ and ’ marks, but **ex** did not. This meant that if the first command  
15659 in **ex** mode was **visual** or if an **ex** command was executed first (for example, `vi +10 file`), **vi** was  
15660 entered without the marks being initialized. Because the standard developers believed the marks  
15661 to be generally useful, and for consistency and simplicity of specification, IEEE Std 1003.1-2001  
15662 requires that they always be initialized if in open or visual mode, or if in **ex** mode and the edit  
15663 buffer is not empty. Not initializing it in **ex** mode if the edit buffer is empty is historical practice;  
15664 however, it has always been possible to set (and use) marks in empty edit buffers in open and  
15665 visual mode edit sessions.

## 15666 Addressing

15667 Historically, **ex** and **vi** accepted the additional addressing forms ‘\//’ and ‘\?’. They were  
15668 equivalent to “//” and “??”, respectively. They are not required by IEEE Std 1003.1-2001,  
15669 mostly because nobody can remember whether they ever did anything different historically.

15670 Historically, **ex** and **vi** permitted an address of zero for several commands, and permitted the %  
15671 address in empty files for others. For consistency, IEEE Std 1003.1-2001 requires support for the  
15672 former in the few commands where it makes sense, and disallows it otherwise. In addition,  
15673 because IEEE Std 1003.1-2001 requires that % be logically equivalent to “1,\$”, it is also  
15674 supported where it makes sense and disallowed otherwise.

15675 Historically, the % address could not be followed by further addresses. For consistency and  
15676 simplicity of specification, IEEE Std 1003.1-2001 requires that additional addresses be supported.

15677 All of the following are valid *addresses*:

15678 `+++` Three lines after the current line.

15679 `/re/-` One line before the next occurrence of *re*.

15680 `-2` Two lines before the current line.

15681 `3 ----- 2` Line one (note intermediate negative address).

15682 `1 2 3` Line six.

15683 Any number of addresses can be provided to commands taking addresses; for example,  
15684 “1,2,3,4,5P” prints lines 4 and 5, because two is the greatest valid number of addresses  
15685 accepted by the **print** command. This, in combination with the semicolon delimiter, permits  
15686 users to create commands based on ordered patterns in the file. For example, the command  
15687 `3;/foo;/+2print` will display the first line after line 3 that contains the pattern *foo*, plus the next  
15688 two lines. Note that the address 3; must be evaluated before being discarded because the search

15689 origin for the **/foo/** command depends on this.

15690 Historically, values could be added to addresses by including them after one or more <blank>s;  
15691 for example, **3 - 5p** wrote the seventh line of the file, and **/foo/ 5** was the same as **/foo/+5**.  
15692 However, only absolute values could be added; for example, **5 /foo/** was an error.  
15693 IEEE Std 1003.1-2001 requires conformance to historical practice. Address offsets are separately  
15694 specified from addresses because they could historically be provided to visual mode search  
15695 commands.

15696 Historically, any missing addresses defaulted to the current line. This was true for leading and  
15697 trailing comma-delimited addresses, and for trailing semicolon-delimited addresses. For  
15698 consistency, IEEE Std 1003.1-2001 requires it for leading semicolon addresses as well.

15699 Historically, **ex** and **vi** accepted the **'^'** character as both an address and as a flag offset for  
15700 commands. In both cases it was identical to the **'-'** character. IEEE Std 1003.1-2001 does not  
15701 require or prohibit this behavior.

15702 Historically, the enhancements to basic regular expressions could be used in addressing; for  
15703 example, **'~'**, **'\<'**, and **'\>'**. IEEE Std 1003.1-2001 requires conformance to historical  
15704 practice; that is, that regular expression usage be consistent, and that regular expression  
15705 enhancements be supported wherever regular expressions are used.

## 15706 Command Line Parsing in ex

15707 Historical **ex** command parsing was even more complex than that described here.  
15708 IEEE Std 1003.1-2001 requires the subset of the command parsing that the standard developers  
15709 believed was documented and that users could reasonably be expected to use in a portable  
15710 fashion, and that was historically consistent between implementations. (The discarded  
15711 functionality is obscure, at best.) Historical implementations will require changes in order to  
15712 comply with IEEE Std 1003.1-2001; however, users are not expected to notice any of these  
15713 changes. Most of the complexity in **ex** parsing is to handle three special termination cases:

- 15714 1. The **!**, **global**, **v**, and the filter versions of the **read** and **write** commands are delimited by  
15715 <newline>s (they can contain vertical-line characters that are usually shell pipes).
- 15716 2. The **ex**, **edit**, **next**, and **visual** in open and visual mode commands all take **ex** commands,  
15717 optionally containing vertical-line characters, as their first arguments.
- 15718 3. The **s** command takes a regular expression as its first argument, and uses the delimiting  
15719 characters to delimit the command.

15720 Historically, vertical-line characters in the **+command** argument of the **ex**, **edit**, **next**, **vi**, and  
15721 **visual** commands, and in the **pattern** and **replacement** parts of the **s** command, did not delimit the  
15722 command, and in the filter cases for **read** and **write**, and the **!**, **global**, and **v** commands, they did  
15723 not delimit the command at all. For example, the following commands are all valid:

```
15724 :edit +25 | s/abc/ABC/ file.c
15725 :s/ | /PIPE/
15726 :read !spell % | colummate
15727 :global/pattern/p | l
15728 :s/a/b/ | s/c/d | set
```

15729 Historically, empty or <blank> filled lines in **.exrc** files and **sourced** files (as well as **EXINIT**  
15730 variables and **ex** command scripts) were treated as default commands; that is, **print** commands.  
15731 IEEE Std 1003.1-2001 specifically requires that they be ignored when encountered in **.exrc** and  
15732 **sourced** files to eliminate a common source of new user error.

15733 Historically, *ex* commands with multiple adjacent (or <blank>-separated) vertical lines were  
15734 handled oddly when executed from *ex* mode. For example, the command ||| <carriage-return>,  
15735 when the cursor was on line 1, displayed lines 2, 3, and 5 of the file. In addition, the command |  
15736 would only display the line after the next line, instead of the next two lines. The former worked  
15737 more logically when executed from *vi* mode, and displayed lines 2, 3, and 4.  
15738 IEEE Std 1003.1-2001 requires the *vi* behavior; that is, a single default command and line number  
15739 increment for each command separator, and trailing <newline>s after vertical-line separators are  
15740 discarded.

15741 Historically, *ex* permitted a single extra colon as a leading command character; for example,  
15742 :g/pattern/:p was a valid command. IEEE Std 1003.1-2001 generalizes this to require that any  
15743 number of leading colon characters be stripped.

15744 Historically, any prefix of the **delete** command could be followed without intervening <blank>s  
15745 by a flag character because in the command d p, p is interpreted as the buffer p.  
15746 IEEE Std 1003.1-2001 requires conformance to historical practice.

15747 Historically, the **k** command could be followed by the mark name without intervening  
15748 <blank>s. IEEE Std 1003.1-2001 requires conformance to historical practice.

15749 Historically, the **s** command could be immediately followed by flag and option characters; for  
15750 example, s/e/E/|s|sgc3p was a valid command. However, flag characters could not stand alone;  
15751 for example, the commands sp and s l would fail, while the command sgp and s gl would  
15752 succeed. (Obviously, the '#' flag character was used as a delimiter character if it followed the  
15753 command.) Another issue was that option characters had to precede flag characters even when  
15754 the command was fully specified; for example, the command s/e/E/pg would fail, while the  
15755 command s/e/E/gp would succeed. IEEE Std 1003.1-2001 requires conformance to historical  
15756 practice.

15757 Historically, the first command name that had a prefix matching the input from the user was the  
15758 executed command; for example, ve, ver, and vers all executed the **version** command.  
15759 Commands were in a specific order, however, so that a matched **append**, not **abbreviate**.  
15760 IEEE Std 1003.1-2001 requires conformance to historical practice. The restriction on command  
15761 search order for implementations with extensions is to avoid the addition of commands such  
15762 that the historical prefixes would fail to work portably.

15763 Historical implementations of *ex* and *vi* did not correctly handle multiple *ex* commands,  
15764 separated by vertical-line characters, that entered or exited visual mode or the editor. Because  
15765 implementations of *vi* exist that do not exhibit this failure mode, IEEE Std 1003.1-2001 does not  
15766 permit it.

15767 The requirement that alphabetic command names consist of all following alphabetic characters  
15768 up to the next non-alphabetic character means that alphabetic command names must be  
15769 separated from their arguments by one or more non-alphabetic characters, normally a <blank>  
15770 or '!' character, except as specified for the exceptions, the **delete**, **k**, and **s** commands.

15771 Historically, the repeated execution of the *ex* default **print** commands (<control>-D, eof,  
15772 <newline>, <carriage-return>) erased any prompting character and displayed the next lines  
15773 without scrolling the terminal; that is, immediately below any previously displayed lines. This  
15774 provided a cleaner presentation of the lines in the file for the user. IEEE Std 1003.1-2001 does not  
15775 require this behavior because it may be impossible in some situations; however,  
15776 implementations are strongly encouraged to provide this semantic if possible.

15777 Historically, it was possible to change files in the middle of a command, and have the rest of the  
15778 command executed in the new file; for example:

15779 :edit +25 file.c | s/abc/ABC/ | 1  
15780 was a valid command, and the substitution was attempted in the newly edited file.  
15781 IEEE Std 1003.1-2001 requires conformance to historical practice. The following commands are  
15782 examples that exercise the ex parser:  
15783 echo 'foo | bar' > file1; echo 'foo/bar' > file2;  
15784 vi  
15785 :edit +1 | s/|/PIPE/ | w file1 | e file2 | 1 | s/\//SLASH/ | wq  
15786 Historically, there was no protection in editor implementations to avoid ex **global**, **v**, **@**, or **\***  
15787 commands changing edit buffers during execution of their associated commands. Because this  
15788 would almost invariably result in catastrophic failure of the editor, and implementations exist  
15789 that do exhibit these problems, IEEE Std 1003.1-2001 requires that changing the edit buffer  
15790 during a **global** or **v** command, or during a **@** or **\*** command for which there will be more than a  
15791 single execution, be an error. Implementations supporting multiple edit buffers simultaneously  
15792 are strongly encouraged to apply the same semantics to switching between buffers as well.  
15793 The ex command quoting required by IEEE Std 1003.1-2001 is a superset of the quoting in  
15794 historical implementations of the editor. For example, it was not historically possible to escape a  
15795 <blank> in a filename; for example, :edit foo\\ bar would report that too many filenames had  
15796 been entered for the edit command, and there was no method of escaping a <blank> in the first  
15797 argument of an edit, ex, next, or visual command at all. IEEE Std 1003.1-2001 extends historical  
15798 practice, requiring that quoting behavior be made consistent across all ex commands, except for  
15799 the map, unmap, abbreviate, and unabbreviate commands, which historically used <control>-V  
15800 instead of backslashes for quoting. For those four commands, IEEE Std 1003.1-2001 requires  
15801 conformance to historical practice.  
15802 Backslash quoting in ex is non-intuitive. Backslash escapes are ignored unless they escape a  
15803 special character; for example, when performing file argument expansion, the string "\\\%" is  
15804 equivalent to '\%', not "\<current pathname>". This can be confusing for users because  
15805 backslash is usually one of the characters that causes shell expansion to be performed, and  
15806 therefore shell quoting rules must be taken into consideration. Generally, quoting characters are  
15807 only considered if they escape a special character, and a quoting character must be provided for  
15808 each layer of parsing for which the character is special. As another example, only a single  
15809 backslash is necessary for the '\1' sequence in substitute replacement patterns, because the  
15810 character '1' is not special to any parsing layer above it.  
15811 <control>-V quoting in ex is slightly different from backslash quoting. In the four commands  
15812 where <control>-V quoting applies (abbreviate, unabbreviate, map, and unmap), any character  
15813 may be escaped by a <control>-V whether it would have a special meaning or not.  
15814 IEEE Std 1003.1-2001 requires conformance to historical practice.  
15815 Historical implementations of the editor did not require delimiters within character classes to be  
15816 escaped; for example, the command :s/[]/ on the string "xxx/yyy" would delete the '/' from  
15817 the string. IEEE Std 1003.1-2001 disallows this historical practice for consistency and because it  
15818 places a large burden on implementations by requiring that knowledge of regular expressions be  
15819 built into the editor parser.  
15820 Historically, quoting <newline>s in ex commands was handled inconsistently. In most cases, the  
15821 <newline> always terminated the command, regardless of any preceding escape character,  
15822 because backslash characters did not escape <newline>s for most ex commands. However, some  
15823 ex commands (for example, s, map, and abbreviation) permitted <newline>s to be escaped  
15824 (although in the case of map and abbreviation, <control>-V characters escaped them instead of  
15825 backslashes). This was true in not only the command line, but also .exrc and sourced files. For  
15826 example, the command:

15827 map = foo<control-V><newline>bar  
15828 would succeed, although it was sometimes difficult to get the <control>-V and the inserted  
15829 <newline> passed to the *ex* parser. For consistency and simplicity of specification,  
15830 IEEE Std 1003.1-2001 requires that it be possible to escape <newline>s in *ex* commands at all  
15831 times, using backslashes for most *ex* commands, and using <control>-V characters for the **map**  
15832 and **abbreviation** commands. For example, the command **print<newline>list** is required to be  
15833 parsed as the single command **print<newline>list**. While this differs from historical practice,  
15834 IEEE Std 1003.1-2001 developers believed it unlikely that any script or user depended on the  
15835 historical behavior.  
15836 Historically, an error in a command specified using the **-c** option did not cause the rest of the **-c**  
15837 commands to be discarded. IEEE Std 1003.1-2001 disallows this for consistency with mapped  
15838 keys, the **@**, **global**, **source**, and **v** commands, the *EXINIT* environment variable, and the **.exrc**  
15839 files.  
15840 **Input Editing in ex**  
15841 One of the common uses of the historical *ex* editor is over slow network connections. Editors  
15842 that run in canonical mode can require far less traffic to and from, and far less processing on, the  
15843 host machine, as well as more easily supporting block-mode terminals. For these reasons,  
15844 IEEE Std 1003.1-2001 requires that *ex* be implemented using canonical mode input processing, as  
15845 was done historically.  
15846 IEEE Std 1003.1-2001 does not require the historical 4 BSD input editing characters “word erase”  
15847 or “literal next”. For this reason, it is unspecified how they are handled by *ex*, although they  
15848 must have the required effect. Implementations that resolve them after the line has been ended  
15849 using a <newline> or <control>-M character, and implementations that rely on the underlying  
15850 system terminal support for this processing, are both conforming. Implementations are strongly  
15851 urged to use the underlying system functionality, if at all possible, for compatibility with other  
15852 system text input interfaces.  
15853 Historically, when the **eof** character was used to decrement the **autoindent** level, the cursor  
15854 moved to display the new end of the **autoindent** characters, but did not move the cursor to a  
15855 new line, nor did it erase the <control>-D character from the line. IEEE Std 1003.1-2001 does not  
15856 specify that the cursor remain on the same line or that the rest of the line is erased; however,  
15857 implementations are strongly encouraged to provide the best possible user interface; that is, the  
15858 cursor should remain on the same line, and any <control>-D character on the line should be  
15859 erased.  
15860 IEEE Std 1003.1-2001 does not require the historical 4 BSD input editing character “reprint”,  
15861 traditionally <control>-R, which redisplayed the current input from the user. For this reason,  
15862 and because the functionality cannot be implemented after the line has been terminated by the  
15863 user, IEEE Std 1003.1-2001 makes no requirements about this functionality. Implementations are  
15864 strongly urged to make this historical functionality available, if possible.  
15865 Historically, <control>-Q did not perform a literal next function in *ex*, as it did in *vi*.  
15866 IEEE Std 1003.1-2001 requires conformance to historical practice to avoid breaking historical *ex*  
15867 scripts and **.exrc** files.

|       |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
|-------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 15868 | <b>eof</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| 15869 | Whether the <i>eof</i> character immediately modifies the <b>autoindent</b> characters in the prompt is left unspecified so that implementations can conform in the presence of systems that do not support this functionality. Implementations are encouraged to modify the line and redisplay it immediately, if possible.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| 15870 |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| 15871 |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| 15872 |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| 15873 | The specification of the handling of the <i>eof</i> character differs from historical practice only in that <i>eof</i> characters are not discarded if they follow normal characters in the text input. Historically, they were always discarded.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
| 15874 |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| 15875 |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| 15876 | <b>Command Descriptions in ex</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
| 15877 | Historically, several commands (for example, <b>global</b> , <b>v</b> , <b>visual</b> , <b>s</b> , <b>write</b> , <b>wq</b> , <b>yank</b> , <b>!</b> , <b>&lt;</b> , <b>&gt;</b> , <b>&amp;</b> , and <b>¬</b> ) were executable in empty files (that is, the default address(es) were 0), or permitted explicit addresses of 0 (for example, 0 was a valid address, or 0,0 was a valid range). Addresses of 0, or command execution in an empty file, make sense only for commands that add new text to the edit buffer or write commands (because users may wish to write empty files). IEEE Std 1003.1-2001 requires this behavior for such commands and disallows it otherwise, for consistency and simplicity of specification.                                                                                                                                                                                      |
| 15878 |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| 15879 |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| 15880 |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| 15881 |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| 15882 |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| 15883 |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| 15884 | A count to an <i>ex</i> command has been historically corrected to be no greater than the last line in a file; for example, in a five-line file, the command <b>1,6print</b> would fail, but the command <b>1print300</b> would succeed. IEEE Std 1003.1-2001 requires conformance to historical practice.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| 15885 |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| 15886 |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| 15887 | Historically, the use of flags in <i>ex</i> commands could be obscure. General historical practice was as described by IEEE Std 1003.1-2001, but there were some special cases. For instance, the <b>list</b> , <b>number</b> , and <b>print</b> commands ignored trailing address offsets; for example, <b>3p +++#</b> would display line 3, and 3 would be the current line after the execution of the command. The <b>open</b> and <b>visual</b> commands ignored both the trailing offsets and the trailing flags. Also, flags specified to the <b>open</b> and <b>visual</b> commands interacted badly with the <b>list</b> edit option, and setting and then unsetting it during the open/visual session would cause <i>vi</i> to stop displaying lines in the specified format. For consistency and simplicity of specification, IEEE Std 1003.1-2001 does not permit any of these exceptions to the general rule. |
| 15888 |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| 15889 |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| 15890 |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| 15891 |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| 15892 |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| 15893 |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| 15894 |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| 15895 |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| 15896 | IEEE Std 1003.1-2001 uses the word <i>copy</i> in several places when discussing buffers. This is not intended to imply implementation.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
| 15897 |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| 15898 | Historically, <i>ex</i> users could not specify numeric buffers because of the ambiguity this would cause; for example, in the command <b>3 delete 2</b> , it is unclear whether 2 is a buffer name or a <i>count</i> . IEEE Std 1003.1-2001 requires conformance to historical practice by default, but does not preclude extensions.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
| 15899 |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| 15900 |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| 15901 |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| 15902 | Historically, the contents of the unnamed buffer were frequently discarded after commands that did not explicitly affect it; for example, when using the <b>edit</b> command to switch files. For consistency and simplicity of specification, IEEE Std 1003.1-2001 does not permit this behavior.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| 15903 |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| 15904 |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| 15905 | The <i>ex</i> utility did not historically have access to the numeric buffers, and, furthermore, deleting lines in <i>ex</i> did not modify their contents. For example, if, after doing a delete in <i>vi</i> , the user switched to <i>ex</i> , did another delete, and then switched back to <i>vi</i> , the contents of the numeric buffers would not have changed. IEEE Std 1003.1-2001 requires conformance to historical practice. Numeric buffers are described in the <i>ex</i> utility in order to confine the description of buffers to a single location in IEEE Std 1003.1-2001.                                                                                                                                                                                                                                                                                                                             |
| 15906 |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| 15907 |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| 15908 |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| 15909 |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| 15910 |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| 15911 | The metacharacters that trigger shell expansion in <i>file</i> arguments match historical practice, as does the method for doing shell expansion. Implementations wishing to provide users with the flexibility to alter the set of metacharacters are encouraged to provide a <b>shellmeta</b> string edit                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
| 15912 |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| 15913 |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |

15914 option.

15915 Historically, ex commands executed from vi refreshed the screen when it did not strictly need to  
15916 do so; for example, **!date > /dev/null** does not require a screen refresh because the output of the  
15917 UNIX *date* command requires only a single line of the screen. IEEE Std 1003.1-2001 requires that  
15918 the screen be refreshed if it has been overwritten, but makes no requirements as to how an  
15919 implementation should make that determination. Implementations may prompt and refresh the  
15920 screen regardless.

### 15921 Abbreviate

15922 Historical practice was that characters that were entered as part of an abbreviation replacement  
15923 were subject to **map** expansions, the **showmatch** edit option, further abbreviation expansions,  
15924 and so on; that is, they were logically pushed onto the terminal input queue, and were not a  
15925 simple replacement. IEEE Std 1003.1-2001 requires conformance to historical practice. Historical  
15926 practice was that whenever a non-word character (that had not been escaped by a <control>-V)  
15927 was entered after a word character, vi would check for abbreviations. The check was based on  
15928 the type of the character entered before the word character of the word/non-word pair that  
15929 triggered the check. The word character of the word/non-word pair that triggered the check and  
15930 all characters entered before the trigger pair that were of that type were included in the check,  
15931 with the exception of <blank>s, which always delimited the abbreviation.

15932 This means that, for the abbreviation to work, the *lhs* must end with a word character, there can  
15933 be no transitions from word to non-word characters (or *vice versa*) other than between the last  
15934 and next-to-last characters in the *lhs*, and there can be no <blank>s in the *lhs*. In addition,  
15935 because of the historical quoting rules, it was impossible to enter a literal <control>-V in the *lhs*.  
15936 IEEE Std 1003.1-2001 requires conformance to historical practice. Historical implementations did  
15937 not inform users when abbreviations that could never be used were entered; implementations  
15938 are strongly encouraged to do so.

15939 For example, the following abbreviations will work:

15940 :ab (p REPLACE  
15941 :ab p REPLACE  
15942 :ab ((p REPLACE

15943 The following abbreviations will not work:

15944 :ab ( REPLACE  
15945 :ab (pp REPLACE

15946 Historical practice is that words on the vi colon command line were subject to abbreviation  
15947 expansion, including the arguments to the **abbrev** (and more interestingly) the **unabbrev**  
15948 command. Because there are implementations that do not do abbreviation expansion for the first  
15949 argument to those commands, this is permitted, but not required, by IEEE Std 1003.1-2001.  
15950 However, the following sequence:

15951 :ab foo bar  
15952 :ab foo baz

15953 resulted in the addition of an abbreviation of "baz" for the string "bar" in historical ex/vi, and  
15954 the sequence:

15955 :ab fool bar  
15956 :ab foo2 bar  
15957 :unabbreviate foo2

15958 deleted the abbreviation "foo1", not "foo2". These behaviors are not permitted by  
15959 IEEE Std 1003.1-2001 because they clearly violate the expectations of the user.

15960 It was historical practice that <control>-V, not backslash, characters be interpreted as escaping  
15961 subsequent characters in the **abbreviate** command. IEEE Std 1003.1-2001 requires conformance  
15962 to historical practice; however, it should be noted that an abbreviation containing a <blank> will  
15963 never work.

#### 15964 Append

15965 Historically, any text following a vertical-line command separator after an **append**, **change**, or  
15966 **insert** command became part of the insert text. For example, in the command:

15967 :g/pattern/append|stuff1

15968 a line containing the text "stuff1" would be appended to each line matching pattern. It was  
15969 also historically valid to enter:

15970 :append|stuff1

15971 stuff2

15972 .

15973 and the text on the *ex* command line would be appended along with the text inserted after it.  
15974 There was an historical bug, however, that the user had to enter two terminating lines (the ' . ' lines)  
15975 to terminate text input mode in this case. IEEE Std 1003.1-2001 requires conformance to  
15976 historical practice, but disallows the historical need for multiple terminating lines.

#### 15977 Change

15978 See the RATIONALE for the **append** command. Historical practice for cursor positioning after  
15979 the change command when no text is input, is as described in IEEE Std 1003.1-2001. However,  
15980 one System V implementation is known to have been modified such that the cursor is positioned  
15981 on the first address specified, and not on the line before the first address. IEEE Std 1003.1-2001  
15982 disallows this modification for consistency.

15983 Historically, the **change** command did not support buffer arguments, although some  
15984 implementations allow the specification of an optional buffer. This behavior is neither required  
15985 nor disallowed by IEEE Std 1003.1-2001.

#### 15986 Change Directory

15987 A common extension in *ex* implementations is to use the elements of a **cdpath** edit option as  
15988 prefix directories for *path* arguments to **chdir** that are relative pathnames and that do not have  
15989 ' .' or " .. " as their first component. Elements in the **cdpath** edit option are colon-separated.  
15990 The initial value of the **cdpath** edit option is the value of the shell *CDPATH* environment  
15991 variable. This feature was not included in IEEE Std 1003.1-2001 because it does not exist in any  
15992 of the implementations considered historical practice.

#### 15993 Copy

15994 Historical implementations of *ex* permitted copies to lines inside of the specified range; for  
15995 example, :2,5copy3 was a valid command. IEEE Std 1003.1-2001 requires conformance to  
15996 historical practice.

15997

**Delete**

15998

IEEE Std 1003.1-2001 requires support for the historical parsing of a **delete** command followed by flags, without any intervening <blank>s. For example:

16000

**1dp** Deletes the first line and prints the line that was second.

16001

**1deleP** As for **1dp**.

16002

**1d** Deletes the first line, saving it in buffer *p*.

16003

**1d p1l** (Pee-one-ell.) Deletes the first line, saving it in buffer *p*, and listing the line that was second.

16004

**Edit**

16005

Historically, any *ex* command could be entered as a *+command* argument to the **edit** command, although some (for example, **insert** and **append**) were known to confuse historical implementations. For consistency and simplicity of specification, IEEE Std 1003.1-2001 requires that any command be supported as an argument to the **edit** command.

16010

Historically, the command argument was executed with the current line set to the last line of the file, regardless of whether the **edit** command was executed from visual mode or not. IEEE Std 1003.1-2001 requires conformance to historical practice.

16013

Historically, the *+command* specified to the **edit** and **next** commands was delimited by the first <blank>, and there was no way to quote them. For consistency, IEEE Std 1003.1-2001 requires that the usual *ex* backslash quoting be provided.

16016

Historically, specifying the *+command* argument to the **edit** command required a filename to be specified as well; for example, **:edit +100** would always fail. For consistency and simplicity of specification, IEEE Std 1003.1-2001 does not permit this usage to fail for that reason.

16019

Historically, only the cursor position of the last file edited was remembered by the editor. IEEE Std 1003.1-2001 requires that this be supported; however, implementations are permitted to remember and restore the cursor position for any file previously edited.

16022

**File**

16023

Historical versions of the *ex* editor **file** command displayed a current line and number of lines in the edit buffer of 0 when the file was empty, while the *vi* <control>-G command displayed a current line and number of lines in the edit buffer of 1 in the same situation. IEEE Std 1003.1-2001 does not permit this discrepancy, instead requiring that a message be displayed indicating that the file is empty.

16028

**Global**

16029

The two-pass operation of the **global** and **v** commands is not intended to imply implementation, only the required result of the operation.

16031

The current line and column are set as specified for the individual *ex* commands. This requirement is cumulative; that is, the current line and column must track across all the commands executed by the **global** or **v** commands.

16034

**Insert**

16035

See the RATIONALE for the **append** command.

16036

Historically, **insert** could not be used with an address of zero; that is, not when the edit buffer was empty. IEEE Std 1003.1-2001 requires that this command behave consistently with the **append** command.

16039

**Join**

16040

The action of the **join** command in relation to the special characters is only defined for the POSIX locale because the correct amount of white space after a period varies; in Japanese none is required, in French only a single space, and so on.

16043

**List**

16044

The historical output of the **list** command was potentially ambiguous. The standard developers believed correcting this to be more important than adhering to historical practice, and IEEE Std 1003.1-2001 requires unambiguous output.

16047

**Map**

16048

Historically, command mode maps only applied to command names; for example, if the character 'x' was mapped to 'y', the command fx searched for the 'x' character, not the 'y' character. IEEE Std 1003.1-2001 requires this behavior. Historically, entering <control>-V as the first character of a vi command was an error. Several implementations have extended the semantics of vi such that <control>-V means that the subsequent command character is not mapped. This is permitted, but not required, by IEEE Std 1003.1-2001. Regardless, using <control>-V to escape the second or later character in a sequence of characters that might match a **map** command, or any character in text input mode, is historical practice, and stops the entered keys from matching a map. IEEE Std 1003.1-2001 requires conformance to historical practice.

16057

Historical implementations permitted digits to be used as a **map** command *lhs*, but then ignored the map. IEEE Std 1003.1-2001 requires that the mapped digits not be ignored.

16059

The historical implementation of the **map** command did not permit **map** commands that were more than a single character in length if the first character was printable. This behavior is permitted, but not required, by IEEE Std 1003.1-2001.

16062

Historically, mapped characters were remapped unless the **remap** edit option was not set, or the prefix of the mapped characters matched the mapping characters; for example, in the **map**:

16064

```
:map ab abcd
```

16065

the characters "ab" were used as is and were not remapped, but the characters "cd" were mapped if appropriate. This can cause infinite loops in the vi mapping mechanisms. IEEE Std 1003.1-2001 requires conformance to historical practice, and that such loops be interruptible.

16069

Text input maps had the same problems with expanding the *lhs* for the ex **map!** and **unmap!** command as did the ex **abbreviate** and **unabbreviate** commands. See the RATIONALE for the ex **abbreviate** command. IEEE Std 1003.1-2001 requires similar modification of some historical practice for the **map** and **unmap** commands, as described for the **abbreviate** and **unabbreviate** commands.

16074

Historically, **maps** that were subsets of other **maps** behaved differently depending on the order in which they were defined. For example:

16076 :map! ab short  
16077 :map! abc long

16078 would always translate the characters "ab" to "short", regardless of how fast the characters  
16079 "abc" were entered. If the entry order was reversed:

16080 :map! abc long  
16081 :map! ab short

16082 the characters "ab" would cause the editor to pause, waiting for the completing 'c' character,  
16083 and the characters might never be mapped to "short". For consistency and simplicity of  
16084 specification, IEEE Std 1003.1-2001 requires that the shortest match be used at all times.

16085 The length of time the editor spends waiting for the characters to complete the *lhs* is unspecified  
16086 because the timing capabilities of systems are often inexact and variable, and it may depend on  
16087 other factors such as the speed of the connection. The time should be long enough for the user to  
16088 be able to complete the sequence, but not long enough for the user to have to wait. Some  
16089 implementations of vi have added a **keytime** option, which permits users to set the number of  
16090 0.1 seconds the editor waits for the completing characters. Because mapped terminal function  
16091 and cursor keys tend to start with an <ESC> character, and <ESC> is the key ending vi text input  
16092 mode, **maps** starting with <ESC> characters are generally exempted from this timeout period,  
16093 or, at least timed out differently.

## 16094 **Mark**

16095 Historically, users were able to set the "previous context" marks explicitly. In addition, the **ex**  
16096 commands " and " and the **vi** commands ", ", and " all referred to the same mark. In addition,  
16097 the previous context marks were not set if the command, with which the address setting the  
16098 mark was associated, failed. IEEE Std 1003.1-2001 requires conformance to historical practice.  
16099 Historically, if marked lines were deleted, the mark was also deleted, but would reappear if the  
16100 change was undone. IEEE Std 1003.1-2001 requires conformance to historical practice.

16101 The description of the special events that set the ' and ' marks matches historical practice. For  
16102 example, historically the command /a/,/b/ did not set the ' and ' marks, but the command  
16103 /a/,/b/delete did.

## 16104 **Next**

16105 Historically, any **ex** command could be entered as a *+command* argument to the **next** command,  
16106 although some (for example, **insert** and **append**) were known to confuse historical  
16107 implementations. IEEE Std 1003.1-2001 requires that any command be permitted and that it  
16108 behave as specified. The **next** command can accept more than one file, so usage such as:

16109 next `ls [abc] '

16110 is valid; it need not be valid for the **edit** or **read** commands, for example, because they expect  
16111 only one filename.

16112 Historically, the **next** command behaved differently from the **:rewind** command in that it  
16113 ignored the force flag if the **autowrite** flag was set. For consistency, IEEE Std 1003.1-2001 does  
16114 not permit this behavior.

16115 Historically, the **next** command positioned the cursor as if the file had never been edited before,  
16116 regardless. IEEE Std 1003.1-2001 does not permit this behavior, for consistency with the **edit**  
16117 command.

16118 Implementations wanting to provide a counterpart to the **next** command that edited the  
16119 previous file have used the command **prev[ious]**, which takes no *file* argument.

16120 IEEE Std 1003.1-2001 does not require this command.

### 16121 Open

16122 Historically, the **open** command would fail if the **open** edit option was not set.  
16123 IEEE Std 1003.1-2001 does not mention the **open** edit option and does not require this behavior.  
16124 Some historical implementations do not permit entering open mode from open or visual mode,  
16125 only from ex mode. For consistency, IEEE Std 1003.1-2001 does not permit this behavior.

16126 Historically, entering open mode from the command line (that is, **vi +open**) resulted in  
16127 anomalous behaviors; for example, the **ex** file and **set** commands, and the **vi** command  
16128 <control>-G did not work. For consistency, IEEE Std 1003.1-2001 does not permit this behavior.

16129 Historically, the **open** command only permitted ' / ' characters to be used as the search pattern  
16130 delimiter. For consistency, IEEE Std 1003.1-2001 requires that the search delimiters used by the **s**,  
16131 **global**, and **v** commands be accepted as well.

### 16132 Preserve

16133 The **preserve** command does not historically cause the file to be considered unmodified for the  
16134 purposes of future commands that may exit the editor. IEEE Std 1003.1-2001 requires  
16135 conformance to historical practice.

16136 Historical documentation stated that mail was not sent to the user when **preserve** was executed;  
16137 however, historical implementations did send mail in this case. IEEE Std 1003.1-2001 requires  
16138 conformance to the historical implementations.

### 16139 Print

16140 The writing of NUL by the **print** command is not specified as a special case because the standard  
16141 developers did not want to require **ex** to support NUL characters. Historically, characters were  
16142 displayed using the ARPA standard mappings, which are as follows:

- 16143 1. Printable characters are left alone.
- 16144 2. Control characters less than \177 are represented as '^' followed by the character offset  
16145 from the '@' character in the ASCII map; for example, \007 is represented as '^G'.
- 16146 3. \177 is represented as '^' followed by '?'.

16147 The display of characters having their eighth bit set was less standard. Existing implementations  
16148 use hex (0x00), octal (\000), and a meta-bit display. (The latter displayed bytes that had their  
16149 eighth bit set as the two characters "M-" followed by the seven-bit display as described above.)  
16150 The latter probably has the best claim to historical practice because it was used for the -v option  
16151 of 4 BSD and 4 BSD-derived versions of the **cat** utility since 1980.

16152 No specific display format is required by IEEE Std 1003.1-2001.

16153 Explicit dependence on the ASCII character set has been avoided where possible, hence the use  
16154 of the phrase an "implementation-defined multi-character sequence" for the display of non-  
16155 printable characters in preference to the historical usage of, for instance, "^I" for the <tab>. Implementations  
16156 are encouraged to conform to historical practice in the absence of any strong  
16157 reason to diverge.

16158 Historically, all **ex** commands beginning with the letter 'p' could be entered using capitalized  
16159 versions of the commands; for example, **P[rint]**, **Pre[serve]**, and **Pu[t]** were all valid command  
16160 names. IEEE Std 1003.1-2001 permits, but does not require, this historical practice because  
16161 capital forms of the commands are used by some implementations for other purposes.

16162

**Put**

16163

Historically, an **ex put** command, executed from open or visual mode, was the same as the open or visual mode **P** command, if the buffer was named and was cut in character mode, and the same as the **p** command if the buffer was named and cut in line mode. If the unnamed buffer was the source of the text, the entire line from which the text was taken was usually **put**, and the buffer was handled as if in line mode, but it was possible to get extremely anomalous behavior. In addition, using the **Q** command to switch into **ex** mode, and then doing a **put** often resulted in errors as well, such as appending text that was unrelated to the (supposed) contents of the buffer. For consistency and simplicity of specification, IEEE Std 1003.1-2001 does not permit these behaviors. All **ex put** commands are required to operate in line mode, and the contents of the buffers are not altered by changing the mode of the editor.

16164

16165

16166

16167

16168

16169

16170

16171

16172

16173

**Read**

16174

16175

16176

16177

16178

Historically, an **ex read** command executed from open or visual mode, executed in an empty file, left an empty line as the first line of the file. For consistency and simplicity of specification, IEEE Std 1003.1-2001 does not permit this behavior. Historically, a **read** in open or visual mode from a program left the cursor at the last line read in, not the first. For consistency, IEEE Std 1003.1-2001 does not permit this behavior.

16179

16180

Historical implementations of **ex** were unable to undo **read** commands that read from the output of a program. For consistency, IEEE Std 1003.1-2001 does not permit this behavior.

16181

16182

16183

16184

16185

Historically, the **ex** and **vi** message after a successful **read** or **write** command specified “characters”, not “bytes”. IEEE Std 1003.1-2001 requires that the number of bytes be displayed, not the number of characters, because it may be difficult in multi-byte implementations to determine the number of characters read. Implementations are encouraged to clarify the message displayed to the user.

16186

16187

16188

16189

Historically, reads were not permitted on files other than type regular, except that FIFO files could be read (probably only because they did not exist when **ex** and **vi** were originally written). Because the historical **ex** evaluated **read!** and **read !** equivalently, there can be no optional way to force the read. IEEE Std 1003.1-2001 permits, but does not require, this behavior.

16190

**Recover**

16191

16192

16193

16194

Some historical implementations of the editor permitted users to recover the edit buffer contents from a previous edit session, and then exit without saving those contents (or explicitly discarding them). The intent of IEEE Std 1003.1-2001 in requiring that the edit buffer be treated as already modified is to prevent this user error.

16195

**Rewind**

16196

16197

16198

Historical implementations supported the **rewind** command when the user was editing the first file in the list; that is, the file that the **rewind** command would edit. IEEE Std 1003.1-2001 requires conformance to historical practice.

16199        **Substitute**

16200        Historically, *ex* accepted an **r** option to the **s** command. The effect of the **r** option was to use the  
16201        last regular expression used in any command as the pattern, the same as the **~** command. The **r**  
16202        option is not required by IEEE Std 1003.1-2001. Historically, the **c** and **g** options were toggled; for  
16203        example, the command **:s/abc/def/** was the same as **s/abc/def/ccccgggg**. For simplicity of  
16204        specification, IEEE Std 1003.1-2001 does not permit this behavior.

16205        The tilde command is often used to replace the last search RE. For example, in the sequence:

16206        **s/red/blue/**  
16207        **/green**  
16208        **~**

16209        the **~** command is equivalent to:

16210        **s/green/blue/**

16211        Historically, *ex* accepted all of the following forms:

16212        **s/abc/def/**  
16213        **s/abc/def**  
16214        **s/abc/**  
16215        **s/abc**

16216        IEEE Std 1003.1-2001 requires conformance to this historical practice.

16217        The **s** command presumes that the **'^'** character only occupies a single column in the display.  
16218        Much of the *ex* and *vi* specification presumes that the **<space>** only occupies a single column in  
16219        the display. There are no known character sets for which this is not true.

16220        Historically, the final column position for the substitute commands was based on previous  
16221        column movements; a search for a pattern followed by a substitution would leave the column  
16222        position unchanged, while a **0** command followed by a substitution would change the column  
16223        position to the first non-<blank>. For consistency and simplicity of specification,  
16224        IEEE Std 1003.1-2001 requires that the final column position always be set to the first non-  
16225        <blank>.

16226        **Set**

16227        Historical implementations redisplayed all of the options for each occurrence of the **all** keyword.  
16228        IEEE Std 1003.1-2001 permits, but does not require, this behavior.

16229        **Tag**

16230        No requirement is made as to where *ex* and *vi* shall look for the file referenced by the tag entry.  
16231        Historical practice has been to look for the path found in the **tags** file, based on the current  
16232        directory. A useful extension found in some implementations is to look based on the directory  
16233        containing the **tags** file that held the entry, as well. No requirement is made as to which  
16234        reference for the tag in the **tags** file is used. This is deliberate, in order to permit extensions such  
16235        as multiple entries in a **tags** file for a tag.

16236        Because users often specify many different **tags** files, some of which need not be relevant or exist  
16237        at any particular time, IEEE Std 1003.1-2001 requires that error messages about problem **tags**  
16238        files be displayed only if the requested tag is not found, and then, only once for each time that  
16239        the **tag** edit option is changed.

16240        The requirement that the current edit buffer be unmodified is only necessary if the file indicated  
16241        by the **tag** entry is not the same as the current file (as defined by the current pathname).

16242 Historically, the file would be reloaded if the filename had changed, as well as if the filename  
16243 was different from the current pathname. For consistency and simplicity of specification,  
16244 IEEE Std 1003.1-2001 does not permit this behavior, requiring that the name be the only factor in  
16245 the decision.

16246 Historically, *vi* only searched for tags in the current file from the current cursor to the end of the  
16247 file, and therefore, if the **wrapscan** option was not set, tags occurring before the current cursor  
16248 were not found. IEEE Std 1003.1-2001 considers this a bug, and implementations are required to  
16249 search for the first occurrence in the file, regardless.

## 16250 Undo

16251 The **undo** description deliberately uses the word “modified”. The **undo** command is not  
16252 intended to undo commands that replace the contents of the edit buffer, such as **edit**, **next**, **tag**,  
16253 or **recover**.

16254 Cursor positioning after the **undo** command was inconsistent in the historical *vi*, sometimes  
16255 attempting to restore the original cursor position (**global**, **undo**, and **v** commands), and  
16256 sometimes, in the presence of maps, placing the cursor on the last line added or changed instead  
16257 of the first. IEEE Std 1003.1-2001 requires a simplified behavior for consistency and simplicity of  
16258 specification.

## 16259 Version

16260 The **version** command cannot be exactly specified since there is no widely-accepted definition of  
16261 what the version information should contain. Implementations are encouraged to do something  
16262 reasonably intelligent.

## 16263 Write

16264 Historically, the **ex** and *vi* message after a successful **read** or **write** command specified  
16265 “characters”, not “bytes”. IEEE Std 1003.1-2001 requires that the number of bytes be displayed,  
16266 not the number of characters because it may be difficult in multi-byte implementations to  
16267 determine the number of characters written. Implementations are encouraged to clarify the  
16268 message displayed to the user.

16269 Implementation-defined tests are permitted so that implementations can make additional  
16270 checks; for example, for locks or file modification times.

16271 Historically, attempting to append to a nonexistent file caused an error. It has been left  
16272 unspecified in IEEE Std 1003.1-2001 to permit implementations to let the **write** succeed, so that  
16273 the append semantics are similar to those of the historical *csh*.

16274 Historical *vi* permitted empty edit buffers to be written. However, since the way *vi* got around  
16275 dealing with “empty” files was to always have a line in the edit buffer, no matter what, it wrote  
16276 them as files of a single, empty line. IEEE Std 1003.1-2001 does not permit this behavior.

16277 Historically, **ex** restored standard output and standard error to their values as of when **ex** was  
16278 invoked, before writes to programs were performed. This could disturb the terminal  
16279 configuration as well as be a security issue for some terminals. IEEE Std 1003.1-2001 does not  
16280 permit this, requiring that the program output be captured and displayed as if by the **ex print**  
16281 command.

- 16282      **Adjust Window**
- 16283      Historically, the line count was set to the value of the **scroll** option if the type character was  
16284      end-of-file. This feature was broken on most historical implementations long ago, however, and  
16285      is not documented anywhere. For this reason, IEEE Std 1003.1-2001 is resolutely silent.
- 16286      Historically, the **z** command was <blank>-sensitive and **z +** and **z -** did different things than **z+**  
16287      and **z-** because the type could not be distinguished from a flag. (The commands **z .** and **z =**  
16288      were historically invalid.) IEEE Std 1003.1-2001 requires conformance to this historical practice.
- 16289      Historically, the **z** command was further <blank>-sensitive in that the *count* could not be  
16290      <blank>-delimited; for example, the commands **z= 5** and **z- 5** were also invalid. Because the  
16291      *count* is not ambiguous with respect to either the type character or the flags, this is not permitted  
16292      by IEEE Std 1003.1-2001.
- 16293      **Escape**
- 16294      Historically, **ex** filter commands only read the standard output of the commands, letting  
16295      standard error appear on the terminal as usual. The **vi** utility, however, read both standard  
16296      output and standard error. IEEE Std 1003.1-2001 requires the latter behavior for both **ex** and **vi**,  
16297      for consistency.
- 16298      **Shift Left and Shift Right**
- 16299      Historically, it was possible to add shift characters to increase the effect of the command; for  
16300      example, <<< outdented (or >>> indented) the lines 3 levels of indentation instead of the default  
16301      1. IEEE Std 1003.1-2001 requires conformance to historical practice.
- 16302      **<control>-D**
- 16303      Historically, the <control>-D command erased the prompt, providing the user with an unbroken  
16304      presentation of lines from the edit buffer. This is not required by IEEE Std 1003.1-2001;  
16305      implementations are encouraged to provide it if possible. Historically, the <control>-D  
16306      command took, and then ignored, a *count*. IEEE Std 1003.1-2001 does not permit this behavior.
- 16307      **Write Line Number**
- 16308      Historically, the **ex =** command, when executed in **ex** mode in an empty edit buffer, reported 0,  
16309      and from open or visual mode, reported 1. For consistency and simplicity of specification,  
16310      IEEE Std 1003.1-2001 does not permit this behavior.
- 16311      **Execute**
- 16312      Historically, **ex** did not correctly handle the inclusion of text input commands (that is, **append**,  
16313      **insert**, and **change**) in executed buffers. IEEE Std 1003.1-2001 does not permit this exclusion for  
16314      consistency.
- 16315      Historically, the logical contents of the buffer being executed did not change if the buffer itself  
16316      were modified by the commands being executed; that is, buffer execution did not support self-  
16317      modifying code. IEEE Std 1003.1-2001 requires conformance to historical practice.
- 16318      Historically, the **@** command took a range of lines, and the **@** buffer was executed once per line,  
16319      with the current line ('.' ) set to each specified line. IEEE Std 1003.1-2001 requires conformance to  
16320      historical practice.
- 16321      Some historical implementations did not notice if errors occurred during buffer execution. This,  
16322      coupled with the ability to specify a range of lines for the **ex @** command, makes it trivial to  
16323      cause them to drop **core**. IEEE Std 1003.1-2001 requires that implementations stop buffer

16324 execution if any error occurs, if the specified line doesn't exist, or if the contents of the edit buffer  
16325 itself are replaced (for example, the buffer executes the **ex :edit** command).

## 16326 Regular Expressions in ex

16327 Historical practice is that the characters in the replacement part of the last **s** command—that is,  
16328 those matched by entering a '`~`' in the regular expression—were not further expanded by the  
16329 regular expression engine. So, if the characters contained the string "`a. ,`" they would match  
16330 '`a`' followed by '`. ,`' and not '`a`' followed by any character. IEEE Std 1003.1-2001 requires  
16331 conformance to historical practice.

## 16332 Edit Options in ex

16333 The following paragraphs describe the historical behavior of some edit options that were not, for  
16334 whatever reason, included in IEEE Std 1003.1-2001. Implementations are strongly encouraged to  
16335 only use these names if the functionality described here is fully supported.

- |                        |                                                                                                                                                                                                                                                                                                                                                                                           |
|------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 16336 <b>extended</b>  | The <b>extended</b> edit option has been used in some implementations of <b>vi</b> to provide<br>16337 extended regular expressions instead of basic regular expressions. This option was<br>16338 omitted from IEEE Std 1003.1-2001 because it is not widespread historical practice.                                                                                                    |
| 16339 <b>flash</b>     | The <b>flash</b> edit option historically caused the screen to flash instead of beeping on<br>16340 error. This option was omitted from IEEE Std 1003.1-2001 because it is not found in<br>16341 some historical implementations.                                                                                                                                                         |
| 16342 <b>hardtabs</b>  | The <b>hardtabs</b> edit option historically defined the number of columns between<br>16343 hardware tab settings. This option was omitted from IEEE Std 1003.1-2001 because<br>16344 it was believed to no longer be generally useful.                                                                                                                                                   |
| 16345 <b>modeline</b>  | The <b>modeline</b> (sometimes named <b>modelines</b> ) edit option historically caused <b>ex</b> or<br>16346 <b>vi</b> to read the five first and last lines of the file for editor commands. This option is<br>16347 a security problem, and vendors are strongly encouraged to delete it from<br>16348 historical implementations.                                                     |
| 16349 <b>open</b>      | The <b>open</b> edit option historically disallowed the <b>ex open</b> and <b>visual</b> commands.<br>16350 This edit option was omitted because these commands are required by<br>16351 IEEE Std 1003.1-2001.                                                                                                                                                                            |
| 16352 <b>optimize</b>  | The <b>optimize</b> edit option historically expedited text throughput by setting the<br>16353 terminal to not do automatic <carriage-return>s when printing more than one<br>16354 logical line of output. This option was omitted from IEEE Std 1003.1-2001 because<br>16355 it was intended for terminals without addressable cursors, which are rarely, if ever,<br>16356 still used. |
| 16357 <b>ruler</b>     | The <b>ruler</b> edit option has been used in some implementations of <b>vi</b> to present a<br>16358 current row/column ruler for the user. This option was omitted from<br>16359 IEEE Std 1003.1-2001 because it is not widespread historical practice.                                                                                                                                 |
| 16360 <b>sourceany</b> | The <b>sourceany</b> edit option historically caused <b>ex</b> or <b>vi</b> to source start-up files that<br>16361 were owned by users other than the user running the editor. This option is a<br>16362 security problem, and vendors are strongly encouraged to remove it from their<br>16363 implementations.                                                                          |
| 16364 <b>timeout</b>   | The <b>timeout</b> edit option historically enabled the (now standard) feature of only<br>16365 waiting for a short period before returning keys that could be part of a macro. This<br>16366 feature was omitted from IEEE Std 1003.1-2001 because its behavior is now<br>16367 standard, it is not widely useful, and it was rarely documented.                                         |

|       |                       |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
|-------|-----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 16368 | <b>verbose</b>        | The <b>verbose</b> edit option has been used in some implementations of <i>vi</i> to cause <i>vi</i> to output error messages for common errors; for example, attempting to move the cursor past the beginning or end of the line instead of only alerting the screen. (The historical <i>vi</i> only alerted the terminal and presented no message for such errors. The historical editor option <b>terse</b> did not select when to present error messages, it only made existing error messages more or less verbose.) This option was omitted from IEEE Std 1003.1-2001 because it is not widespread historical practice; however, implementors are encouraged to use it if they wish to provide error messages for naive users. |
| 16377 | <b>wraplen</b>        | The <b>wraplen</b> edit option has been used in some implementations of <i>vi</i> to specify an automatic margin measured from the left margin instead of from the right margin. This is useful when multiple screen sizes are being used to edit a single file. This option was omitted from IEEE Std 1003.1-2001 because it is not widespread historical practice; however, implementors are encouraged to use it if they add this functionality.                                                                                                                                                                                                                                                                                  |
| 16383 | <b>autoindent, ai</b> | Historically, the command <b>0a</b> did not do any autoindentation, regardless of the current indentation of line 1. IEEE Std 1003.1-2001 requires that any indentation present in line 1 be used.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
| 16386 | <b>autoprint, ap</b>  | Historically, the <b>autoprint</b> edit option was not completely consistent or based solely on modifications to the edit buffer. Exceptions were the <b>read</b> command (when reading from a file, but not from a filter), the <b>append</b> , <b>change</b> , <b>insert</b> , <b>global</b> , and <b>v</b> commands, all of which were not affected by <b>autoprint</b> , and the <b>tag</b> command, which was affected by <b>autoprint</b> . IEEE Std 1003.1-2001 requires conformance to historical practice.                                                                                                                                                                                                                  |
| 16392 |                       | Historically, the <b>autoprint</b> option only applied to the last of multiple commands entered using vertical-bar delimiters; for example, <b>delete &lt;newline&gt;</b> was affected by <b>autoprint</b> , but <b>delete   version &lt;newline&gt;</b> was not. IEEE Std 1003.1-2001 requires conformance to historical practice.                                                                                                                                                                                                                                                                                                                                                                                                  |
| 16396 | <b>autowrite, aw</b>  | Appending the '!' character to the <b>ex next</b> command to avoid performing an automatic write was not supported in historical implementations. IEEE Std 1003.1-2001 requires that the behavior match the other <b>ex</b> commands for consistency.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| 16400 | <b>ignorecase, ic</b> | Historical implementations of case-insensitive matching (the <b>ignorecase</b> edit option) lead to counterintuitive situations when uppercase characters were used in range expressions. Historically, the process was as follows:                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| 16404 | 1.                    | Take a line of text from the edit buffer.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
| 16405 | 2.                    | Convert uppercase to lowercase in text line.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
| 16406 | 3.                    | Convert uppercase to lowercase in regular expressions, except in character class specifications.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| 16408 | 4.                    | Match regular expressions against text.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| 16409 |                       | This would mean that, with <b>ignorecase</b> in effect, the text:                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |

16410       The cat sat on the mat  
16411       would be matched by  
16412       /^the/  
16413       but not by:  
16414       /^ [A-Z ]he/  
16415       For consistency with other commands implementing regular expressions, IEEE Std 1003.1-2001  
16416       does not permit this behavior.

#### 16417       **paragraphs, para**

16418       The ISO POSIX-2:1993 standard made the default **paragraphs** and **sections** edit options  
16419       implementation-defined, arguing they were historically oriented to the UNIX system *troff* text  
16420       formatter, and a “portable user” could use the {, }, [[, ]], (, and ) commands in open or visual  
16421       mode and have the cursor stop in unexpected places. IEEE Std 1003.1-2001 specifies their values  
16422       in the POSIX locale because the unusual grouping (they only work when grouped into two  
16423       characters at a time) means that they cannot be used for general-purpose movement, regardless.

#### 16424       **readonly**

16425       Implementations are encouraged to provide the best possible information to the user as to the  
16426       read-only status of the file, with the exception that they should not consider the current special  
16427       privileges of the process. This provides users with a safety net because they must force the  
16428       overwrite of read-only files, even when running with additional privileges.

16429       The **readonly** edit option specification largely conforms to historical practice. The only  
16430       difference is that historical implementations did not notice that the user had set the **readonly**  
16431       edit option in cases where the file was already marked read-only for some reason, and would  
16432       therefore reinitialize the **readonly** edit option the next time the contents of the edit buffer were  
16433       replaced. This behavior is disallowed by IEEE Std 1003.1-2001.

#### 16434       **report**

16435       The requirement that lines copied to a buffer interact differently than deleted lines is historical  
16436       practice. For example, if the **report** edit option is set to 3, deleting 3 lines will cause a report to be  
16437       written, but 4 lines must be copied before a report is written.

16438       The requirement that the **ex global**, **v**, **open**, **undo**, and **visual** commands present reports based  
16439       on the total number of lines added or deleted during the command execution, and that  
16440       commands executed by the **global** and **v** commands not present reports, is historical practice.  
16441       IEEE Std 1003.1-2001 extends historical practice by requiring that buffer execution be treated  
16442       similarly. The reasons for this are two-fold. Historically, only the report by the last command  
16443       executed from the buffer would be seen by the user, as each new report would overwrite the  
16444       last. In addition, the standard developers believed that buffer execution had more in common  
16445       with **global** and **v** commands than it did with other **ex** commands, and should behave similarly,  
16446       for consistency and simplicity of specification.

16447       **showmatch, sm**

16448       The length of time the cursor spends on the matching character is unspecified because the  
16449       timing capabilities of systems are often inexact and variable. The time should be long enough for  
16450       the user to notice, but not long enough for the user to become annoyed. Some implementations  
16451       of *vi* have added a **matchtime** option that permits users to set the number of 0.1 second intervals  
16452       the cursor pauses on the matching character.

16453       **showmode**

16454       The **showmode** option has been used in some historical implementations of *ex* and *vi* to display  
16455       the current editing mode when in open or visual mode. The editing modes have generally  
16456       included “command” and “input”, and sometimes other modes such as “replace” and  
16457       “change”. The string was usually displayed on the bottom line of the screen at the far right-hand  
16458       corner. In addition, a preceding ‘\*’ character often denoted whether the contents of the edit  
16459       buffer had been modified. The latter display has sometimes been part of the **showmode** option,  
16460       and sometimes based on another option. This option was not available in the 4 BSD historical  
16461       implementation of *vi*, but was viewed as generally useful, particularly to novice users, and is  
16462       required by IEEE Std 1003.1-2001.

16463       The **smd** shorthand for the **showmode** option was not present in all historical implementations  
16464       of the editor. IEEE Std 1003.1-2001 requires it, for consistency.

16465       Not all historical implementations of the editor displayed a mode string for command mode,  
16466       differentiating command mode from text input mode by the absence of a mode string.  
16467       IEEE Std 1003.1-2001 permits this behavior for consistency with historical practice, but  
16468       implementations are encouraged to provide a display string for both modes.

16469       **slowopen**

16470       Historically the **slowopen** option was automatically set if the terminal baud rate was less than  
16471       1 200 baud, or if the baud rate was 1 200 baud and the **redraw** option was not set. The **slowopen**  
16472       option had two effects. First, when inserting characters in the middle of a line, characters after  
16473       the cursor would not be pushed ahead, but would appear to be overwritten. Second, when  
16474       creating a new line of text, lines after the current line would not be scrolled down, but would  
16475       appear to be overwritten. In both cases, ending text input mode would cause the screen to be  
16476       refreshed to match the actual contents of the edit buffer. Finally, terminals that were sufficiently  
16477       intelligent caused the editor to ignore the **slowopen** option. IEEE Std 1003.1-2001 permits most  
16478       historical behavior, extending historical practice to require **slowopen** behaviors if the edit option  
16479       is set by the user.

16480       **tags**

16481       The default path for tags files is left unspecified as implementations may have their own **tags**  
16482       implementations that do not correspond to the historical ones. The default **tags** option value  
16483       should probably at least include the file **./tags**.

|       |                                                                                                                        |
|-------|------------------------------------------------------------------------------------------------------------------------|
| 16484 | <b>term</b>                                                                                                            |
| 16485 | Historical implementations of <b>ex</b> and <b>vi</b> ignored changes to the <b>term</b> edit option after the initial |
| 16486 | terminal information was loaded. This is permitted by IEEE Std 1003.1-2001; however,                                   |
| 16487 | implementations are encouraged to permit the user to modify their terminal type at any time.                           |
| 16488 | <b>terse</b>                                                                                                           |
| 16489 | Historically, the <b>terse</b> edit option optionally provided a shorter, less descriptive error message,              |
| 16490 | for some error messages. This is permitted, but not required, by IEEE Std 1003.1-2001.                                 |
| 16491 | Historically, most common visual mode errors (for example, trying to move the cursor past the                          |
| 16492 | end of a line) did not result in an error message, but simply alerted the terminal.                                    |
| 16493 | Implementations wishing to provide messages for novice users are urged to do so based on the                           |
| 16494 | <b>edit</b> option <b>verbose</b> , and not <b>terse</b> .                                                             |
| 16495 | <b>window</b>                                                                                                          |
| 16496 | In historical implementations, the default for the <b>window</b> edit option was based on the baud                     |
| 16497 | rate as follows:                                                                                                       |
| 16498 | 1. If the baud rate was less than 1 200, the <b>edit</b> option <b>w300</b> set the window value; for                  |
| 16499 | example, the line:                                                                                                     |
| 16500 | set w300=12                                                                                                            |
| 16501 | would set the window option to 12 if the baud rate was less than 1 200.                                                |
| 16502 | 2. If the baud rate was equal to 1 200, the <b>edit</b> option <b>w1200</b> set the window value.                      |
| 16503 | 3. If the baud rate was greater than 1 200, the <b>edit</b> option <b>w9600</b> set the window value.                  |
| 16504 | The <b>w300</b> , <b>w1200</b> , and <b>w9600</b> options do not appear in IEEE Std 1003.1-2001 because of their       |
| 16505 | dependence on specific baud rates.                                                                                     |
| 16506 | In historical implementations, the size of the window displayed by various commands was                                |
| 16507 | related to, but not necessarily the same as, the <b>window</b> edit option. For example, the size of the               |
| 16508 | window was set by the <b>ex</b> command <b>visual 10</b> , but it did not change the value of the <b>window</b>        |
| 16509 | edit option. However, changing the value of the <b>window</b> edit option did change the number of                     |
| 16510 | lines that were displayed when the screen was repainted. IEEE Std 1003.1-2001 does not permit                          |
| 16511 | this behavior in the interests of consistency and simplicity of specification, and requires that all                   |
| 16512 | commands that change the number of lines that are displayed do it by setting the value of the                          |
| 16513 | <b>window</b> edit option.                                                                                             |
| 16514 | <b>wrapmargin, wm</b>                                                                                                  |
| 16515 | Historically, the <b>wrapmargin</b> option did not affect maps inserting characters that also had                      |
| 16516 | associated <i>counts</i> ; for example :map K 5aABC DEF. Unfortunately, there are widely used                          |
| 16517 | maps that depend on this behavior. For consistency and simplicity of specification,                                    |
| 16518 | IEEE Std 1003.1-2001 does not permit this behavior.                                                                    |
| 16519 | Historically, <b>wrapmargin</b> was calculated using the column display width of all characters on the                 |
| 16520 | screen. For example, an implementation using " ^I " to represent <tab>s when the <b>list</b> edit                      |
| 16521 | option was set, where ' ^ ' and ' I ' each took up a single column on the screen, would calculate                      |
| 16522 | the <b>wrapmargin</b> based on a value of 2 for each <tab>. The <b>number</b> edit option similarly                    |
| 16523 | changed the effective length of the line as well. IEEE Std 1003.1-2001 requires conformance to                         |
| 16524 | historical practice.                                                                                                   |
| 16525 | Previous versions of this standard allowed for implementations with bytes other than eight bits,                       |
| 16526 | but this has been modified in this version.                                                                            |

**16527 FUTURE DIRECTIONS**

16528 None.

**16529 SEE ALSO**

16530 Section 2.9.1.1 (on page 48), *ctags*, *ed*, *sed*, *sh*, *stty*, *vi*, the System Interfaces volume of  
16531 IEEE Std 1003.1-2001, *access()*

**16532 CHANGE HISTORY**

16533 First released in Issue 2.

**16534 Issue 5**

16535 The FUTURE DIRECTIONS section is added.

**16536 Issue 6**

16537 This utility is marked as part of the User Portability Utilities option.

16538 The obsolescent SYNOPSIS is removed, removing the *+command* and *-* options.

16539 The following new requirements on POSIX implementations derive from alignment with the  
16540 Single UNIX Specification:

- In the **map** command description, the sequence *#digit* is added.

- The **directory**, **edcompatible**, **redraw**, and **slowopen** edit options are added.

16541 The **ex** utility is extensively changed for alignment with the IEEE P1003.2b draft standard. This  
16542 includes changes as a result of the IEEE PASC Interpretations 1003.2 #31, #38, #49, #50, #51, #52,  
16543 #55, #56, #57, #61, #62, #63, #64, #65, and #78.

16544 The **-l** option is removed.

16545 IEEE Std 1003.1-2001/Cor 1-2002, item XCU/TC1/D6/23 is applied, correcting a URL. 1

16546 IEEE Std 1003.1-2001/Cor 2-2004, item XCU/TC2/D6/8 is applied, making an editorial 2  
16547 correction in the EXTENDED DESCRIPTION. 2

16548 IEEE Std 1003.1-2001/Cor 2-2004, item XCU/TC2/D6/9 is applied, removing text describing 2  
16549 behavior on systems with bytes consisting of more than eight bits. 2

## 16552 NAME

16553 expand — convert tabs to spaces

## 16554 SYNOPSIS

16555 UP `expand [-t tablist][file ...]`

16556

## 16557 DESCRIPTION

16558 The *expand* utility shall write files or the standard input to the standard output with <tab>s replaced with one or more <space>s needed to pad to the next tab stop. Any <backspace>s 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.

## 16562 OPTIONS

16563 The *expand* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section 12.2, Utility Syntax Guidelines.

16565 The following option shall be supported:

16566 **-t tablist** Specify the tab stops. The application shall ensure that the argument *tablist* consists of either a single positive decimal integer or a list of tabstops. If a single number is given, tabs shall be set that number of column positions apart instead of the default 8.

16570 If a list of tabstops is given, the application shall ensure that it consists of a list of two or more positive decimal integers, separated by <blank>s or commas, in ascending order. The tabs shall be set at those specific column positions. Each tab stop *N* shall be an integer value greater than zero, and the list is in strictly ascending order. This is 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.

16577 In the event of *expand* having to process a <tab> 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.

## 16580 OPERANDS

16581 The following operand shall be supported:

16582 *file* The pathname of a text file to be used as input.

## 16583 STDIN

16584 See the INPUT FILES section.

## 16585 INPUT FILES

16586 Input files shall be text files.

## 16587 ENVIRONMENT VARIABLES

16588 The following environment variables shall affect the execution of *expand*:

16589 *LANG* Provide a default value for the internationalization variables that are unset or null. (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2, Internationalization Variables for the precedence of internationalization variables used to determine the values of locale categories.)

16593 *LC\_ALL* If set to a non-empty string value, override the values of all the other internationalization variables.

16595 *LC\_CTYPE* Determine the locale for the interpretation of sequences of bytes of text data as characters (for example, single-byte as opposed to multi-byte characters in

16597 arguments and input files), the processing of <tab>s and <space>s, and for the  
16598 determination of the width in column positions each character would occupy on  
16599 an output device.

16600 ***LC\_MESSAGES***

16601 Determine the locale that should be used to affect the format and contents of  
16602 diagnostic messages written to standard error.

16603 XSI ***NLSPATH*** Determine the location of message catalogs for the processing of ***LC\_MESSAGES***.

16604 **ASYNCHRONOUS EVENTS**

16605 Default.

16606 **STDOUT**

16607 The standard output shall be equivalent to the input files with <tab>s converted into the  
16608 appropriate number of <space>s.

16609 **STDERR**

16610 The standard error shall be used only for diagnostic messages.

16611 **OUTPUT FILES**

16612 None.

16613 **EXTENDED DESCRIPTION**

16614 None.

16615 **EXIT STATUS**

16616 The following exit values shall be returned:

16617 0 Successful completion

16618 >0 An error occurred.

16619 **CONSEQUENCES OF ERRORS**

16620 The *expand* utility shall terminate with an error message and non-zero exit status upon  
16621 encountering difficulties accessing one of the *file* operands.

16622 **APPLICATION USAGE**

16623 None.

16624 **EXAMPLES**

16625 None.

16626 **RATIONALE**

16627 The *expand* utility is useful for preprocessing text files (before sorting, looking at specific  
16628 columns, and so on) that contain <tab>s.

16629 See the Base Definitions volume of IEEE Std 1003.1-2001, Section 3.103, Column Position.

16630 The *tablist* option-argument consists of integers in ascending order. Utility Syntax Guideline 8  
16631 mandates that *expand* shall accept the integers (within the single argument) separated using  
16632 either commas or <blank>s.

16633 **FUTURE DIRECTIONS**

16634 None.

16635 **SEE ALSO**

16636 *tabs*, *unexpand*

**16637 CHANGE HISTORY**

16638 First released in Issue 4.

**16639 Issue 6**

16640 This utility is marked as part of the User Portability Utilities option.

16641 The APPLICATION USAGE section is added.

16642 The obsolescent SYNOPSIS is removed.

16643 The *LC\_CTYPE* environment variable description is updated to align with the IEEE P1003.2b draft standard.

16645 The normative text is reworded to avoid use of the term “must” for application requirements.

**16646 NAME**

16647        *expr* — evaluate arguments as an expression

**16648 SYNOPSIS**

16649        *expr* *operand*

**16650 DESCRIPTION**

16651        The *expr* utility shall evaluate an expression and write the result to standard output.

**16652 OPTIONS**

16653        None.

**16654 OPERANDS**

16655        The single expression evaluated by *expr* shall be formed from the operands, as described in the  
 16656        EXTENDED DESCRIPTION section. The application shall ensure that each of the expression  
 16657        operator symbols:

16658        ( ) | & = > >= < <= != + - \* / % :

16659        and the symbols *integer* and *string* in the table are provided as separate arguments to *expr*.

**16660 STDIN**

16661        Not used.

**16662 INPUT FILES**

16663        None.

**16664 ENVIRONMENT VARIABLES**

16665        The following environment variables shall affect the execution of *expr*:

16666        *LANG*        Provide a default value for the internationalization variables that are unset or null.  
 16667                  (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2,  
 16668                  Internationalization Variables for the precedence of internationalization variables  
 16669                  used to determine the values of locale categories.)

16670        *LC\_ALL*      If set to a non-empty string value, override the values of all the other  
 16671                  internationalization variables.

*LC\_COLLATE*

16673        Determine the locale for the behavior of ranges, equivalence classes, and multi-  
 16674                  character collating elements within regular expressions and by the string  
 16675                  comparison operators.

16676        *LC\_CTYPE*     Determine the locale for the interpretation of sequences of bytes of text data as  
 16677                  characters (for example, single-byte as opposed to multi-byte characters in  
 16678                  arguments) and the behavior of character classes within regular expressions.

*LC\_MESSAGES*

16680        Determine the locale that should be used to affect the format and contents of  
 16681                  diagnostic messages written to standard error.

16682 XSI      *NLSPATH*     Determine the location of message catalogs for the processing of *LC\_MESSAGES*.

**16683 ASYNCHRONOUS EVENTS**

16684        Default.

**16685 STDOUT**

16686        The *expr* utility shall evaluate the expression and write the result, followed by a <newline>, to  
 16687                  standard output.

16688 **STDERR**

16689 The standard error shall be used only for diagnostic messages.

16690 **OUTPUT FILES**

16691 None.

16692 **EXTENDED DESCRIPTION**

16693 The formation of the expression to be evaluated is shown in the following table. The symbols  
 16694 *expr*, *expr1*, and *expr2* represent expressions formed from *integer* and *string* symbols and the  
 16695 expression operator symbols (all separate arguments) by recursive application of the constructs  
 16696 described in the table. The expressions are listed in order of increasing precedence, with equal-  
 16697 precedence operators grouped between horizontal lines. All of the operators shall be left-  
 16698 associative.

| 16699 | <b>Expression</b>            | <b>Description</b>                                                                                                                                                                                                                                                                                 |
|-------|------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 16700 | <i>expr1</i>   <i>expr2</i>  | Returns the evaluation of <i>expr1</i> if it is neither null nor zero; otherwise, returns the evaluation of <i>expr2</i> if it is not null; otherwise, zero.                                                                                                                                       |
| 16701 | <i>expr1</i> & <i>expr2</i>  | Returns the evaluation of <i>expr1</i> if neither expression evaluates to null or zero; otherwise, returns zero.                                                                                                                                                                                   |
| 16702 |                              |                                                                                                                                                                                                                                                                                                    |
| 16703 |                              |                                                                                                                                                                                                                                                                                                    |
| 16704 |                              |                                                                                                                                                                                                                                                                                                    |
| 16705 |                              |                                                                                                                                                                                                                                                                                                    |
| 16706 |                              |                                                                                                                                                                                                                                                                                                    |
| 16707 |                              |                                                                                                                                                                                                                                                                                                    |
| 16708 |                              |                                                                                                                                                                                                                                                                                                    |
| 16709 |                              |                                                                                                                                                                                                                                                                                                    |
| 16710 |                              |                                                                                                                                                                                                                                                                                                    |
| 16711 | <i>expr1</i> = <i>expr2</i>  | Returns the result of a decimal integer comparison if both arguments are integers; otherwise, returns the result of a string comparison using the locale-specific collation sequence. The result of each comparison is 1 if the specified relationship is true, or 0 if the relationship is false. |
| 16712 | <i>expr1</i> > <i>expr2</i>  | Equal.                                                                                                                                                                                                                                                                                             |
| 16713 | <i>expr1</i> >= <i>expr2</i> | Greater than.                                                                                                                                                                                                                                                                                      |
| 16714 | <i>expr1</i> < <i>expr2</i>  | Greater than or equal.                                                                                                                                                                                                                                                                             |
| 16715 | <i>expr1</i> <= <i>expr2</i> | Less than.                                                                                                                                                                                                                                                                                         |
| 16716 | <i>expr1</i> != <i>expr2</i> | Less than or equal.                                                                                                                                                                                                                                                                                |
| 16717 |                              | Not equal.                                                                                                                                                                                                                                                                                         |
| 16718 | <i>expr1</i> + <i>expr2</i>  | Addition of decimal integer-valued arguments.                                                                                                                                                                                                                                                      |
| 16719 | <i>expr1</i> - <i>expr2</i>  | Subtraction of decimal integer-valued arguments.                                                                                                                                                                                                                                                   |
| 16720 |                              |                                                                                                                                                                                                                                                                                                    |
| 16721 | <i>expr1</i> * <i>expr2</i>  | Multiplication of decimal integer-valued arguments.                                                                                                                                                                                                                                                |
| 16722 | <i>expr1</i> / <i>expr2</i>  | Integer division of decimal integer-valued arguments, producing an integer result.                                                                                                                                                                                                                 |
| 16723 | <i>expr1</i> % <i>expr2</i>  | Remainder of integer division of decimal integer-valued arguments.                                                                                                                                                                                                                                 |
| 16724 | <i>expr1</i> : <i>expr2</i>  | Matching expression; see below.                                                                                                                                                                                                                                                                    |
| 16725 | ( <i>expr</i> )              | Grouping symbols. Any expression can be placed within parentheses. Parentheses can be nested to a depth of {EXPR_NEST_MAX}.                                                                                                                                                                        |
| 16726 |                              |                                                                                                                                                                                                                                                                                                    |
| 16727 | <i>integer</i>               | An argument consisting only of an (optional) unary minus followed by digits.                                                                                                                                                                                                                       |
| 16728 | <i>string</i>                | A string argument; see below.                                                                                                                                                                                                                                                                      |
| 16729 |                              |                                                                                                                                                                                                                                                                                                    |

16730

## Matching Expression

16731  
16732  
16733  
16734  
16735  
16736  
16737  
16738  
16739

The ': ' matching operator shall compare the string resulting from the evaluation of *expr1* with the regular expression pattern resulting from the evaluation of *expr2*. Regular expression syntax shall be that defined in the Base Definitions volume of IEEE Std 1003.1-2001, Section 9.3, Basic Regular Expressions, except that all patterns are anchored to the beginning of the string (that is, only sequences starting at the first character of a string are matched by the regular expression) and, therefore, it is unspecified whether '^' is a special character in that context. Usually, the matching operator shall return a string representing the number of characters matched ('0' on failure). Alternatively, if the pattern contains at least one regular expression subexpression "[ \(\ .\ ) ]", the string corresponding to "\1" shall be returned.

16740

## String Operand

16741  
16742

A string argument is an argument that cannot be identified as an *integer* argument or as one of the expression operator symbols shown in the OPERANDS section.

16743

The use of string arguments **length**, **substr**, **index**, or **match** produces unspecified results.

16744

## EXIT STATUS

16745

The following exit values shall be returned:

16746  
16747  
16748  
16749

- 0 The *expression* evaluates to neither null nor zero.
- 1 The *expression* evaluates to null or zero.
- 2 Invalid *expression*.
- >2 An error occurred.

16750

## CONSEQUENCES OF ERRORS

16751

Default.

16752

## APPLICATION USAGE

16753  
16754

After argument processing by the shell, *expr* is not required to be able to tell the difference between an operator and an operand except by the value. If "\$a" is '=' , the command:

16755

```
expr $a = '='
```

16756

looks like:

16757

```
expr = = =
```

16758  
16759

as the arguments are passed to *expr* (and they all may be taken as the '=' operator). The following works reliably:

16760

```
expr X$a = X=
```

16761  
16762  
16763  
16764  
16765  
16766

Also note that this volume of IEEE Std 1003.1-2001 permits implementations to extend utilities. The *expr* utility permits the integer arguments to be preceded with a unary minus. This means that an integer argument could look like an option. Therefore, the conforming application must employ the "--" construct of Guideline 10 of the Base Definitions volume of IEEE Std 1003.1-2001, Section 12.2, Utility Syntax Guidelines to protect its operands if there is any chance the first operand might be a negative integer (or any string with a leading minus).

16767

## EXAMPLES

16768

The *expr* utility has a rather difficult syntax:

16769  
16770

- Many of the operators are also shell control operators or reserved words, so they have to be escaped on the command line.

- 16771 • Each part of the expression is composed of separate arguments, so liberal usage of <blank>s  
 16772 is required. For example:

| Invalid                       | Valid                                    |
|-------------------------------|------------------------------------------|
| <code>expr 1+2</code>         | <code>expr 1 + 2</code>                  |
| <code>expr "1 + 2"</code>     | <code>expr 1 + 2</code>                  |
| <code>expr 1 + (2 * 3)</code> | <code>expr 1 + \(\ 2 \ * \ 3 \ \)</code> |

16778 In many cases, the arithmetic and string features provided as part of the shell command  
 16779 language are easier to use than their equivalents in *expr*. Newly written scripts should avoid  
 16780 *expr* in favor of the new features within the shell; see Section 2.5 (on page 33) and Section 2.6.4  
 16781 (on page 41).

16782 The following command:

16783 `a=$(expr $a + 1)`

16784 adds 1 to the variable *a*.

16785 The following command, for "\$a" equal to either /usr/abc/file or just file:

16786 `expr $a : '.* /\(.*\)' \| $a`

16787 returns the last segment of a pathname (that is, *file*). Applications should avoid the character  
 16788 '/' used alone as an argument; *expr* may interpret it as the division operator.

16789 The following command:

16790 `expr "//$a" : '.* /\(.*\)'`

16791 is a better representation of the previous example. The addition of the "://" characters  
 16792 eliminates any ambiguity about the division operator and simplifies the whole expression. Also  
 16793 note that pathnames may contain characters contained in the *IFS* variable and should be quoted  
 16794 to avoid having "\$a" expand into multiple arguments.

16795 The following command:

16796 `expr "$VAR" : '.*'`

16797 returns the number of characters in *VAR*.

## 16798 RATIONALE

In an early proposal, EREs were used in the matching expression syntax. This was changed to BREs to avoid breaking historical applications.

The use of a leading circumflex in the BRE is unspecified because many historical implementations have treated it as a special character, despite their system documentation. For example:

`expr foo : ^foo expr ^foo : ^foo`

return 3 and 0, respectively, on those systems; their documentation would imply the reverse. Thus, the anchoring condition is left unspecified to avoid breaking historical scripts relying on this undocumented feature.

## 16808 FUTURE DIRECTIONS

16809 None.

**16810 SEE ALSO**

16811       Section 2.5 (on page 33), Section 2.6.4 (on page 41)

**16812 CHANGE HISTORY**

16813       First released in Issue 2.

**16814 Issue 5**

16815       The FUTURE DIRECTIONS section is added.

**16816 Issue 6**

16817       The *expr* utility is aligned with the IEEE P1003.2b draft standard, to include resolution of IEEE  
16818       PASC Interpretation 1003.2 #104.

16819       The normative text is reworded to avoid use of the term “must” for application requirements.

**16820 NAME**

16821       **false** — return false value

**16822 SYNOPSIS**

16823       **false**

**16824 DESCRIPTION**

16825       The *false* utility shall return with a non-zero exit code.

**16826 OPTIONS**

16827       None.

**16828 OPERANDS**

16829       None.

**16830 STDIN**

16831       Not used.

**16832 INPUT FILES**

16833       None.

**16834 ENVIRONMENT VARIABLES**

16835       None.

**16836 ASYNCHRONOUS EVENTS**

16837       Default.

**16838 STDOUT**

16839       Not used.

1

**16840 STDERR**

16841       Not used.

**16842 OUTPUT FILES**

16843       None.

**16844 EXTENDED DESCRIPTION**

16845       None.

**16846 EXIT STATUS**

16847       The *false* utility shall always exit with a value other than zero.

**16848 CONSEQUENCES OF ERRORS**

16849       Default.

**16850 APPLICATION USAGE**

16851       None.

**16852 EXAMPLES**

16853       None.

**16854 RATIONALE**

16855       None.

**16856 FUTURE DIRECTIONS**

16857       None.

**16858 SEE ALSO**

16859       *true*

**16860 CHANGE HISTORY**

16861 First released in Issue 2.

**16862 Issue 6**

16863 IEEE Std 1003.1-2001/Cor 1-2002, item XCU/TC1/D6/24 is applied, changing the STDERR 1  
16864 section from “None.” to “Not used.” for alignment with Section 1.11 (on page 20). 1

## 16865 NAME

16866 fc — process the command history list

## 16867 SYNOPSIS

16868 UP fc [-r][-e editor] [first[last]]

16869 fc -l[-nr] [first[last]]

16870 fc -s[old=new][first]

16871

## 16872 DESCRIPTION

16873 The *fc* utility shall list, or shall edit and re-execute, commands previously entered to an  
16874 interactive *sh*.

16875 The command history list shall reference commands by number. The first number in the list is  
16876 selected arbitrarily. The relationship of a number to its command shall not change except when  
16877 the user logs in and no other process is accessing the list, at which time the system may reset the  
16878 numbering to start the oldest retained command at another number (usually 1). When the  
16879 number reaches an implementation-defined upper limit, which shall be no smaller than the  
16880 value in *HISTSIZE* or 32 767 (whichever is greater), the shell may wrap the numbers, starting the  
16881 next command with a lower number (usually 1). However, despite this optional wrapping of  
16882 numbers, *fc* shall maintain the time-ordering sequence of the commands. For example, if four  
16883 commands in sequence are given the numbers 32 766, 32 767, 1 (wrapped), and 2 as they are  
16884 executed, command 32 767 is considered the command previous to 1, even though its number is  
16885 higher.

16886 When commands are edited (when the **-l** option is not specified), the resulting lines shall be  
16887 entered at the end of the history list and then re-executed by *sh*. The *fc* command that caused the  
16888 editing shall not be entered into the history list. If the editor returns a non-zero exit status, this  
16889 shall suppress the entry into the history list and the command re-execution. Any command line  
16890 variable assignments or redirection operators used with *fc* shall affect both the *fc* command itself  
16891 as well as the command that results; for example:

16892 fc -s -- -l 2&gt;/dev/null

16893 reinvoques the previous command, suppressing standard error for both *fc* and the previous  
16894 command.

## 16895 OPTIONS

16896 The *fc* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section 12.2,  
16897 Utility Syntax Guidelines.

16898 The following options shall be supported:

16899 **-e editor** Use the editor named by *editor* to edit the commands. The *editor* string is a utility  
16900 name, subject to search via the *PATH* variable (see the Base Definitions volume of  
16901 IEEE Std 1003.1-2001, Chapter 8, Environment Variables). The value in the *FCEDIT*  
16902 variable shall be used as a default when **-e** is not specified. If *FCEDIT* is null or  
16903 unset, *ed* shall be used as the editor.

16904 **-l** (The letter ell.) List the commands rather than invoking an editor on them. The  
16905 commands shall be written in the sequence indicated by the *first* and *last* operands,  
16906 as affected by **-r**, with each command preceded by the command number.

16907 **-n** Suppress command numbers when listing with **-l**.

16908 **-r** Reverse the order of the commands listed (with **-l**) or edited (with neither **-l** nor  
16909 **-s**).

16910        **-s**        Re-execute the command without invoking an editor.

16911 **OPERANDS**

16912        The following operands shall be supported:

16913        *first, last*        Select the commands to list or edit. The number of previous commands that can be  
 16914        accessed shall be determined by the value of the *HISTSIZE* variable. The value of  
 16915        *first* or *last* or both shall be one of the following:

16916        **[+]number**        A positive number representing a command number; command  
 16917        numbers can be displayed with the **-l** option.

16918        **-number**        A negative decimal number representing the command that was  
 16919        executed *number* of commands previously. For example, **-1** is the  
 16920        immediately previous command.

16921        **string**        A string indicating the most recently entered command that begins  
 16922        with that string. If the *old=new* operand is not also specified with **-s**,  
 16923        the string form of the *first* operand cannot contain an embedded  
 16924        equal sign.

16925        When the synopsis form with **-s** is used:

- If *first* is omitted, the previous command shall be used.

16927        For the synopsis forms without **-s**:

- If *last* is omitted, *last* shall default to the previous command when **-l** is specified; otherwise, it shall default to *first*.

- If *first* and *last* are both omitted, the previous 16 commands shall be listed or the previous single command shall be edited (based on the **-l** option).

- If *first* and *last* are both present, all of the commands from *first* to *last* shall be edited (without **-l**) or listed (with **-l**). Editing multiple commands shall be accomplished by presenting to the editor all of the commands at one time, each command starting on a new line. If *first* represents a newer command than *last*, the commands shall be listed or edited in reverse sequence, equivalent to using **-r**. For example, the following commands on the first line are equivalent to the corresponding commands on the second:

```
16939 fc -r 10 20 fc 30 40
16940 fc 20 10 fc -r 40 30
```

- When a range of commands is used, it shall not be an error to specify *first* or *last* values that are not in the history list; *fc* shall substitute the value representing the oldest or newest command in the list, as appropriate. For example, if there are only ten commands in the history list, numbered 1 to 10:

```
16945 fc -l
16946 fc 1 99
```

16947        shall list and edit, respectively, all ten commands.

16948        **old=new**        Replace the first occurrence of string *old* in the commands to be re-executed by the  
 16949        string *new*.

16950 **STDIN**

16951 Not used.

16952 **INPUT FILES**

16953 None.

16954 **ENVIRONMENT VARIABLES**16955 The following environment variables shall affect the execution of *fc*:16956 ***FCEDIT*** This variable, when expanded by the shell, shall determine the default value for  
16957 the **-e editor** option's *editor* option-argument. If *FCEDIT* is null or unset, *ed* shall be  
16958 used as the editor.16959 ***HISTFILE*** Determine a pathname naming a command history file. If the *HISTFILE* variable is  
16960 not set, the shell may attempt to access or create a file **.sh\_history** in the directory  
16961 referred to by the *HOME* environment variable. If the shell cannot obtain both read  
16962 and write access to, or create, the history file, it shall use an unspecified  
16963 mechanism that allows the history to operate properly. (References to history  
16964 "file" in this section shall be understood to mean this unspecified mechanism in  
16965 such cases.) An implementation may choose to access this variable only when  
16966 initializing the history file; this initialization shall occur when *fc* or *sh* first attempt  
16967 to retrieve entries from, or add entries to, the file, as the result of commands issued  
16968 by the user, the file named by the *ENV* variable, or implementation-defined system  
16969 start-up files. In some historical shells, the history file is initialized just after the  
16970 *ENV* file has been processed. Therefore, it is implementation-defined whether  
16971 changes made to *HISTFILE* after the history file has been initialized are effective.  
16972 Implementations may choose to disable the history list mechanism for users with  
16973 appropriate privileges who do not set *HISTFILE*; the specific circumstances under  
16974 which this occurs are implementation-defined. If more than one instance of the  
16975 shell is using the same history file, it is unspecified how updates to the history file  
16976 from those shells interact. As entries are deleted from the history file, they shall be  
16977 deleted oldest first. It is unspecified when history file entries are physically  
16978 removed from the history file.16979 ***HISTSIZE*** Determine a decimal number representing the limit to the number of previous  
16980 commands that are accessible. If this variable is unset, an unspecified default  
16981 greater than or equal to 128 shall be used. The maximum number of commands in  
16982 the history list is unspecified, but shall be at least 128. An implementation may  
16983 choose to access this variable only when initializing the history file, as described  
16984 under *HISTFILE*. Therefore, it is unspecified whether changes made to *HISTSIZE*  
16985 after the history file has been initialized are effective.16986 ***LANG*** Provide a default value for the internationalization variables that are unset or null.  
16987 (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2,  
16988 Internationalization Variables for the precedence of internationalization variables  
16989 used to determine the values of locale categories.)16990 ***LC\_ALL*** If set to a non-empty string value, override the values of all the other  
16991 internationalization variables.16992 ***LC\_CTYPE*** Determine the locale for the interpretation of sequences of bytes of text data as  
16993 characters (for example, single-byte as opposed to multi-byte characters in  
16994 arguments and input files).16995 ***LC\_MESSAGES*** Determine the locale that should be used to affect the format and contents of  
16996 diagnostic messages written to standard error.  
16997

16998 XSI **NLSPATH** Determine the location of message catalogs for the processing of *LC\_MESSAGES*.

## 16999 ASYNCHRONOUS EVENTS

17000 Default.

## 17001 STDOUT

17002 When the **-l** option is used to list commands, the format of each command in the list shall be as follows:

17004 "%d\t%s\n", <line number>, <command>

17005 If both the **-l** and **-n** options are specified, the format of each command shall be:

17006 "\t%s\n", <command>

17007 If the <command> consists of more than one line, the lines after the first shall be displayed as:

17008 "\t%s\n", <continued-command>

## 17009 STDERR

17010 The standard error shall be used only for diagnostic messages.

## 17011 OUTPUT FILES

17012 None.

## 17013 EXTENDED DESCRIPTION

17014 None.

## 17015 EXIT STATUS

17016 The following exit values shall be returned:

17017 0 Successful completion of the listing.

17018 >0 An error occurred.

17019 Otherwise, the exit status shall be that of the commands executed by *fc*.

## 17020 CONSEQUENCES OF ERRORS

17021 Default.

## 17022 APPLICATION USAGE

17023 Since editors sometimes use file descriptors as integral parts of their editing, redirecting their file descriptors as part of the *fc* command can produce unexpected results. For example, if *vi* is the *FCEDIT* editor, the command:

17026 fc -s | more

17027 does not work correctly on many systems.

17028 Users on windowing systems may want to have separate history files for each window by setting *HISTFILE* as follows:

17030 HISTFILE=\$HOME/.sh\_hist\$\$

## 17031 EXAMPLES

17032 None.

## 17033 RATIONALE

17034 This utility is based on the *fc* built-in of the KornShell.

17035 An early proposal specified the **-e** option as **[**-e** editor [*old= new*]]**, which is not historical practice. Historical practice in *fc* of either **[**-e** editor]** or **[**-e -** [*old= new*]]** is acceptable, but not both together. To clarify this, a new option **-s** was introduced replacing the **[**-e -**]**. This resolves the conflict and makes *fc* conform to the Utility Syntax Guidelines.

|       |                            |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
|-------|----------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 17039 | <b>HISTFILE</b>            | Some implementations of the KornShell check for the superuser and do not create a history file unless <i>HISTFILE</i> is set. This is done primarily to avoid creating unlinked files in the root file system when logging in during single-user mode. <i>HISTFILE</i> must be set for the superuser to have history.                                                                                                                                                                                                                                                                                                                                                                                                                        |
| 17040 |                            |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| 17041 |                            |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| 17042 |                            |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| 17043 | <b>HISTSIZE</b>            | Needed to limit the size of history files. It is the intent of the standard developers that when two shells share the same history file, commands that are entered in one shell shall be accessible by the other shell. Because of the difficulties of synchronization over a network, the exact nature of the interaction is unspecified.                                                                                                                                                                                                                                                                                                                                                                                                   |
| 17044 |                            |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| 17045 |                            |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| 17046 |                            |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| 17047 |                            | The initialization process for the history file can be dependent on the system start-up files, in that they may contain commands that effectively preempt the settings the user has for <i>HISTFILE</i> and <i>HISTSIZE</i> . For example, function definition commands are recorded in the history file. If the system administrator includes function definitions in some system start-up file called before the <i>ENV</i> file, the history file is initialized before the user can influence its characteristics. In some historical shells, the history file is initialized just after the <i>ENV</i> file has been processed. Because of these situations, the text requires the initialization process to be implementation-defined. |
| 17048 |                            |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| 17049 |                            |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| 17050 |                            |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| 17051 |                            |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| 17052 |                            |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| 17053 |                            |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| 17054 |                            | Consideration was given to omitting the <i>fc</i> utility in favor of the command line editing feature in <i>sh</i> . For example, in <i>vi</i> editing mode, typing "<ESC> v" is equivalent to:                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| 17055 |                            |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| 17056 | EDITOR=vi fc               |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| 17057 |                            | However, the <i>fc</i> utility allows the user the flexibility to edit multiple commands simultaneously (such as <i>fc</i> 10 20) and to use editors other than those supported by <i>sh</i> for command line editing.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| 17058 |                            |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| 17059 |                            | In the KornShell, the alias <i>r</i> ('re-do') is preset to <i>fc -e</i> – (equivalent to the POSIX <i>fc -s</i> ). This is probably an easier command name to remember than <i>fc</i> ("fix command"), but it does not meet the Utility Syntax Guidelines. Renaming <i>fc</i> to <i>hist</i> or <i>redo</i> was considered, but since this description closely matches historical KornShell practice already, such a renaming was seen as gratuitous. Users are free to create aliases whenever odd historical names such as <i>fc</i> , <i>awk</i> , <i>cat</i> , <i>grep</i> , or <i>yacc</i> are standardized by POSIX.                                                                                                                  |
| 17060 |                            |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| 17061 |                            |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| 17062 |                            |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| 17063 |                            |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| 17064 |                            |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| 17065 |                            | Command numbers have no ordering effects; they are like serial numbers. The <i>-r</i> option and <i>-number</i> operand address the sequence of command execution, regardless of serial numbers. So, for example, if the command number wrapped back to 1 at some arbitrary point, there would be no ambiguity associated with traversing the wrap point. For example, if the command history were:                                                                                                                                                                                                                                                                                                                                          |
| 17066 |                            |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| 17067 |                            |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| 17068 |                            |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| 17069 |                            |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| 17070 | 32766: echo 1              |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| 17071 | 32767: echo 2              |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| 17072 | 1: echo 3                  |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| 17073 |                            | the number -2 refers to command 32767 because it is the second previous command, regardless of serial number.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| 17074 |                            |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| 17075 | <b>FUTURE DIRECTIONS</b>   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| 17076 | None.                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| 17077 | <b>SEE ALSO</b>            |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| 17078 | <i>sh</i>                  |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| 17079 | <b>CHANGE HISTORY</b>      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| 17080 | First released in Issue 4. |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |

**17081 Issue 5**

17082 The FUTURE DIRECTIONS section is added.

**17083 Issue 6**

17084 This utility is marked as part of the User Portability Utilities option.

17085 In the ENVIRONMENT VARIABLES section, the text “user’s home directory” is updated to  
17086 “directory referred to by the *HOME* environment variable”.

**17087 NAME**

17088        *fg* — run jobs in the foreground

**17089 SYNOPSIS**

17090 UP        **fg** [*job\_id*]

17091

**17092 DESCRIPTION**

17093        If job control is enabled (see the description of *set -m*), the *fg* utility shall move a background job  
17094        from the current environment (see Section 2.12 (on page 61)) into the foreground.

17095        Using *fg* to place a job into the foreground shall remove its process ID from the list of those  
17096        “known in the current shell execution environment”; see Section 2.9.3.1 (on page 50).

**17097 OPTIONS**

17098        None.

**17099 OPERANDS**

17100        The following operand shall be supported:

17101        *job\_id*        Specify the job to be run as a foreground job. If no *job\_id* operand is given, the  
17102        *job\_id* for the job that was most recently suspended, placed in the background, or  
17103        run as a background job shall be used. The format of *job\_id* is described in the Base  
17104        Definitions volume of IEEE Std 1003.1-2001, Section 3.203, Job Control Job ID.

**17105 STDIN**

17106        Not used.

**17107 INPUT FILES**

17108        None.

**17109 ENVIRONMENT VARIABLES**

17110        The following environment variables shall affect the execution of *fg*:

17111        *LANG*        Provide a default value for the internationalization variables that are unset or null.  
17112        (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2,  
17113        Internationalization Variables for the precedence of internationalization variables  
17114        used to determine the values of locale categories.)

17115        *LC\_ALL*      If set to a non-empty string value, override the values of all the other  
17116        internationalization variables.

17117        *LC\_CTYPE*     Determine the locale for the interpretation of sequences of bytes of text data as  
17118        characters (for example, single-byte as opposed to multi-byte characters in  
17119        arguments).

**17120 *LC\_MESSAGES***

17121        Determine the locale that should be used to affect the format and contents of  
17122        diagnostic messages written to standard error.

17123 XSI      *NLSPATH*     Determine the location of message catalogs for the processing of *LC\_MESSAGES*.

**17124 ASYNCHRONOUS EVENTS**

17125        Default.

**17126 STDOUT**

17127        The *fg* utility shall write the command line of the job to standard output in the following format:

17128        "%s\n", <command>

**17129 STDERR**

17130 The standard error shall be used only for diagnostic messages.

**17131 OUTPUT FILES**

17132 None.

**17133 EXTENDED DESCRIPTION**

17134 None.

**17135 EXIT STATUS**

17136 The following exit values shall be returned:

17137 0 Successful completion.

17138 >0 An error occurred.

**17139 CONSEQUENCES OF ERRORS**

17140 If job control is disabled, the *fg* utility shall exit with an error and no job shall be placed in the  
17141 foreground.

**17142 APPLICATION USAGE**

17143 The *fg* utility does not work as expected when it is operating in its own utility execution  
17144 environment because that environment has no applicable jobs to manipulate. See the  
17145 APPLICATION USAGE section for *bg*. For this reason, *fg* is generally implemented as a shell  
17146 regular built-in.

**17147 EXAMPLES**

17148 None.

**17149 RATIONALE**

17150 The extensions to the shell specified in this volume of IEEE Std 1003.1-2001 have mostly been  
17151 based on features provided by the KornShell. The job control features provided by *bg*, *fg*, and *jobs*  
17152 are also based on the KornShell. The standard developers examined the characteristics of the C  
17153 shell versions of these utilities and found that differences exist. Despite widespread use of the C  
17154 shell, the KornShell versions were selected for this volume of IEEE Std 1003.1-2001 to maintain a  
17155 degree of uniformity with the rest of the KornShell features selected (such as the very popular  
17156 command line editing features).

**17157 FUTURE DIRECTIONS**

17158 None.

**17159 SEE ALSO**

17160 Section 2.9.3.1 (on page 50), Section 2.12 (on page 61), *bg*, *kill*, *jobs*, *wait*

**17161 CHANGE HISTORY**

17162 First released in Issue 4.

**17163 Issue 6**

17164 This utility is marked as part of the User Portability Utilities option.

17165 The APPLICATION USAGE section is added.

17166 The JC marking is removed from the SYNOPSIS since job control is mandatory in this issue.

## 17167 NAME

17168 file — determine file type

## 17169 SYNOPSIS

17170 UP file [-dh][-M file][-m file] file ...

1

17171 file -i [-h] file ...

1

17172

1

## 17173 DESCRIPTION

17174 The *file* utility shall perform a series of tests in sequence on each specified *file* in an attempt to 1  
17175 classify it:17176 1. If *file* does not exist, cannot be read, or its file status could not be determined, the output 1  
17177 shall indicate that the file was processed, but that its type could not be determined.17178 2. If the file is not a regular file, its file type shall be identified. The file types directory, FIFO, 1  
17179 socket, block special, and character special shall be identified as such. Other 1  
17180 implementation-defined file types may also be identified. If *file* is a symbolic link, by 1  
17181 default the link shall be resolved and *file* shall test the type of file referenced by the 1  
17182 symbolic link. (See the **-h** and **-i** options below.) 117183 3. If the length of *file* is zero, it shall be identified as an empty file. 117184 4. The *file* utility shall examine an initial segment of *file* and shall make a guess at identifying 1  
17185 its contents based on position-sensitive tests. (The answer is not guaranteed to be correct; 1  
17186 see the **-d**, **-M**, and **-m** options below.) 117187 5. The *file* utility shall examine *file* and make a guess at identifying its contents based on 1  
17188 context-sensitive default system tests. (The answer is not guaranteed to be correct.) 1

17189 6. The file shall be identified as a data file. 1

17190 If *file* does not exist, cannot be read, or its file status could not be determined, the output shall 1  
17191 indicate that the file was processed, but that its type could not be determined.17192 If *file* is a symbolic link, by default the link shall be resolved and *file* shall test the type of file 1  
17193 referenced by the symbolic link.

## 17194 OPTIONS

17195 The *file* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section 12.2, 1  
17196 Utility Syntax Guidelines, except that the order of the **-m**, **-d**, and **-M** options shall be 1  
17197 significant. 1

17198 The following options shall be supported by the implementation:

17199 **-d** Apply any position-sensitive default system tests and context-sensitive default 1  
17200 system tests to the file. This is the default if no **-M** or **-m** option is specified. 117201 **-h** When a symbolic link is encountered, identify the file as a symbolic link. If **-h** is 1  
17202 not specified and *file* is a symbolic link that refers to a nonexistent file, *file* shall 1  
17203 identify the file as a symbolic link, as if **-h** had been specified.17204 **-i** If a file is a regular file, do not attempt to classify the type of the file further, but 1  
17205 identify the file as specified in the STDOUT section. 117206 **-M** *file* Specify the name of a file containing position-sensitive tests that shall be applied to 1  
17207 a file in order to classify it (see the EXTENDED DESCRIPTION). No position- 1  
17208 sensitive default system tests nor context-sensitive default system tests shall be 1  
17209 applied unless the **-d** option is also specified. 1

|       |                              |                                                                                                                                                                                                                                                                                                          |
|-------|------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 17210 | <b>-m file</b>               | Specify the name of a file containing position-sensitive tests that shall be applied to a file in order to classify it (see the EXTENDED DESCRIPTION).                                                                                                                                                   |
| 17211 |                              |                                                                                                                                                                                                                                                                                                          |
| 17212 |                              | If the <b>-m</b> option is specified without specifying the <b>-d</b> option or the <b>-M</b> option, position-                                                                                                                                                                                          |
| 17213 |                              | sensitive default system tests shall be applied after the position-sensitive tests specified by the                                                                                                                                                                                                      |
| 17214 |                              | <b>-m</b> option. If the <b>-M</b> option is specified with the <b>-d</b> option, the <b>-m</b> option, or both, or the <b>-m</b>                                                                                                                                                                        |
| 17215 |                              | option is specified with the <b>-d</b> option, the concatenation of the position-sensitive tests specified                                                                                                                                                                                               |
| 17216 |                              | by these options shall be applied in the order specified by the appearance of these options. If a                                                                                                                                                                                                        |
| 17217 |                              | <b>-M</b> or <b>-m file</b> option-argument is <b>-</b> , the results are unspecified.                                                                                                                                                                                                                   |
| 17218 | <b>OPERANDS</b>              |                                                                                                                                                                                                                                                                                                          |
| 17219 |                              | The following operand shall be supported:                                                                                                                                                                                                                                                                |
| 17220 | <b>file</b>                  | A pathname of a file to be tested.                                                                                                                                                                                                                                                                       |
| 17221 | <b>STDIN</b>                 |                                                                                                                                                                                                                                                                                                          |
| 17222 |                              | Not used.                                                                                                                                                                                                                                                                                                |
| 17223 | <b>INPUT FILES</b>           |                                                                                                                                                                                                                                                                                                          |
| 17224 |                              | The <i>file</i> can be any file type.                                                                                                                                                                                                                                                                    |
| 17225 | <b>ENVIRONMENT VARIABLES</b> |                                                                                                                                                                                                                                                                                                          |
| 17226 |                              | The following environment variables shall affect the execution of <i>file</i> :                                                                                                                                                                                                                          |
| 17227 | <b>LANG</b>                  | Provide a default value for the internationalization variables that are unset or null.<br>(See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2, Internationalization Variables for the precedence of internationalization variables used to determine the values of locale categories.) |
| 17228 |                              |                                                                                                                                                                                                                                                                                                          |
| 17229 |                              |                                                                                                                                                                                                                                                                                                          |
| 17230 |                              |                                                                                                                                                                                                                                                                                                          |
| 17231 | <b>LC_ALL</b>                | If set to a non-empty string value, override the values of all the other internationalization variables.                                                                                                                                                                                                 |
| 17232 |                              |                                                                                                                                                                                                                                                                                                          |
| 17233 | <b>LC_CTYPE</b>              | Determine the locale for the interpretation of sequences of bytes of text data as characters (for example, single-byte as opposed to multi-byte characters in arguments and input files).                                                                                                                |
| 17234 |                              |                                                                                                                                                                                                                                                                                                          |
| 17235 |                              |                                                                                                                                                                                                                                                                                                          |
| 17236 | <b>LC_MESSAGES</b>           |                                                                                                                                                                                                                                                                                                          |
| 17237 |                              | Determine the locale that should be used to affect the format and contents of diagnostic messages written to standard error and informative messages written to standard output.                                                                                                                         |
| 17238 |                              |                                                                                                                                                                                                                                                                                                          |
| 17239 |                              |                                                                                                                                                                                                                                                                                                          |
| 17240 | <b>XSI</b>                   | <b>NLSPATH</b> Determine the location of message catalogs for the processing of <b>LC_MESSAGES</b> .                                                                                                                                                                                                     |
| 17241 | <b>ASYNCHRONOUS EVENTS</b>   |                                                                                                                                                                                                                                                                                                          |
| 17242 |                              | Default.                                                                                                                                                                                                                                                                                                 |
| 17243 | <b>STDOUT</b>                |                                                                                                                                                                                                                                                                                                          |
| 17244 |                              | In the POSIX locale, the following format shall be used to identify each operand, <i>file</i> specified:                                                                                                                                                                                                 |
| 17245 |                              | "%s: %s\n", <file>, <type>                                                                                                                                                                                                                                                                               |
| 17246 |                              | The values for <type> are unspecified, except that in the POSIX locale, if <i>file</i> is identified as one                                                                                                                                                                                              |
| 17247 |                              | of the types listed in the following table, <type> shall contain (but is not limited to) the                                                                                                                                                                                                             |
| 17248 |                              | corresponding string, unless the file is identified by a position-sensitive test specified by a <b>-M</b> or                                                                                                                                                                                             |
| 17249 |                              | <b>-m</b> option. Each space shown in the strings shall be exactly one <space>.                                                                                                                                                                                                                          |

**Table 4-8** File Utility Output Strings

| <b>If file is:</b>                            | <b>&lt;type&gt; shall contain the string:</b> | <b>Notes</b> |
|-----------------------------------------------|-----------------------------------------------|--------------|
| Nonexistent                                   | cannot open                                   | 1            |
| Block special                                 | block special                                 | 1            |
| Character special                             | character special                             | 1            |
| Directory                                     | directory                                     | 1            |
| FIFO                                          | fifo                                          | 1            |
| Socket                                        | socket                                        | 1            |
| Symbolic link                                 | symbolic link to                              | 1            |
| Regular file                                  | regular file                                  | 1,2          |
| Empty regular file                            | empty                                         | 3            |
| Regular file that cannot be read              | cannot open                                   | 3            |
| Executable binary                             | executable                                    | 4,6          |
| ar archive library (see ar)                   | archive                                       | 4,6          |
| Extended cpio format (see pax)                | cpio archive                                  | 4,6          |
| Extended tar format (see <b>ustar</b> in pax) | tar archive                                   | 4,6          |
| Shell script                                  | commands text                                 | 5,6          |
| C-language source                             | c program text                                | 5,6          |
| FORTRAN source                                | fortran program text                          | 5,6          |
| Regular file whose type cannot be determined  | data                                          | 1            |

**Notes:**

1. This is a file type test.
2. This test is applied only if the **-i** option is specified.
3. This test is applied only if the **-i** option is not specified.
4. This is a position-sensitive default system test.
5. This is a context-sensitive default system test.
6. Position-sensitive default system tests and context-sensitive default system tests are not applied if the **-M** option is specified unless the **-d** option is also specified.

In the POSIX locale, if *file* is identified as a symbolic link (see the **-h** option), the following alternative output format shall be used:

`"%s: %s %s\n", <file>, <type>, <contents of link>"`

If the file named by the *file* operand does not exist, cannot be read, or the type of the file named by the *file* operand cannot be determined, this shall not be considered an error that affects the exit status.

**STDERR**

The standard error shall be used only for diagnostic messages.

**OUTPUT FILES**

None.

**EXTENDED DESCRIPTION**

A file specified as an option-argument to the **-m** or **-M** options shall contain one position-sensitive test per line, which shall be applied to the file. If the test succeeds, the message field of the line shall be printed and no further tests shall be applied, with the exception that tests on immediately following lines beginning with a single '**'**' character shall be applied.

|       |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |   |
|-------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---|
| 17294 | Each line shall be composed of the following four <tab>-separated fields. (Implementations may allow any combination of one or more white space characters other than <newline> to act as field separators.)                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                | 2 |
| 17295 |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             | 2 |
| 17296 |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             | 2 |
| 17297 | <i>offset</i> An unsigned number (optionally preceded by a single '>' character) specifying the <i>offset</i> , in bytes, of the value in the file that is to be compared against the <i>value</i> field of the line. If the file is shorter than the specified offset, the test shall fail.                                                                                                                                                                                                                                                                                                                                                                                                                |   |
| 17298 |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |   |
| 17299 |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |   |
| 17300 | If the <i>offset</i> begins with the character '>', the test contained in the line shall not be applied to the file unless the test on the last line for which the <i>offset</i> did not begin with a '>' was successful. By default, the <i>offset</i> shall be interpreted as an unsigned decimal number. With a leading 0x or 0X, the <i>offset</i> shall be interpreted as a hexadecimal number; otherwise, with a leading 0, the <i>offset</i> shall be interpreted as an octal number.                                                                                                                                                                                                                |   |
| 17301 |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |   |
| 17302 |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |   |
| 17303 |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |   |
| 17304 |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |   |
| 17305 |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |   |
| 17306 | <i>type</i> The type of the value in the file to be tested. The type shall consist of the type specification characters d, s, and u, specifying signed decimal, string, and unsigned decimal, respectively.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |   |
| 17307 |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |   |
| 17308 |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             | 2 |
| 17309 | The <i>type</i> string shall be interpreted as the bytes from the file starting at the specified <i>offset</i> and including the same number of bytes specified by the <i>value</i> field.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |   |
| 17310 |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |   |
| 17311 | If insufficient bytes remain in the file past the <i>offset</i> to match the <i>value</i> field, the test shall fail.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |   |
| 17312 |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |   |
| 17313 | The type specification characters d and u can be followed by an optional unsigned decimal integer that specifies the number of bytes represented by the type. The type specification characters d and u can be followed by an optional C, S, I, or L, indicating that the value is of type <b>char</b> , <b>short</b> , <b>int</b> , or <b>long</b> , respectively.                                                                                                                                                                                                                                                                                                                                         | 2 |
| 17314 |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |   |
| 17315 |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |   |
| 17316 |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             | 2 |
| 17317 | The default number of bytes represented by the type specifiers d, f, and u shall correspond to their respective C-language types as follows. If the system claims conformance to the C-Language Development Utilities option, those specifiers shall correspond to the default sizes used in the c99 utility. Otherwise, the default sizes shall be implementation-defined.                                                                                                                                                                                                                                                                                                                                 |   |
| 17318 |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |   |
| 17319 |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |   |
| 17320 |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |   |
| 17321 |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |   |
| 17322 | For the type specifier characters d and u, the default number of bytes shall correspond to the size of a basic integer type of the implementation. For these specifier characters, the implementation shall support values of the optional number of bytes to be converted corresponding to the number of bytes in the C-language types <b>char</b> , <b>short</b> , <b>int</b> , or <b>long</b> . These numbers can also be specified by an application as the characters C, S, I, and L, respectively. The byte order used when interpreting numeric values is implementation-defined, but shall correspond to the order in which a constant of the corresponding type is stored in memory on the system. |   |
| 17323 |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |   |
| 17324 |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |   |
| 17325 |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |   |
| 17326 |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |   |
| 17327 |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |   |
| 17328 |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |   |
| 17329 |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |   |
| 17330 |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |   |
| 17331 | All type specifiers, except for s, can be followed by a mask specifier of the form & <i>number</i> . The mask value shall be AND'ed with the value of the input file before the comparison with the <i>value</i> field of the line is made. By default, the mask shall be interpreted as an unsigned decimal number. With a leading 0x or 0X, the mask shall be interpreted as an unsigned hexadecimal number; otherwise, with a leading 0, the mask shall be interpreted as an unsigned octal number.                                                                                                                                                                                                      | 2 |
| 17332 |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |   |
| 17333 |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |   |
| 17334 |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |   |
| 17335 |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |   |
| 17336 |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |   |
| 17337 | The strings <b>byte</b> , <b>short</b> , <b>long</b> , and <b>string</b> shall also be supported as type fields, being interpreted as dC, dS, dL, and s, respectively.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |   |
| 17338 |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |   |
| 17339 | <i>value</i> The <i>value</i> to be compared with the value from the file.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |   |

17340 If the specifier from the type field is **s** or **string**, then interpret the value as a string.  
 17341 Otherwise, interpret it as a number. If the value is a string, then the test shall  
 17342 succeed only when a string value exactly matches the bytes from the file.

17343 If the *value* is a string, it can contain the following sequences:

|                       |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |                                 |
|-----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---------------------------------|
| 17344      \character | The backslash-escape sequences as specified in the Base<br>Definitions volume of IEEE Std 1003.1-2001, Table 5-1, Escape<br>Sequences and Associated Actions ('\\', '\a', '\b', '\f',<br>'\n', '\r', '\t', '\v'). In addition, the escape sequence<br>'\ ' (the <backslash> character followed by a <space><br>character) shall be recognized to represent a <space> character.<br>The results of using any other character, other than an octal<br>digit, following the backslash are unspecified. | 2<br>2<br>2<br>2<br>2<br>2<br>2 |
|-----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---------------------------------|

|                   |                                                                                                                                                                                                                                        |   |
|-------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---|
| 17352      \octal | Octal sequences that can be used to represent characters with<br>specific coded values. An octal sequence shall consist of a<br>backslash followed by the longest sequence of one, two, or three<br>octal-digit characters (01234567). | 2 |
|-------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---|

17356 By default, any value that is not a string shall be interpreted as a signed decimal  
17357 number. Any such value, with a leading 0x or 0X, shall be interpreted as an  
17358 unsigned hexadecimal number; otherwise, with a leading zero, the value shall be  
17359 interpreted as an unsigned octal number.

17360 If the value is not a string, it can be preceded by a character indicating the  
17361 comparison to be performed. Permissible characters and the comparisons they  
17362 specify are as follows:

- 17363 = The test shall succeed if the value from the file equals the *value* field.
- 17364 < The test shall succeed if the value from the file is less than the *value* field.
- 17365 > The test shall succeed if the value from the file is greater than the *value* field.
- 17366 & The test shall succeed if all of the set bits in the *value* field are set in the value  
17367 from the file.
- 17368 ^ The test shall succeed if at least one of the set bits in the *value* field is not set in  
17369 the value from the file.
- 17370 x The test shall succeed if the file is large enough to contain a value of the type  
17371 specified starting at the offset specified.

|                    |                                                                                                                                                                                                                                                                                                                                                                                                     |
|--------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 17372      message | The <i>message</i> to be printed if the test succeeds. The <i>message</i> shall be interpreted<br>using the notation for the <i>printf</i> formatting specification; see <i>printf</i> . If the <i>value</i><br>field was a string, then the value from the file shall be the argument for the <i>printf</i><br>formatting specification; otherwise, the value from the file shall be the argument. |
|--------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

#### 17376 EXIT STATUS

17377 The following exit values shall be returned:

- 17378 0 Successful completion.
- 17379 >0 An error occurred.

#### 17380 CONSEQUENCES OF ERRORS

17381 Default.

## 17382 APPLICATION USAGE

17383 The *file* utility can only be required to guess at many of the file types because only exhaustive  
 17384 testing can determine some types with certainty. For example, binary data on some  
 17385 implementations might match the initial segment of an executable or a *tar* archive.

17386 Note that the table indicates that the output contains the stated string. Systems may add text  
 17387 before or after the string. For executables, as an example, the machine architecture and various  
 17388 facts about how the file was link-edited may be included. Note also that on systems that  
 17389 recognize shell script files starting with "#!" as executable files, these may be identified as  
 17390 executable binary files rather than as shell scripts.

## 17391 EXAMPLES

17392 Determine whether an argument is a binary executable file:

```
17393 file "$1" | grep -Fq executable &&
17394 printf "%s is executable.\n" "$1"
```

## 17395 RATIONALE

17396 The **-f** option was omitted because the same effect can (and should) be obtained using the *xargs*  
 17397 utility.

17398 Historical versions of the *file* utility attempt to identify the following types of files: symbolic link,  
 17399 directory, character special, block special, socket, *tar* archive, *cpio* archive, SCCS archive, archive  
 17400 library, empty, *compress* output, *pack* output, binary data, C source, FORTRAN source, assembler  
 17401 source, *nroff/troff/eqn/tbl* source *troff* output, shell script, C shell script, English text, ASCII text,  
 17402 various executables, APL workspace, compiled terminfo entries, and CURSES screen images.  
 17403 Only those types that are reasonably well specified in POSIX or are directly related to POSIX  
 17404 utilities are listed in the table.

17405 Historical systems have used a “magic file” named */etc/magic* to help identify file types. Because  
 17406 it is generally useful for users and scripts to be able to identify special file types, the **-m** flag and  
 17407 a portable format for user-created magic files has been specified. No requirement is made that an  
 17408 implementation of *file* use this method of identifying files, only that users be permitted to add  
 17409 their own classifying tests.

17410 In addition, three options have been added to historical practice. The **-d** flag has been added to  
 17411 permit users to cause their tests to follow any default system tests. The **-i** flag has been added to  
 17412 permit users to test portably for regular files in shell scripts. The **-M** flag has been added to  
 17413 permit users to ignore any default system tests.

17414 The IEEE Std 1003.1-2001 description of default system tests and the interaction between the **-d**, 1  
 17415 **-M**, and **-m** options did not clearly indicate that there were two types of “default system tests”. 1  
 17416 The “position-sensitive tests” determine file types by looking for certain string or binary values 1  
 17417 at specific offsets in the file being examined. These position-sensitive tests were implemented in 1  
 17418 historical systems using the magic file described above. Some of these tests are now built into 1  
 17419 the *file* utility itself on some implementations so the output can provide more detail than can be 1  
 17420 provided by magic files. For example, a magic file can easily identify a **core** file on most 1  
 17421 implementations, but cannot name the program file that dropped the core. A magic file could 1  
 17422 produce output such as: 1

```
17423 /home/dwc/core: ELF 32-bit MSB core file SPARC Version 1 1
```

17424 but by building the test into the *file* utility, you could get output such as: 1

```
17425 /home/dwc/core: ELF 32-bit MSB core file SPARC Version 1, from 'testprog' 1
```

17426 These extended built-in tests are still to be treated as position-sensitive default system tests even 1  
 17427 if they are not listed in */etc/magic* or any other magic file. 1

17428 The context-sensitive default system tests were always built into the *file* utility. These tests 1  
 17429 looked for language constructs in text files trying to identify shell scripts, C, FORTRAN, and 1  
 17430 other computer language source files, and even plain text files. With the addition of the **-m** and 1  
 17431 **-M** options the distinction between position-sensitive and context-sensitive default system 1  
 17432 tests became important because the order of testing is important. The context-sensitive system 1  
 17433 default tests should never be applied before any position-sensitive tests even if the **-d** option is 1  
 17434 specified before a **-m** option or **-M** option due to the high probability that the context-sensitive 1  
 17435 system default tests will incorrectly identify arbitrary text files as text files before position- 1  
 17436 sensitive tests specified by the **-m** or **-M** option would be applied to give a more accurate 1  
 17437 identification. 1

17438 Leaving the meaning of **-M** – and **-m** – unspecified allows an existing prototype of these 1  
 17439 options to continue to work in a backwards-compatible manner. (In that implementation, **-M** – 1  
 17440 was roughly equivalent to **-d** in IEEE Std 1003.1-2001.) 1

17441 The historical **-c** option was omitted as not particularly useful to users or portable shell scripts. 1  
 17442 In addition, a reasonable implementation of the *file* utility would report any errors found each 1  
 17443 time the magic file is read.

17444 The historical format of the magic file was the same as that specified by the Rationale in the 1  
 17445 ISO POSIX-2: 1993 standard for the *offset*, *value*, and *message* fields; however, it used less precise 1  
 17446 type fields than the format specified by the current normative text. The new type field values are 1  
 17447 a superset of the historical ones.

17448 The following is an example magic file:

```
17449 0 short 070707 cpio archive
17450 0 short 0143561 Byte-swapped cpio archive
17451 0 string 070707 ASCII cpio archive
17452 0 long 0177555 Very old archive
17453 0 short 0177545 Old archive
17454 0 short 017437 Old packed data
17455 0 string \037\036 Packed data
17456 0 string \377\037 Compacted data
17457 0 string \037\235 Compressed data
17458 >2 byte&0x80 >0 Block compressed
17459 >2 byte&0x1f x %d bits
17460 0 string \032\001 Compiled Termino Entry
17461 0 short 0433 Curses screen image
17462 0 short 0434 Curses screen image
17463 0 string <ar> System V Release 1 archive
17464 0 string !<arch>\n__.SYMDEF Archive random library
17465 0 string !<arch> Archive
17466 0 string ARF_BEGARF PHIGS clear text archive
17467 0 long 0x137A2950 Scalable OpenFont binary
17468 0 long 0x137A2951 Encrypted scalable OpenFont binary
```

17469 The use of a basic integer data type is intended to allow the implementation to choose a word 1  
 17470 size commonly used by applications on that architecture.

17471 Previous versions of this standard allowed for implementations with bytes other than eight bits, 2  
 17472 but this has been modified in this version. 2

**17473 FUTURE DIRECTIONS**

17474 None.

**17475 SEE ALSO**

17476 *ar, ls, pax*

**17477 CHANGE HISTORY**

17478 First released in Issue 4.

**17479 Issue 6**

17480 This utility is marked as part of the User Portability Utilities option.

17481 Options and an EXTENDED DESCRIPTION are added as specified in the IEEE P1003.2b draft  
17482 standard.

17483 IEEE PASC Interpretations 1003.2 #192 and #178 are applied.

17484 IEEE Std 1003.1-2001/Cor 1-2002, item XCU/TC1/D6/25 is applied, making major changes to 1  
17485 address ambiguities raised in defect reports. 1

17486 IEEE Std 1003.1-2001/Cor 1-2002, item XCU/TC1/D6/26 is applied, making it clear in the 1  
17487 OPTIONS section that the **-m**, **-d**, and **-M** options do not comply with Guideline 11 of the 1  
17488 Utility Syntax Guidelines. 1

17489 IEEE Std 1003.1-2001/Cor 2-2004, item XCU/TC2/D6/10 is applied, clarifying the specification 2  
17490 characters. 2

17491 IEEE Std 1003.1-2001/Cor 2-2004, item XCU/TC2/D6/11 is applied, allowing application 2  
17492 writers to create portable magic files that can match characters in strings, and allowing common 2  
17493 extensions found in existing implementations. 2

17494 IEEE Std 1003.1-2001/Cor 2-2004, item XCU/TC2/D6/12 is applied, removing text describing 2  
17495 behavior on systems with bytes consisting of more than eight bits. 2

## 17496 NAME

17497 find — find files

## 17498 SYNOPSIS

17499 `find [-H | -L] path ... [operand_expression ...]`

## 17500 DESCRIPTION

17501 The *find* utility shall recursively descend the directory hierarchy from each file specified by *path*,  
17502 evaluating a Boolean expression composed of the primaries described in the OPERANDS section  
17503 for each file encountered.

17504 The *find* utility shall be able to descend to arbitrary depths in a file hierarchy and shall not fail  
17505 due to path length limitations (unless a *path* operand specified by the application exceeds  
17506 {PATH\_MAX} requirements).

17507 The *find* utility shall detect infinite loops; that is, entering a previously visited directory that is an  
17508 ancestor of the last file encountered. When it detects an infinite loop, *find* shall write a  
17509 diagnostic message to standard error and shall either recover its position in the hierarchy or  
17510 terminate.

## 17511 OPTIONS

17512 The *find* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section  
17513 12.2, Utility Syntax Guidelines.

17514 The following options shall be supported by the implementation:

17515 **-H** Cause the file information and file type evaluated for each symbolic link  
17516 encountered on the command line to be those of the file referenced by the link, and  
17517 not the link itself. If the referenced file does not exist, the file information and type  
17518 shall be for the link itself. File information for all symbolic links not on the  
17519 command line shall be that of the link itself.

17520 **-L** Cause the file information and file type evaluated for each symbolic link to be  
17521 those of the file referenced by the link, and not the link itself.

17522 Specifying more than one of the mutually-exclusive options **-H** and **-L** shall not be considered  
17523 an error. The last option specified shall determine the behavior of the utility.

## 17524 OPERANDS

17525 The following operands shall be supported:

17526 The *path* operand is a pathname of a starting point in the directory hierarchy.

17527 The first argument that starts with a '**-**', or is a '**!**' or a '**(**', and all subsequent arguments  
17528 shall be interpreted as an *expression* made up of the following primaries and operators. In the  
17529 descriptions, wherever *n* is used as a primary argument, it shall be interpreted as a decimal  
17530 integer optionally preceded by a plus ('+') or minus ('-') sign, as follows:

17531 **+n** More than *n*.

17532 **n** Exactly *n*.

17533 **-n** Less than *n*.

17534 The following primaries shall be supported:

17535 **-name pattern**

17536 The primary shall evaluate as true if the basename of the filename being examined  
17537 matches *pattern* using the pattern matching notation described in Section 2.13 (on  
17538 page 62).

|       |                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
|-------|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 17539 | <b>-nouser</b>       | The primary shall evaluate as true if the file belongs to a user ID for which the <i>getpwuid()</i> function defined in the System Interfaces volume of IEEE Std 1003.1-2001 (or equivalent) returns NULL.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| 17540 |                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| 17541 |                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| 17542 | <b>-nogroup</b>      | The primary shall evaluate as true if the file belongs to a group ID for which the <i>getgrgid()</i> function defined in the System Interfaces volume of IEEE Std 1003.1-2001 (or equivalent) returns NULL.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| 17543 |                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| 17544 |                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| 17545 | <b>-xdev</b>         | The primary shall always evaluate as true; it shall cause <i>find</i> not to continue descending past directories that have a different device ID ( <i>st_dev</i> , see the <i>stat()</i> function defined in the System Interfaces volume of IEEE Std 1003.1-2001). If any <b>-xdev</b> primary is specified, it shall apply to the entire expression even if the <b>-xdev</b> primary would not normally be evaluated.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| 17546 |                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| 17547 |                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| 17548 |                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| 17549 |                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| 17550 | <b>-prune</b>        | The primary shall always evaluate as true; it shall cause <i>find</i> not to descend the current pathname if it is a directory. If the <b>-depth</b> primary is specified, the <b>-prune</b> primary shall have no effect.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| 17551 |                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| 17552 |                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| 17553 | <b>-perm [-]mode</b> | <p>The <i>mode</i> argument is used to represent file mode bits. It shall be identical in format to the <i>symbolic_mode</i> operand described in <i>chmod</i>, and shall be interpreted as follows. To start, a template shall be assumed with all file mode bits cleared. An <i>op</i> symbol of '+' shall set the appropriate mode bits in the template; '-' shall clear the appropriate bits; '=' shall set the appropriate mode bits, without regard to the contents of process' file mode creation mask. The <i>op</i> symbol of '-' cannot be the first character of <i>mode</i>; this avoids ambiguity with the optional leading hyphen. Since the initial mode is all bits off, there are not any symbolic modes that need to use '-' as the first character.</p> <p>If the hyphen is omitted, the primary shall evaluate as true when the file permission bits exactly match the value of the resulting template.</p> <p>Otherwise, if <i>mode</i> is prefixed by a hyphen, the primary shall evaluate as true if at least all the bits in the resulting template are set in the file permission bits.</p> |
| 17554 |                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| 17555 |                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| 17556 |                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| 17557 |                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| 17558 |                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| 17559 |                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| 17560 |                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| 17561 |                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| 17562 |                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| 17563 |                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| 17564 |                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| 17565 |                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| 17566 |                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| 17567 | <b>-perm [-]onum</b> | <p>If the hyphen is omitted, the primary shall evaluate as true when the file permission bits exactly match the value of the octal number <i>onum</i> and only the bits corresponding to the octal mask 07777 shall be compared. (See the description of the octal <i>mode</i> in <i>chmod</i>.) Otherwise, if <i>onum</i> is prefixed by a hyphen, the primary shall evaluate as true if at least all of the bits specified in <i>onum</i> that are also set in the octal mask 07777 are set.</p>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
| 17568 |                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| 17569 |                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| 17570 |                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| 17571 |                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| 17572 |                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| 17573 |                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| 17574 | <b>-type c</b>       | The primary shall evaluate as true if the type of the file is <i>c</i> , where <i>c</i> is 'b', 'c', 'd', 'l', 'p', 'f', or 's' for block special file, character special file, directory, symbolic link, FIFO, regular file, or socket, respectively.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
| 17575 |                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| 17576 |                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| 17577 | <b>-links n</b>      | The primary shall evaluate as true if the file has <i>n</i> links.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
| 17578 | <b>-user uname</b>   | The primary shall evaluate as true if the file belongs to the user <i>uname</i> . If <i>uname</i> is a decimal integer and the <i>getpwnam()</i> (or equivalent) function does not return a valid user name, <i>uname</i> shall be interpreted as a user ID.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
| 17579 |                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| 17580 |                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| 17581 | <b>-group gname</b>  | The primary shall evaluate as true if the file belongs to the group <i>gname</i> . If <i>gname</i> is a decimal integer and the <i>getgrnam()</i> (or equivalent) function does not return a valid group name, <i>gname</i> shall be interpreted as a group ID.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| 17582 |                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| 17583 |                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| 17584 |                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |

|       |                                                              |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
|-------|--------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 17585 | <b>-size</b> <i>n[c]</i>                                     | The primary shall evaluate as true if the file size in bytes, divided by 512 and rounded up to the next integer, is <i>n</i> . If <i>n</i> is followed by the character 'c', the size shall be in bytes.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
| 17588 | <b>-atime</b> <i>n</i>                                       | The primary shall evaluate as true if the file access time subtracted from the initialization time, divided by 86 400 (with any remainder discarded), is <i>n</i> .                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| 17590 | <b>-ctime</b> <i>n</i>                                       | The primary shall evaluate as true if the time of last change of file status information subtracted from the initialization time, divided by 86 400 (with any remainder discarded), is <i>n</i> .                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| 17593 | <b>-mtime</b> <i>n</i>                                       | The primary shall evaluate as true if the file modification time subtracted from the initialization time, divided by 86 400 (with any remainder discarded), is <i>n</i> .                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| 17595 | <b>-exec</b> <i>utility_name</i> [ <i>argument</i> ...];     |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
| 17596 | <b>-exec</b> <i>utility_name</i> [ <i>argument</i> ...] {} + | The end of the primary expression shall be punctuated by a semicolon or by a plus sign. Only a plus sign that follows an argument containing the two characters "{}" shall punctuate the end of the primary expression. Other uses of the plus sign shall not be treated as special.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| 17601 |                                                              | If the primary expression is punctuated by a semicolon, the utility <i>utility_name</i> shall be invoked once for each pathname and the primary shall evaluate as true if the utility returns a zero value as exit status. A <i>utility_name</i> or <i>argument</i> containing only the two characters "{}" shall be replaced by the current pathname.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| 17605 |                                                              | If the primary expression is punctuated by a plus sign, the primary shall always evaluate as true, and the pathnames for which the primary is evaluated shall be aggregated into sets. The utility <i>utility_name</i> shall be invoked once for each set of aggregated pathnames. Each invocation shall begin after the last pathname in the set is aggregated, and shall be completed before the <i>find</i> utility exits and before the first pathname in the next set (if any) is aggregated for this primary, but it is otherwise unspecified whether the invocation occurs before, during, or after the evaluations of other primaries. If any invocation returns a non-zero value as exit status, the <i>find</i> utility shall return a non-zero exit status. An argument containing only the two characters "{}" shall be replaced by the set of aggregated pathnames, with each pathname passed as a separate argument to the invoked utility in the same order that it was aggregated. The size of any set of two or more pathnames shall be limited such that execution of the utility does not cause the system's {ARG_MAX} limit to be exceeded. If more than one argument containing only the two characters "{}" is present, the behavior is unspecified. |
| 17620 |                                                              | If a <i>utility_name</i> or <i>argument</i> string contains the two characters "{}", but not just the two characters "{}", it is implementation-defined whether <i>find</i> replaces those two characters or uses the string without change. The current directory for the invocation of <i>utility_name</i> shall be the same as the current directory when the <i>find</i> utility was started. If the <i>utility_name</i> names any of the special built-in utilities (see Section 2.14 (on page 64)), the results are undefined.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| 17626 | <b>-ok</b> <i>utility_name</i> [ <i>argument</i> ...];       | The <b>-ok</b> primary shall be equivalent to <b>-exec</b> , except that the use of a plus sign to punctuate the end of the primary expression need not be supported, and <i>find</i> shall request affirmation of the invocation of <i>utility_name</i> using the current file as an argument by writing to standard error as described in the STDERR section. If the response on standard input is affirmative, the utility shall be invoked. Otherwise, the command shall not be invoked and the value of the <b>-ok</b> operand shall be false.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |

|       |                                                                                                                                                                                                                                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
|-------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 17633 | <b>-print</b>                                                                                                                                                                                                                                                     | The primary shall always evaluate as true; it shall cause the current pathname to be written to standard output.                                                                                                                                                                                                                                                                                                                                                 |
| 17634 |                                                                                                                                                                                                                                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| 17635 | <b>-newer file</b>                                                                                                                                                                                                                                                | The primary shall evaluate as true if the modification time of the current file is more recent than the modification time of the file named by the pathname <i>file</i> .                                                                                                                                                                                                                                                                                        |
| 17636 |                                                                                                                                                                                                                                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| 17637 | <b>-depth</b>                                                                                                                                                                                                                                                     | The primary shall always evaluate as true; it shall cause descent of the directory hierarchy to be done so that all entries in a directory are acted on before the directory itself. If a <b>-depth</b> primary is not specified, all entries in a directory shall be acted on after the directory itself. If any <b>-depth</b> primary is specified, it shall apply to the entire expression even if the <b>-depth</b> primary would not normally be evaluated. |
| 17638 |                                                                                                                                                                                                                                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| 17639 |                                                                                                                                                                                                                                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| 17640 |                                                                                                                                                                                                                                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| 17641 |                                                                                                                                                                                                                                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| 17642 |                                                                                                                                                                                                                                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| 17643 |                                                                                                                                                                                                                                                                   | The primaries can be combined using the following operators (in order of decreasing precedence):                                                                                                                                                                                                                                                                                                                                                                 |
| 17644 |                                                                                                                                                                                                                                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| 17645 | ( <i>expression</i> )                                                                                                                                                                                                                                             | True if <i>expression</i> is true.                                                                                                                                                                                                                                                                                                                                                                                                                               |
| 17646 | ! <i>expression</i>                                                                                                                                                                                                                                               | Negation of a primary; the unary NOT operator.                                                                                                                                                                                                                                                                                                                                                                                                                   |
| 17647 | <i>expression</i> [-a] <i>expression</i>                                                                                                                                                                                                                          |                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| 17648 |                                                                                                                                                                                                                                                                   | Conjunction of primaries; the AND operator is implied by the juxtaposition of two primaries or made explicit by the optional <b>-a</b> operator. The second expression shall not be evaluated if the first expression is false.                                                                                                                                                                                                                                  |
| 17649 |                                                                                                                                                                                                                                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| 17650 |                                                                                                                                                                                                                                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| 17651 | <i>expression</i> -o <i>expression</i>                                                                                                                                                                                                                            |                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| 17652 |                                                                                                                                                                                                                                                                   | Alternation of primaries; the OR operator. The second expression shall not be evaluated if the first expression is true.                                                                                                                                                                                                                                                                                                                                         |
| 17653 |                                                                                                                                                                                                                                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| 17654 | If no <i>expression</i> is present, <b>-print</b> shall be used as the expression. Otherwise, if the given expression does not contain any of the primaries <b>-exec</b> , <b>-ok</b> , or <b>-print</b> , the given expression shall be effectively replaced by: |                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| 17655 |                                                                                                                                                                                                                                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| 17656 |                                                                                                                                                                                                                                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| 17657 | ( <i>given_expression</i> ) -print                                                                                                                                                                                                                                |                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| 17658 | The <b>-user</b> , <b>-group</b> , and <b>-newer</b> primaries each shall evaluate their respective arguments only once.                                                                                                                                          |                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| 17659 |                                                                                                                                                                                                                                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| 17660 | <b>STDIN</b>                                                                                                                                                                                                                                                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| 17661 | If the <b>-ok</b> primary is used, the response shall be read from the standard input. An entire line shall be read as the response. Otherwise, the standard input shall not be used.                                                                             |                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| 17662 |                                                                                                                                                                                                                                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| 17663 | <b>INPUT FILES</b>                                                                                                                                                                                                                                                |                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| 17664 | None.                                                                                                                                                                                                                                                             |                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| 17665 | <b>ENVIRONMENT VARIABLES</b>                                                                                                                                                                                                                                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| 17666 | The following environment variables shall affect the execution of <i>find</i> :                                                                                                                                                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| 17667 | <b>LANG</b>                                                                                                                                                                                                                                                       | Provide a default value for the internationalization variables that are unset or null. (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2, Internationalization Variables for the precedence of internationalization variables used to determine the values of locale categories.)                                                                                                                                                            |
| 17668 |                                                                                                                                                                                                                                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| 17669 |                                                                                                                                                                                                                                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| 17670 |                                                                                                                                                                                                                                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| 17671 | <b>LC_ALL</b>                                                                                                                                                                                                                                                     | If set to a non-empty string value, override the values of all the other internationalization variables.                                                                                                                                                                                                                                                                                                                                                         |
| 17672 |                                                                                                                                                                                                                                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| 17673 | <b>LC_COLLATE</b>                                                                                                                                                                                                                                                 | Determine the locale for the behavior of ranges, equivalence classes, and multi-character collating elements used in the pattern matching notation for the <b>-n</b> option and in the extended regular expression defined for the <b>yesexpr</b> locale                                                                                                                                                                                                         |
| 17674 |                                                                                                                                                                                                                                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| 17675 |                                                                                                                                                                                                                                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| 17676 |                                                                                                                                                                                                                                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |

|           |                               |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
|-----------|-------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 17677     |                               | keyword in the <i>LC_MESSAGES</i> category.                                                                                                                                                                                                                                                                                                                                                                                                                                               |
| 17678     | <i>LC_CTYPE</i>               | This variable determines the locale for the interpretation of sequences of bytes of text data as characters (for example, single-byte as opposed to multi-byte characters in arguments), the behavior of character classes within the pattern matching notation used for the <b>-n</b> option, and the behavior of character classes within regular expressions used in the extended regular expression defined for the <b>yesexpr</b> locale keyword in the <i>LC_MESSAGES</i> category. |
| 17684     | <i>LC_MESSAGES</i>            |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| 17685     |                               | Determine the locale for the processing of affirmative responses that should be used to affect the format and contents of diagnostic messages written to standard error.                                                                                                                                                                                                                                                                                                                  |
| 17688 XSI | <i>NLSPATH</i>                | Determine the location of message catalogs for the processing of <i>LC_MESSAGES</i> .                                                                                                                                                                                                                                                                                                                                                                                                     |
| 17689     | <i>PATH</i>                   | Determine the location of the <i>utility_name</i> for the <b>-exec</b> and <b>-ok</b> primaries, as described in the Base Definitions volume of IEEE Std 1003.1-2001, Chapter 8, Environment Variables.                                                                                                                                                                                                                                                                                   |
| 17692     | <b>ASYNCHRONOUS EVENTS</b>    |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| 17693     |                               | Default.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| 17694     | <b>STDOUT</b>                 |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| 17695     |                               | The <b>-print</b> primary shall cause the current pathnames to be written to standard output. The format shall be:                                                                                                                                                                                                                                                                                                                                                                        |
| 17697     |                               | "%s\n" , <path>                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| 17698     | <b>STDERR</b>                 |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| 17699     |                               | The <b>-ok</b> primary shall write a prompt to standard error containing at least the <i>utility_name</i> to be invoked and the current pathname. In the POSIX locale, the last non-<blank> in the prompt shall be '?'. The exact format used is unspecified.                                                                                                                                                                                                                             |
| 17702     |                               | Otherwise, the standard error shall be used only for diagnostic messages.                                                                                                                                                                                                                                                                                                                                                                                                                 |
| 17703     | <b>OUTPUT FILES</b>           |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| 17704     |                               | None.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| 17705     | <b>EXTENDED DESCRIPTION</b>   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| 17706     |                               | None.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| 17707     | <b>EXIT STATUS</b>            |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| 17708     |                               | The following exit values shall be returned:                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| 17709     |                               | 0 All <i>path</i> operands were traversed successfully.                                                                                                                                                                                                                                                                                                                                                                                                                                   |
| 17710     |                               | >0 An error occurred.                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| 17711     | <b>CONSEQUENCES OF ERRORS</b> |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| 17712     |                               | Default.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |

## 17713 APPLICATION USAGE

17714 When used in operands, pattern matching notation, semicolons, opening parentheses, and  
17715 closing parentheses are special to the shell and must be quoted (see Section 2.2 (on page 30)).

17716 The bit that is traditionally used for sticky (historically 01000) is specified in the **-perm** primary  
17717 using the octal number argument form. Since this bit is not defined by this volume of  
17718 IEEE Std 1003.1-2001, applications must not assume that it actually refers to the traditional  
17719 sticky bit.

## 17720 EXAMPLES

17721 1. The following commands are equivalent:

17722 find .  
17723 find . -print

17724 They both write out the entire directory hierarchy from the current directory.

17725 2. The following command:

17726 find / \(\ -name tmp -o -name '\*.\xx' \) -atime +7 -exec rm {} \;  
17727 removes all files named **tmp** or ending in **.xx** that have not been accessed for seven or more  
17728 24-hour periods.

17729 3. The following command:

17730 find . -perm -o+w,+s

17731 prints (**-print** is assumed) the names of all files in or below the current directory, with all  
17732 of the file permission bits **S\_ISUID**, **S\_ISGID**, and **S\_IWOTH** set.

17733 4. The following command:

17734 find . -name SCCS -prune -o -print

17735 recursively prints pathnames of all files in the current directory and below, but skips  
17736 directories named **SCCS** and files in them.

17737 5. The following command:

17738 find . -print -name SCCS -prune

17739 behaves as in the previous example, but prints the names of the **SCCS** directories.

17740 6. The following command is roughly equivalent to the **-nt** extension to *test*:

17741 if [ -n "\\$(find file1 -prune -newer file2)" ]; then  
17742 printf %s\n "file1 is newer than file2"  
17743 fi

17744 7. The descriptions of **-atime**, **-ctime**, and **-mtime** use the terminology *n* “86 400 second  
17745 periods (days)”. For example, a file accessed at 23:59 is selected by:

17746 find . -atime -1 -print

17747 at 00:01 the next day (less than 24 hours later, not more than one day ago); the midnight  
17748 boundary between days has no effect on the 24-hour calculation.

## 17749 RATIONALE

17750 The **-a** operator was retained as an optional operator for compatibility with historical shell  
17751 scripts, even though it is redundant with expression concatenation.

17752 The descriptions of the '-' modifier on the *mode* and *onum* arguments to the **-perm** primary  
17753 agree with historical practice on BSD and System V implementations. System V and BSD  
17754 documentation both describe it in terms of checking additional bits; in fact, it uses the same bits,  
17755 but checks for having at least all of the matching bits set instead of having exactly the matching  
17756 bits set.

17757 The exact format of the interactive prompts is unspecified. Only the general nature of the  
17758 contents of prompts are specified because:

- 17759 • Implementations may desire more descriptive prompts than those used on historical  
17760 implementations.
- 17761 • Since the historical prompt strings do not terminate with <newline>s, there is no portable  
17762 way for another program to interact with the prompts of this utility via pipes.

17763 Therefore, an application using this prompting option relies on the system to provide the most  
17764 suitable dialog directly with the user, based on the general guidelines specified.

17765 The **-name** *file* operand was changed to use the shell pattern matching notation so that *find* is  
17766 consistent with other utilities using pattern matching.

17767 The **-size** operand refers to the size of a file, rather than the number of blocks it may occupy in  
17768 the file system. The intent is that the *st\_size* field defined in the System Interfaces volume of  
17769 IEEE Std 1003.1-2001 should be used, not the *st\_blocks* found in historical implementations. There  
17770 are at least two reasons for this:

- 17771 1. In both System V and BSD, *find* only uses *st\_size* in size calculations for the operands  
17772 specified by this volume of IEEE Std 1003.1-2001. (BSD uses *st\_blocks* only when processing  
17773 the **-ls** primary.)
- 17774 2. Users usually think of file size in terms of bytes, which is also the unit used by the *ls* utility  
17775 for the output from the **-l** option. (In both System V and BSD, *ls* uses *st\_size* for the **-l**  
17776 option size field and uses *st\_blocks* for the *ls -s* calculations. This volume of  
17777 IEEE Std 1003.1-2001 does not specify *ls -s*.)

17778 The descriptions of **-atime**, **-ctime**, and **-mtime** were changed from the SVID description of *n* 2  
17779 "days" to *n* being the result of the integer division of the time difference in seconds by 86 400. 2  
17780 The description is also different in terms of the exact timeframe for the *n* case (*versus* the *+n* or  
17781 *-n*), but it matches all known historical implementations. It refers to one 86 400 second period in  
17782 the past, not any time from the beginning of that period to the current time. For example, **-atime** 2  
17783 2 is true if the file was accessed any time in the period from 72 hours to 48 hours ago. 2

17784 Historical implementations do not modify "`{}`" when it appears as a substring of an **-exec** or  
17785 **-ok** *utility\_name* or argument string. There have been numerous user requests for this extension,  
17786 so this volume of IEEE Std 1003.1-2001 allows the desired behavior. At least one recent  
17787 implementation does support this feature, but encountered several problems in managing  
17788 memory allocation and dealing with multiple occurrences of "`{}`" in a string while it was being  
17789 developed, so it is not yet required behavior.

17790 Assuming the presence of **-print** was added to correct a historical pitfall that plagues novice  
17791 users, it is entirely upwards-compatible from the historical System V *find* utility. In its simplest  
17792 form (*find directory*), it could be confused with the historical BSD fast *find*. The BSD developers  
17793 agreed that adding **-print** as a default expression was the correct decision and have added the  
17794 fast *find* functionality within a new utility called *locate*.

17795 Historically, the **-L** option was implemented using the primary **-follow**. The **-H** and **-L** options  
17796 were added for two reasons. First, they offer a finer granularity of control and consistency with  
17797 other programs that walk file hierarchies. Second, the **-follow** primary always evaluated to true.

17798 As they were historically really global variables that took effect before the traversal began, some  
17799 valid expressions had unexpected results. An example is the expression **-print -o -follow**.  
17800 Because **-print** always evaluates to true, the standard order of evaluation implies that **-follow**  
17801 would never be evaluated. This was never the case. Historical practice for the **-follow** primary,  
17802 however, is not consistent. Some implementations always follow symbolic links on the  
17803 command line whether **-follow** is specified or not. Others follow symbolic links on the  
17804 command line only if **-follow** is specified. Both behaviors are provided by the **-H** and **-L**  
17805 options, but scripts using the current **-follow** primary would be broken if the **-follow** option is  
17806 specified to work either way.

17807 Since the **-L** option resolves all symbolic links and the **-type l** primary is true for symbolic links  
17808 that still exist after symbolic links have been resolved, the command:

17809 `find -L . -type l`

17810 prints a list of symbolic links reachable from the current directory that do not resolve to  
17811 accessible files.

17812 A feature of SVR4's *find* utility was the **-exec** primary's + terminator. This allowed filenames  
17813 containing special characters (especially <newline>s) to be grouped together without the  
17814 problems that occur if such filenames are piped to *xargs*. Other implementations have added  
17815 other ways to get around this problem, notably a **-print0** primary that wrote filenames with a  
17816 null byte terminator. This was considered here, but not adopted. Using a null terminator meant  
17817 that any utility that was going to process *find*'s **-print0** output had to add a new option to parse  
17818 the null terminators it would now be reading.

17819 The "**-exec ... {} +**" syntax adopted was a result of IEEE PASC Interpretation 1003.2 #210.  
17820 It should be noted that this is an incompatible change to the ISO/IEC 9899:1999 standard. For  
17821 example, the following command prints all files with a '-' after their name if they are regular  
17822 files, and a '+' otherwise:

17823 `find / -type f -exec echo {} - ';' -o -exec echo {} + ';'`

17824 The change invalidates usage like this. Even though the previous standard stated that this usage  
17825 would work, in practice many did not support it and the standard developers felt it better to  
17826 now state that this was not allowable.

## 17827 FUTURE DIRECTIONS

17828 None.

## 17829 SEE ALSO

17830 Section 2.2 (on page 30), Section 2.13 (on page 62), Section 2.14 (on page 64), *chmod*, *pax*, *sh*, *test*,  
17831 the System Interfaces volume of IEEE Std 1003.1-2001, *getgrgid()*, *getpwuid()*, *stat()*

## 17832 CHANGE HISTORY

17833 First released in Issue 2.

## 17834 Issue 5

17835 The FUTURE DIRECTIONS section is added.

## 17836 Issue 6

17837 The following new requirements on POSIX implementations derive from alignment with the  
17838 Single UNIX Specification:

- 17839 • The **-perm [-]onum** primary is supported.

17840 The *find* utility is aligned with the IEEE P1003.2b draft standard, to include processing of  
17841 symbolic links and changes to the description of the **atime**, **ctime**, and **mtime** operands.

|       |                                                                                                                                          |   |
|-------|------------------------------------------------------------------------------------------------------------------------------------------|---|
| 17842 | IEEE PASC Interpretation 1003.2 #210 is applied, extending the <b>-exec</b> operand.                                                     |   |
| 17843 | IEEE Std 1003.1-2001/Cor 2-2004, item XCU/TC2/D6/13 is applied, updating the RATIONALE section to be consistent with the normative text. | 2 |
| 17844 |                                                                                                                                          | 2 |

## 17845 NAME

17846        fold — filter for folding lines

## 17847 SYNOPSIS

17848        fold [-bs][-w width][file...]

## 17849 DESCRIPTION

17850        The *fold* utility is a filter that shall fold lines from its input files, breaking the lines to have a maximum of *width* column positions (or bytes, if the **-b** option is specified). Lines shall be broken by the insertion of a <newline> such that each output line (referred to later in this section as a *segment*) is the maximum width possible that does not exceed the specified number of column positions (or bytes). A line shall not be broken in the middle of a character. The behavior is undefined if *width* is less than the number of columns any single character in the input would occupy.

17857        If the <carriage-return>s, <backspace>s, or <tab>s are encountered in the input, and the **-b** option is not specified, they shall be treated specially:

17859        <backspace> The current count of line width shall be decremented by one, although the count  
17860              never shall become negative. The *fold* utility shall not insert a <newline>  
17861              immediately before or after any <backspace>.

17862        <carriage-return>

17863              The current count of line width shall be set to zero. The *fold* utility shall not insert a  
17864              <newline> immediately before or after any <carriage-return>.

17865        <tab>

17866              Each <tab> encountered shall advance the column position pointer to the next tab  
stop. Tab stops shall be at each column position *n* such that *n* modulo 8 equals 1.

## 17867 OPTIONS

17868        The *fold* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section  
17869              12.2, Utility Syntax Guidelines.

17870        The following options shall be supported:

17871        **-b**        Count *width* in bytes rather than column positions.

17872        **-s**        If a segment of a line contains a <blank> within the first *width* column positions (or  
17873              bytes), break the line after the last such <blank> meeting the width constraints. If  
17874              there is no <blank> meeting the requirements, the **-s** option shall have no effect for  
17875              that output segment of the input line.

17876        **-w width**    Specify the maximum line length, in column positions (or bytes if **-b** is specified).  
17877              The results are unspecified if *width* is not a positive decimal number. The default  
17878              value shall be 80.

## 17879 OPERANDS

17880        The following operand shall be supported:

17881        *file*        A pathname of a text file to be folded. If no *file* operands are specified, the standard  
17882              input shall be used.

## 17883 STDIN

17884        The standard input shall be used only if no *file* operands are specified. See the INPUT FILES  
17885              section.

## 17886 INPUT FILES

17887 If the **-b** option is specified, the input files shall be text files except that the lines are not limited  
17888 to {LINE\_MAX} bytes in length. If the **-b** option is not specified, the input files shall be text files.

## 17889 ENVIRONMENT VARIABLES

17890 The following environment variables shall affect the execution of *fold*:

17891 **LANG** Provide a default value for the internationalization variables that are unset or null.  
17892 (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2,  
17893 Internationalization Variables for the precedence of internationalization variables  
17894 used to determine the values of locale categories.)

17895 **LC\_ALL** If set to a non-empty string value, override the values of all the other  
17896 internationalization variables.

17897 **LC\_CTYPE** Determine the locale for the interpretation of sequences of bytes of text data as  
17898 characters (for example, single-byte as opposed to multi-byte characters in  
17899 arguments and input files), and for the determination of the width in column  
17900 positions each character would occupy on a constant-width font output device.

17901 **LC\_MESSAGES**

17902 Determine the locale that should be used to affect the format and contents of  
17903 diagnostic messages written to standard error.

17904 XSI **NLSPATH** Determine the location of message catalogs for the processing of *LC\_MESSAGES*.

## 17905 ASYNCHRONOUS EVENTS

17906 Default.

17907 **STDOUT**

17908 The standard output shall be a file containing a sequence of characters whose order shall be  
17909 preserved from the input files, possibly with inserted <newline>s.

17910 **STDERR**

17911 The standard error shall be used only for diagnostic messages.

17912 **OUTPUT FILES**

17913 None.

17914 **EXTENDED DESCRIPTION**

17915 None.

17916 **EXIT STATUS**

17917 The following exit values shall be returned:

17918 0 All input files were processed successfully.

17919 >0 An error occurred.

17920 **CONSEQUENCES OF ERRORS**

17921 Default.

**17922 APPLICATION USAGE**

17923     The *cut* and *fold* utilities can be used to create text files out of files with arbitrary line lengths. The  
17924     *cut* utility should be used when the number of lines (or records) needs to remain constant. The  
17925     *fold* utility should be used when the contents of long lines need to be kept contiguous.

17926     The *fold* utility is frequently used to send text files to printers that truncate, rather than fold, lines  
17927     wider than the printer is able to print (usually 80 or 132 column positions).

**17928 EXAMPLES**

17929     An example invocation that submits a file of possibly long lines to the printer (under the  
17930     assumption that the user knows the line width of the printer to be assigned by *lp*):

17931        *fold -w 132 bigfile | lp*

**17932 RATIONALE**

17933     Although terminal input in canonical processing mode requires the erase character (frequently  
17934     set to <backspace>) to erase the previous character (not byte or column position), terminal  
17935     output is not buffered and is extremely difficult, if not impossible, to parse correctly; the  
17936     interpretation depends entirely on the physical device that actually displays/prints/stores the  
17937     output. In all known internationalized implementations, the utilities producing output for mixed  
17938     column-width output assume that a <backspace> backs up one column position and outputs  
17939     enough <backspace>s to return to the start of the character when <backspace> is used to  
17940     provide local line motions to support underlining and emboldening operations. Since *fold*  
17941     without the **-b** option is dealing with these same constraints, <backspace> is always treated as  
17942     backing up one column position rather than backing up one character.

17943     Historical versions of the *fold* utility assumed 1 byte was one character and occupied one column  
17944     position when written out. This is no longer always true. Since the most common usage of *fold* is  
17945     believed to be folding long lines for output to limited-length output devices, this capability was  
17946     preserved as the default case. The **-b** option was added so that applications could *fold* files with  
17947     arbitrary length lines into text files that could then be processed by the standard utilities. Note  
17948     that although the width for the **-b** option is in bytes, a line is never split in the middle of a  
17949     character. (It is unspecified what happens if a width is specified that is too small to hold a single  
17950     character found in the input followed by a <newline>.)

17951     The tab stops are hardcoded to be every eighth column to meet historical practice. No new  
17952     method of specifying other tab stops was invented.

**17953 FUTURE DIRECTIONS**

17954     None.

**17955 SEE ALSO**

17956        *cut*

**17957 CHANGE HISTORY**

17958     First released in Issue 4.

**17959 Issue 6**

17960     The normative text is reworded to avoid use of the term “must” for application requirements.

## 17961 NAME

17962        fort77 — FORTRAN compiler (**FORTRAN**)

## 17963 SYNOPSIS

```
17964 FD fort77 [-c][-g][-L directory]... [-O optlevel][-o outfile][-s][-w]
17965 operand...
17966
```

## 17967 DESCRIPTION

17968        The *fort77* utility is the interface to the FORTRAN compilation system; it shall accept the full  
 17969        FORTRAN-77 language defined by the ANSI X3.9-1978 standard. The system conceptually  
 17970        consists of a compiler and link editor. The files referenced by *operands* are compiled and linked  
 17971        to produce an executable file. It is unspecified whether the linking occurs entirely within the  
 17972        operation of *fort77*; some implementations may produce objects that are not fully resolved until  
 17973        the file is executed.

17974        If the **-c** option is present, for all pathname operands of the form *file.f*, the files:

17975        \$(basename *pathname.f*).o

17976        shall be created or overwritten as the result of successful compilation. If the **-c** option is not  
 17977        specified, it is unspecified whether such .o files are created or deleted for the *file.f* operands.

17978        If there are no options that prevent link editing (such as **-c**) and all operands compile and link  
 17979        without error, the resulting executable file shall be written into the file named by the **-o** option  
 17980        (if present) or to the file **a.out**. The executable file shall be created as specified in the System  
 17981        Interfaces volume of IEEE Std 1003.1-2001, except that the file permissions shall be set to:

17982        S\_IRWXO | S\_IRWXG | S\_IRWXU

1

17983        and that the bits specified by the *umask* of the process shall be cleared.

## 17984 OPTIONS

17985        The *fort77* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section  
 17986        12.2, Utility Syntax Guidelines, except that:

- 17987        • The **-l** *library* operands have the format of options, but their position within a list of  
     17988        operands affects the order in which libraries are searched.
- 17989        • The order of specifying the multiple **-L** options is significant.
- 17990        • Conforming applications shall specify each option separately; that is, grouping option letters  
     17991        (for example, **-cg**) need not be recognized by all implementations.

17992        The following options shall be supported:

- |                                |                                                                                                                                                                                                                                                                                                                                                                                                       |
|--------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 17993 <b>-c</b>                | Suppress the link-edit phase of the compilation, and do not remove any object files<br>17994        that are produced.                                                                                                                                                                                                                                                                                |
| 17995 <b>-g</b>                | Produce symbolic information in the object or executable files; the nature of this<br>17996        information is unspecified, and may be modified by implementation-defined<br>17997        interactions with other options.                                                                                                                                                                         |
| 17998 <b>-s</b>                | Produce object or executable files, or both, from which symbolic and other<br>17999        information not required for proper execution using the <i>exec</i> family of functions<br>18000        defined in the System Interfaces volume of IEEE Std 1003.1-2001 has been removed<br>18001        (stripped). If both <b>-g</b> and <b>-s</b> options are present, the action taken is unspecified. |
| 18002 <b>-o</b> <i>outfile</i> | Use the pathname <i>outfile</i> , instead of the default <b>a.out</b> , for the executable file<br>18003        produced. If the <b>-o</b> option is present with <b>-c</b> , the result is unspecified.                                                                                                                                                                                              |

- 18004       **-L directory**    Change the algorithm of searching for the libraries named in **-l** operands to look in the directory named by the *directory* pathname before looking in the usual places. Directories named in **-L** options shall be searched in the specified order. At least ten instances of this option shall be supported in a single *fort77* command invocation. If a directory specified by a **-L** option contains a file named **libf.a**, the results are unspecified.
- 18010       **-O optlevel**    Specify the level of code optimization. If the *optlevel* option-argument is the digit '0', all special code optimizations shall be disabled. If it is the digit '1', the nature of the optimization is unspecified. If the **-O** option is omitted, the nature of the system's default optimization is unspecified. It is unspecified whether code generated in the presence of the **-O 0** option is the same as that generated when **-O** is omitted. Other *optlevel* values may be supported.
- 18016       **-w**            Suppress warnings.
- 18017        Multiple instances of **-L** options can be specified.

## 18018 OPERANDS

- 18019        An *operand* is either in the form of a pathname or the form **-l library**. At least one operand of the pathname form shall be specified. The following operands shall be supported:
- 18021        **file.f**        The pathname of a FORTRAN source file to be compiled and optionally passed to the link editor. The filename operand shall be of this form if the **-c** option is used.
- 18023        **file.a**        A library of object files typically produced by *ar*, and passed directly to the link editor. Implementations may recognize implementation-defined suffixes other than **.a** as denoting object file libraries.
- 18026        **file.o**        An object file produced by *fort77 -c* and passed directly to the link editor. Implementations may recognize implementation-defined suffixes other than **.o** as denoting object files.
- 18029        The processing of other files is implementation-defined.
- 18030        **-l library**    (The letter ell.) Search the library named:  
18031                      *liblibrary.a*
- 18032        A library is searched when its name is encountered, so the placement of a **-l** operand is significant. Several standard libraries can be specified in this manner, as described in the EXTENDED DESCRIPTION section. Implementations may recognize implementation-defined suffixes other than **.a** as denoting libraries.

## 18036 STDIN

- 18037        Not used.

## 18038 INPUT FILES

- 18039        The input file shall be one of the following: a text file containing FORTRAN source code; an object file in the format produced by *fort77 -c*; or a library of object files, in the format produced by archiving zero or more object files, using *ar*. Implementations may supply additional utilities that produce files in these formats. Additional input files are implementation-defined.
- 18043        A <tab> encountered within the first six characters on a line of source code shall cause the compiler to interpret the following character as if it were the seventh character on the line (that is, in column 7).

## 18046 ENVIRONMENT VARIABLES

18047 The following environment variables shall affect the execution of *fort77*:

18048 ***LANG*** Provide a default value for the internationalization variables that are unset or null.  
 18049 (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2,  
 18050 Internationalization Variables for the precedence of internationalization variables  
 18051 used to determine the values of locale categories.)

18052 ***LC\_ALL*** If set to a non-empty string value, override the values of all the other  
 18053 internationalization variables.

18054 ***LC\_CTYPE*** Determine the locale for the interpretation of sequences of bytes of text data as  
 18055 characters (for example, single-byte as opposed to multi-byte characters in  
 18056 arguments and input files).

18057 ***LC\_MESSAGES***

18058 Determine the locale that should be used to affect the format and contents of  
 18059 diagnostic messages written to standard error.

18060 XSI ***NLSPATH*** Determine the location of message catalogs for the processing of *LC\_MESSAGES*.

18061 ***TMPDIR*** Determine the pathname that should override the default directory for temporary  
 18062 files, if any.

## 18063 ASYNCHRONOUS EVENTS

18064 Default.

18065 **STDOUT**

18066 Not used.

18067 **STDERR**

18068 The standard error shall be used only for diagnostic messages. If more than one *file* operand  
 18069 ending in .f (or possibly other unspecified suffixes) is given, for each such file:

18070 "%s:\n", <file>

18071 may be written to allow identification of the diagnostic message with the appropriate input file.

18072 This utility may produce warning messages about certain conditions that do not warrant  
 18073 returning an error (non-zero) exit value.

18074 **OUTPUT FILES**

18075 Object files, listing files, and executable files shall be produced in unspecified formats.

18076 **EXTENDED DESCRIPTION**18077 **Standard Libraries**

18078 The *fort77* utility shall recognize the following -l operand for the standard library:

18079 **-lf** This library contains all functions referenced in the ANSI X3.9-1978 standard. This  
 18080 operand shall not be required to be present to cause a search of this library.

18081 In the absence of options that inhibit invocation of the link editor, such as -c, the *fort77* utility  
 18082 shall cause the equivalent of a -lf operand to be passed to the link editor as the last -l operand,  
 18083 causing it to be searched after all other object files and libraries are loaded.

18084 It is unspecified whether the library **libf.a** exists as a regular file. The implementation may  
 18085 accept as -l operands names of objects that do not exist as regular files.

|       |                                                                                                                    |
|-------|--------------------------------------------------------------------------------------------------------------------|
| 18086 | <b>External Symbols</b>                                                                                            |
| 18087 | The FORTRAN compiler and link editor shall support the significance of external symbols up to                      |
| 18088 | a length of at least 31 bytes; case folding is permitted. The action taken upon encountering                       |
| 18089 | symbols exceeding the implementation-defined maximum symbol length is unspecified.                                 |
| 18090 | The compiler and link editor shall support a minimum of 511 external symbols per source or                         |
| 18091 | object file, and a minimum of 4 095 external symbols total. A diagnostic message is written to                     |
| 18092 | standard output if the implementation-defined limit is exceeded; other actions are unspecified.                    |
| 18093 | <b>EXIT STATUS</b>                                                                                                 |
| 18094 | The following exit values shall be returned:                                                                       |
| 18095 | 0 Successful compilation or link edit.                                                                             |
| 18096 | >0 An error occurred.                                                                                              |
| 18097 | <b>CONSEQUENCES OF ERRORS</b>                                                                                      |
| 18098 | When <i>fort77</i> encounters a compilation error, it shall write a diagnostic to standard error and               |
| 18099 | continue to compile other source code operands. It shall return a non-zero exit status, but it is                  |
| 18100 | implementation-defined whether an object module is created. If the link edit is unsuccessful, a                    |
| 18101 | diagnostic message shall be written to standard error, and <i>fort77</i> shall exit with a non-zero                |
| 18102 | status.                                                                                                            |
| 18103 | <b>APPLICATION USAGE</b>                                                                                           |
| 18104 | None.                                                                                                              |
| 18105 | <b>EXAMPLES</b>                                                                                                    |
| 18106 | The following usage example compiles <b>xyz.f</b> and creates the executable file <b>foo</b> :                     |
| 18107 | <code>fort77 -o foo xyz.f</code>                                                                                   |
| 18108 | The following example compiles <b>xyz.f</b> and creates the object file <b>xyz.o</b> :                             |
| 18109 | <code>fort77 -c xyz.f</code>                                                                                       |
| 18110 | The following example compiles <b>xyz.f</b> and creates the executable file <b>a.out</b> :                         |
| 18111 | <code>fort77 xyz.f</code>                                                                                          |
| 18112 | The following example compiles <b>xyz.f</b> , links it with <b>b.o</b> , and creates the executable <b>a.out</b> : |
| 18113 | <code>fort77 xyz.f b.o</code>                                                                                      |
| 18114 | <b>RATIONALE</b>                                                                                                   |
| 18115 | The name of this utility was chosen as <i>fort77</i> to parallel the renaming of the C compiler. The               |
| 18116 | name <i>f77</i> was not chosen to avoid problems with historical implementations. The                              |
| 18117 | ANSI X3.9-1978 standard was selected as a normative reference because the ISO/IEC version of                       |
| 18118 | FORTRAN-77 has been superseded by the ISO/IEC 1539:1990 standard (Fortran-90).                                     |
| 18119 | The file inclusion and symbol definition <b>#define</b> mechanisms used by the <i>c99</i> utility were not         |
| 18120 | included in this volume of IEEE Std 1003.1-2001—even though they are commonly                                      |
| 18121 | implemented—since there is no requirement that the FORTRAN compiler use the C                                      |
| 18122 | preprocessor.                                                                                                      |
| 18123 | The <b>-onetrip</b> option was not included in this volume of IEEE Std 1003.1-2001, even though many               |
| 18124 | historical compilers support it, because it is derived from FORTRAN-66; it is an anachronism                       |
| 18125 | that should not be perpetuated.                                                                                    |
| 18126 | Some implementations produce compilation listings. This aspect of FORTRAN has been left                            |
| 18127 | unspecified because there was controversy concerning the various methods proposed for                              |
| 18128 | implementing it: a <b>-V</b> option overlapped with historical vendor practice and a naming                        |

18129 convention of creating files with .I suffixes collided with historical *lex* file naming practice.

18130 There is no -I option in this version of this volume of IEEE Std 1003.1-2001 to specify a directory  
18131 for file inclusion. An INCLUDE directive has been a part of the Fortran-90 discussions, but an  
18132 interface supporting that standard is not in the current scope.

18133 It is noted that many FORTRAN compilers produce an object module even when compilation  
18134 errors occur; during a subsequent compilation, the compiler may patch the object module rather  
18135 than recompiling all the code. Consequently, it is left to the implementor whether or not an  
18136 object file is created.

18137 A reference to MIL-STD-1753 was removed from an early proposal in response to a request from  
18138 the POSIX FORTRAN-binding standard developers. It was not the intention of the standard  
18139 developers to require certification of the FORTRAN compiler, and IEEE Std 1003.9-1992 does not  
18140 specify the military standard or any special preprocessing requirements. Furthermore, use of  
18141 that document would have been inappropriate for an international standard.

18142 The specification of optimization has been subject to changes through early proposals. At one  
18143 time, -O and -N were Booleans: optimize and do not optimize (with an unspecified default).  
18144 Some historical practice led this to be changed to:

18145 -O 0 No optimization.

18146 -O 1 Some level of optimization.

18147 -O n Other, unspecified levels of optimization.

18148 It is not always clear whether “good code generation” is the same thing as optimization. Simple  
18149 optimizations of local actions do not usually affect the semantics of a program. The -O 0 option  
18150 has been included to accommodate the very particular nature of scientific calculations in a  
18151 highly optimized environment; compilers make errors. Some degree of optimization is expected,  
18152 even if it is not documented here, and the ability to shut it off completely could be important  
18153 when porting an application. An implementation may treat -O 0 as “do less than normal” if it  
18154 wishes, but this is only meaningful if any of the operations it performs can affect the semantics  
18155 of a program. It is highly dependent on the implementation whether doing less than normal is  
18156 logical. It is not the intent of the -O 0 option to ask for inefficient code generation, but rather to  
18157 assure that any semantically visible optimization is suppressed.

18158 The specification of standard library access is consistent with the C compiler specification.  
18159 Implementations are not required to have /usr/lib/libf.a, as many historical implementations do,  
18160 but if not they are required to recognize f as a token.

18161 External symbol size limits are in normative text; conforming applications need to know these  
18162 limits. However, the minimum maximum symbol length should be taken as a constraint on a  
18163 conforming application, not on an implementation, and consequently the action taken for a  
18164 symbol exceeding the limit is unspecified. The minimum size for the external symbol table was  
18165 added for similar reasons.

18166 The CONSEQUENCES OF ERRORS section clearly specifies the behavior of the compiler when  
18167 compilation or link-edit errors occur. The behavior of several historical implementations was  
18168 examined, and the choice was made to be silent on the status of the executable, or a.out, file in  
18169 the face of compiler or linker errors. If a linker writes the executable file, then links it on disk  
18170 with lseek()s and write()s, the partially linked executable file can be left on disk and its execute  
18171 bits turned off if the link edit fails. However, if the linker links the image in memory before  
18172 writing the file to disk, it need not touch the executable file (if it already exists) because the link  
18173 edit fails. Since both approaches are historical practice, a conforming application shall rely on  
18174 the exit status of fort77, rather than on the existence or mode of the executable file.

18175        The **-g** and **-s** options are not specified as mutually-exclusive. Historically these two options  
18176        have been mutually-exclusive, but because both are so loosely specified, it seemed appropriate  
18177        to leave their interaction unspecified.

18178        The requirement that conforming applications specify compiler options separately is to reserve  
18179        the multi-character option name space for vendor-specific compiler options, which are known to  
18180        exist in many historical implementations. Implementations are not required to recognize, for  
18181        example, **-gc** as if it were **-g -c**; nor are they forbidden from doing so. The SYNOPSIS shows all  
18182        of the options separately to highlight this requirement on applications.

18183        Echoing filenames to standard error is considered a diagnostic message because it would  
18184        otherwise be difficult to associate an error message with the erring file. They are described with  
18185        “may” to allow implementations to use other methods of identifying files and to parallel the  
18186        description in *c99*.

#### 18187 FUTURE DIRECTIONS

18188        A compilation system based on the ISO/IEC 1539:1990 standard (Fortran-90) may be considered  
18189        for a future version; it may have a different utility name from *fort77*.

#### 18190 SEE ALSO

18191        *ar*, *asa*, *c99*, *umask*, the System Interfaces volume of IEEE Std 1003.1-2001, *exec*

#### 18192 CHANGE HISTORY

18193        First released in Issue 4.

#### 18194 Issue 6

18195        This utility is marked as part of the FORTRAN Development Utilities option.

18196        The normative text is reworded to avoid use of the term “must” for application requirements.

**18197 NAME**

18198 fuser — list process IDs of all processes that have one or more files open

**18199 SYNOPSIS**

18200 XSI fuser [ -cfu ] file ...

18201

**18202 DESCRIPTION**

18203 The *fuser* utility shall write to standard output the process IDs of processes running on the local  
18204 system that have one or more named files open. For block special devices, all processes using  
18205 any file on that device are listed.

18206 The *fuser* utility shall write to standard error additional information about the named files  
18207 indicating how the file is being used.

18208 Any output for processes running on remote systems that have a named file open is unspecified.

18209 A user may need appropriate privilege to invoke the *fuser* utility.

**18210 OPTIONS**

18211 The *fuser* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section  
18212 12.2, Utility Syntax Guidelines.

18213 The following options shall be supported:

18214 **-c** The file is treated as a mount point and the utility shall report on any files open in  
18215 the file system.

18216 **-f** The report shall be only for the named files.

18217 **-u** The user name, in parentheses, associated with each process ID written to standard  
18218 output shall be written to standard error.

**18219 OPERANDS**

18220 The following operand shall be supported:

18221 *file* A pathname on which the file or file system is to be reported.

**18222 STDIN**

18223 Not used.

**18224 INPUT FILES**

18225 The user database.

**18226 ENVIRONMENT VARIABLES**

18227 The following environment variables shall affect the execution of *fuser*:

18228 **LANG** Provide a default value for the internationalization variables that are unset or null.  
18229 (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2,  
18230 Internationalization Variables for the precedence of internationalization variables  
18231 used to determine the values of locale categories.)

18232 **LC\_ALL** If set to a non-empty string value, override the values of all the other  
18233 internationalization variables.

18234 **LC\_CTYPE** Determine the locale for the interpretation of sequences of bytes of text data as  
18235 characters (for example, single-byte as opposed to multi-byte characters in  
18236 arguments).

**18237 LC\_MESSAGES**

18238 Determine the locale that should be used to affect the format and contents of  
18239 diagnostic messages written to standard error.

18240        **NLSPATH**    Determine the location of message catalogs for the processing of *LC\_MESSAGES*.

18241 **ASYNCHRONOUS EVENTS**

18242        Default.

18243 **STDOUT**

18244        The *fuser* utility shall write the process ID for each process using each file given as an operand to  
18245        standard output in the following format:

18246        "%d", <process\_id>

18247 **STDERR**

18248        The *fuser* utility shall write diagnostic messages to standard error.

18249        The *fuser* utility also shall write the following to standard error:

18250        • The pathname of each named file is written followed immediately by a colon.

18251        • For each process ID written to standard output, the character 'c' shall be written to  
18252        standard error if the process is using the file as its current directory and the character 'r'  
18253        shall be written to standard error if the process is using the file as its root directory.  
18254        Implementations may write other alphabetic characters to indicate other uses of files.

18255        • When the **-u** option is specified, characters indicating the use of the file shall be followed  
18256        immediately by the user name, in parentheses, corresponding to the process' real user ID. If  
18257        the user name cannot be resolved from the process' real user ID, the process' real user ID  
18258        shall be written instead of the user name.

18259        When standard output and standard error are directed to the same file, the output shall be  
18260        interleaved so that the filename appears at the start of each line, followed by the process ID and  
18261        characters indicating the use of the file. Then, if the **-u** option is specified, the user name or user  
18262        ID for each process using that file shall be written.

18263        A <newline> shall be written to standard error after the last output described above for each *file*  
18264        operand.

18265 **OUTPUT FILES**

18266        None.

18267 **EXTENDED DESCRIPTION**

18268        None.

18269 **EXIT STATUS**

18270        The following exit values shall be returned:

18271        0    Successful completion.

18272        >0   An error occurred.

18273 **CONSEQUENCES OF ERRORS**

18274        Default.

**18275 APPLICATION USAGE**

18276 None.

**18277 EXAMPLES**

18278 The command:

18279 `fuser -fu .`

18280 writes to standard output the process IDs of processes that are using the current directory and  
18281 writes to standard error an indication of how those processes are using the directory and the  
18282 user names associated with the processes that are using the current directory.

**18283 RATIONALE**

18284 The definition of the *fuser* utility follows existing practice.

**18285 FUTURE DIRECTIONS**

18286 None.

**18287 SEE ALSO**

18288 None.

**18289 CHANGE HISTORY**

18290 First released in Issue 5.

**18291 NAME**

18292 gencat — generate a formatted message catalog

**18293 SYNOPSIS**

18294 XSI gencat *catfile msgfile...*

18295

**18296 DESCRIPTION**

18297 The *gencat* utility shall merge the message text source file *msgfile* into a formatted message  
18298 catalog *catfile*. The file *catfile* shall be created if it does not already exist. If *catfile* does exist, its  
18299 messages shall be included in the new *catfile*. If set and message numbers collide, the new  
18300 message text defined in *msgfile* shall replace the old message text currently contained in *catfile*.

**18301 OPTIONS**

18302 None.

**18303 OPERANDS**

18304 The following operands shall be supported:

18305 *catfile* A pathname of the formatted message catalog. If ‘–’ is specified, standard output  
18306 shall be used. The format of the message catalog produced is unspecified.

18307 *msgfile* A pathname of a message text source file. If ‘–’ is specified for an instance of  
18308 *msgfile*, standard input shall be used. The format of message text source files is  
18309 defined in the EXTENDED DESCRIPTION section.

**18310 STDIN**

18311 The standard input shall not be used unless a *msgfile* operand is specified as ‘–’.

**18312 INPUT FILES**

18313 The input files shall be text files.

**18314 ENVIRONMENT VARIABLES**

18315 The following environment variables shall affect the execution of *gencat*:

18316 *LANG* Provide a default value for the internationalization variables that are unset or null.  
18317 (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2,  
18318 Internationalization Variables for the precedence of internationalization variables  
18319 used to determine the values of locale categories.)

18320 *LC\_ALL* If set to a non-empty string value, override the values of all the other  
18321 internationalization variables.

18322 *LC\_CTYPE* Determine the locale for the interpretation of sequences of bytes of text data as  
18323 characters (for example, single-byte as opposed to multi-byte characters in  
18324 arguments and input files).

**18325 *LC\_MESSAGES***

18326 Determine the locale that should be used to affect the format and contents of  
18327 diagnostic messages written to standard error.

18328 *NLSPATH* Determine the location of message catalogs for the processing of *LC\_MESSAGES*.

**18329 ASYNCHRONOUS EVENTS**

18330 Default.

**18331 STDOUT**

18332 The standard output shall not be used unless the *catfile* operand is specified as ‘–’.

**18333 STDERR**

18334 The standard error shall be used only for diagnostic messages.

**18335 OUTPUT FILES**

18336 None.

**18337 EXTENDED DESCRIPTION**

18338 The content of a message text file shall be in the format defined as follows. Note that the fields of  
18339 a message text source line are separated by a single <blank>. Any other <blank>s are considered  
18340 to be part of the subsequent field.

**18341 \$set n comment**

18342 This line specifies the set identifier of the following messages until the next \$set or  
18343 end-of-file appears. The *n* denotes the set identifier, which is defined as a number  
18344 in the range [1, {NL\_SETMAX}] (see the <limits.h> header defined in the Base  
18345 Definitions volume of IEEE Std 1003.1-2001). The application shall ensure that set  
18346 identifiers are presented in ascending order within a single source file, but need  
18347 not be contiguous. Any string following the set identifier shall be treated as a  
18348 comment. If no \$set directive is specified in a message text source file, all messages  
18349 shall be located in an implementation-defined default message set NL\_SETD (see  
18350 the <nl\_types.h> header defined in the Base Definitions volume of  
18351 IEEE Std 1003.1-2001).

**18352 \$delset n comment**

18353 This line deletes message set *n* from an existing message catalog. The *n* denotes the  
18354 set number [1, {NL\_SETMAX}]. Any string following the set number shall be  
18355 treated as a comment.

**18356 \$ comment** A line beginning with '\$' followed by a <blank> shall be treated as a comment.**18357 m message-text**

18358 The *m* denotes the message identifier, which is defined as a number in the range [1,  
18359 {NL\_MSGMAX}] (see the <limits.h> header). The *message-text* shall be stored in the  
18360 message catalog with the set identifier specified by the last \$set directive, and with  
18361 message identifier *m*. If the *message-text* is empty, and a <blank> field separator is  
18362 present, an empty string shall be stored in the message catalog. If a message source  
18363 line has a message number, but neither a field separator nor *message-text*, the  
18364 existing message with that number (if any) shall be deleted from the catalog. The  
18365 application shall ensure that message identifiers are in ascending order within a  
18366 single set, but need not be contiguous. The application shall ensure that the length  
18367 of *message-text* is in the range [0, {NL\_TEXTMAX}] (see the <limits.h> header).

**18368 \$quote n**

18369 This line specifies an optional quote character *c*, which can be used to surround  
18370 *message-text* so that trailing spaces or null (empty) messages are visible in a  
18371 message source line. By default, or if an empty \$quote directive is supplied, no  
quoting of *message-text* shall be recognized.

18372 Empty lines in a message text source file shall be ignored. The effects of lines starting with any  
18373 character other than those defined above are implementation-defined.

18374 Text strings can contain the special characters and escape sequences defined in the following  
18375 table:

18376  
18377  
18378  
18379  
18380  
18381  
18382  
18383  
18384  
18385

| Description       | Symbol | Sequence |
|-------------------|--------|----------|
| <newline>         | NL(LF) | \n       |
| Horizontal-tab    | HT     | \t       |
| <vertical-tab>    | VT     | \v       |
| <backspace>       | BS     | \b       |
| <carriage-return> | CR     | \r       |
| <form-feed>       | FF     | \f       |
| Backslash         | \      | \\       |
| Bit pattern       | ddd    | \ddd     |

18386 The escape sequence "\ddd" consists of backslash followed by one, two, or three octal digits,  
18387 which shall be taken to specify the value of the desired character. If the character following a  
18388 backslash is not one of those specified, the backslash shall be ignored.

18389 Backslash ('\'') followed by a <newline> is also used to continue a string on the following line.  
18390 Thus, the following two lines describe a single message string:

18391 1 This line continues \  
18392 to the next line

18393 which shall be equivalent to:

18394 1 This line continues to the next line

#### 18395 EXIT STATUS

18396 The following exit values shall be returned:

18397 0 Successful completion.  
18398 >0 An error occurred.

#### 18399 CONSEQUENCES OF ERRORS

18400 Default.

#### 18401 APPLICATION USAGE

18402 Message catalogs produced by *gencat* are binary encoded, meaning that their portability cannot  
18403 be guaranteed between different types of machine. Thus, just as C programs need to be  
18404 recompiled for each type of machine, so message catalogs must be recreated via *gencat*.

#### 18405 EXAMPLES

18406 None.

#### 18407 RATIONALE

18408 None.

#### 18409 FUTURE DIRECTIONS

18410 None.

#### 18411 SEE ALSO

18412 *iconv*, the Base Definitions volume of IEEE Std 1003.1-2001, <limits.h>, <nl\_types.h>

#### 18413 CHANGE HISTORY

18414 First released in Issue 3.

#### 18415 Issue 6

18416 The normative text is reworded to avoid use of the term “must” for application requirements.

## 18417 NAME

18418 get — get a version of an SCCS file (**DEVELOPMENT**)

## 18419 SYNOPSIS

18420 XSI get [-begkmnlLpst][-c cutoff][-i list][-r SID][-x list] file...

18421

## 18422 DESCRIPTION

18423 The *get* utility shall generate a text file from each named SCCS *file* according to the specifications  
18424 given by its options.18425 The generated text shall normally be written into a file called the **g-file** whose name is derived  
18426 from the SCCS filename by simply removing the leading "s.". 

## 18427 OPTIONS

18428 The *get* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section 12.2,  
18429 Utility Syntax Guidelines.

18430 The following options shall be supported:

18431 **-r SID** Indicate the SCCS Identification String (SID) of the version (delta) of an SCCS file  
18432 to be retrieved. The table shows, for the most useful cases, what version of an  
18433 SCCS file is retrieved (as well as the SID of the version to be eventually created by  
18434 *delta* if the **-e** option is also used), as a function of the SID specified.18435 **-c cutoff** Indicate the *cutoff* date-time, in the form:

18436 YY[MM[DD[HH[MM[SS]]]]]

18437 For the YY component, values in the range [69,99] shall refer to years 1969 to 1999  
18438 inclusive, and values in the range [00,68] shall refer to years 2000 to 2068 inclusive.18439 **Note:** It is expected that in a future version of IEEE Std 1003.1-2001 the default  
18440 century inferred from a 2-digit year will change. (This would apply to all  
18441 commands accepting a 2-digit year as input.)18442 No changes (deltas) to the SCCS file that were created after the specified *cutoff*  
18443 date-time shall be included in the generated text file. Units omitted from the date-  
18444 time default to their maximum possible values; for example, **-c 7502** is equivalent  
18445 to **-c 750228235959**.18446 Any number of non-numeric characters may separate the various 2-digit pieces of  
18447 the *cutoff* date-time. This feature allows the user to specify a *cutoff* date in the form:  
18448 **-c "77/2/2 9:22:25"**.18449 **-e** Indicate that the *get* is for the purpose of editing or making a change (delta) to the  
18450 SCCS file via a subsequent use of *delta*. The **-e** option used in a *get* for a particular  
18451 version (SID) of the SCCS file shall prevent further *get* commands from editing on  
18452 the same SID until *delta* is executed or the **j** (joint edit) flag is set in the SCCS file.  
18453 Concurrent use of *get -e* for different SIDs is always allowed.18454 If the **g-file** generated by *get* with a **-e** option is accidentally ruined in the process  
18455 of editing, it may be regenerated by re-executing the *get* command with the **-k**  
18456 option in place of the **-e** option.18457 SCCS file protection specified via the ceiling, floor, and authorized user list stored  
18458 in the SCCS file shall be enforced when the **-e** option is used.18459 **-b** Use with the **-e** option to indicate that the new delta should have an SID in a new  
18460 branch as shown in the table below. This option shall be ignored if the **b** flag is not  
18461 present in the file or if the retrieved delta is not a leaf delta. (A leaf delta is one that

- 18462 has no successors on the SCCS file tree.)
- 18463 **Note:** A branch delta may always be created from a non-leaf delta.
- 18464     **-i** *list* Indicate a *list* of deltas to be included (forced to be applied) in the creation of the generated file. The *list* has the following syntax:
- ```
<list> ::= <range> | <list> , <range>
<range> ::= SID | SID - SID
```
- 18465 SID, the SCCS Identification of a delta, may be in any form shown in the “SID Specified” column of the table in the EXTENDED DESCRIPTION section, except that the result of supplying a partial SID is unspecified. A diagnostic message shall be written if the first SID in the range is not an ancestor of the second SID in the range.
- 18466
- 18467
- 18468
- 18469
- 18470
- 18471
- 18472
- 18473 **-x** *list* Indicate a *list* of deltas to be excluded (forced not to be applied) in the creation of the generated file. See the **-i** option for the *list* format.
- 18474
- 18475 **-k** Suppress replacement of identification keywords (see below) in the retrieved text by their value. The **-k** option shall be implied by the **-e** option.
- 18476
- 18477 **-l** Write a delta summary into an **l-file**.
- 18478 **-L** Write a delta summary to standard output. All informative output that normally is written to standard output shall be written to standard error instead, unless the **-s** option is used, in which case it shall be suppressed.
- 18479
- 18480
- 18481 **-p** Write the text retrieved from the SCCS file to the standard output. No **g-file** shall be created. All informative output that normally goes to the standard output shall go to standard error instead, unless the **-s** option is used, in which case it shall disappear.
- 18482
- 18483
- 18484
- 18485 **-s** Suppress all informative output normally written to standard output. However, fatal error messages (which shall always be written to the standard error) shall remain unaffected.
- 18486
- 18487
- 18488 **-m** Precede each text line retrieved from the SCCS file by the SID of the delta that inserted the text line in the SCCS file. The format shall be:
- 18489 "**%s\t%s**", <SID>, <text line>
- 18490
- 18491 **-n** Precede each generated text line with the **%M%** identification keyword value (see below). The format shall be:
- 18492 "**%s\t%s**", <%M% value>, <text line>
- 18493
- 18494 When both the **-m** and **-n** options are used, the <text line> shall be replaced by the **-m** option-generated format.
- 18495
- 18496 **-g** Suppress the actual retrieval of text from the SCCS file. It is primarily used to generate an **l-file**, or to verify the existence of a particular SID.
- 18497
- 18498 **-t** Use to access the most recently created (top) delta in a given release (for example, **-r 1**), or release and level (for example, **-r 1.2**).
- 18499

18500 OPERANDS

- 18501 The following operands shall be supported:
- 18502 *file* A pathname of an existing SCCS file or a directory. If *file* is a directory, the *get* utility shall behave as though each file in the directory were specified as a named file, except that non-SCCS files (last component of the pathname does not begin

18505 with **s**.) and unreadable files shall be silently ignored.

18506 If exactly one *file* operand appears, and it is '**-**', the standard input shall be read;

18507 each line of the standard input is taken to be the name of an SCCS file to be

18508 processed. Non-SCCS files and unreadable files shall be silently ignored.

18509 **STDIN**

18510 The standard input shall be a text file used only if the *file* operand is specified as '**-**'. Each line

18511 of the text file shall be interpreted as an SCCS pathname.

18512 **INPUT FILES**

18513 The SCCS files shall be files of an unspecified format.

18514 **ENVIRONMENT VARIABLES**

18515 The following environment variables shall affect the execution of *get*:

18516 **LANG** Provide a default value for the internationalization variables that are unset or null.
 18517 (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2,
 18518 Internationalization Variables for the precedence of internationalization variables
 18519 used to determine the values of locale categories.)

18520 **LC_ALL** If set to a non-empty string value, override the values of all the other
 18521 internationalization variables.

18522 **LC_CTYPE** Determine the locale for the interpretation of sequences of bytes of text data as
 18523 characters (for example, single-byte as opposed to multi-byte characters in
 18524 arguments and input files).

18525 **LC_MESSAGES**

18526 Determine the locale that should be used to affect the format and contents of
 18527 diagnostic messages written to standard error, and informative messages written
 18528 to standard output (or standard error, if the **-p** option is used).

18529 **NLSPATH** Determine the location of message catalogs for the processing of **LC_MESSAGES**.

18530 **TZ** Determine the timezone in which the times and dates written in the SCCS file are
 18531 evaluated. If the **TZ** variable is unset or NULL, an unspecified system default
 18532 timezone is used.

18533 **ASYNCHRONOUS EVENTS**

18534 Default.

18535 **STDOUT**

18536 For each file processed, *get* shall write to standard output the SID being accessed and the number
 18537 of lines retrieved from the SCCS file, in the following format:

18538 "**%s\n%d** lines\n", <SID>, <number of lines>

18539 If the **-e** option is used, the SID of the delta to be made shall appear after the SID accessed and
 18540 before the number of lines generated, in the POSIX locale:

18541 "**%s\nnew delta %s\n%d** lines\n", <SID accessed>,
 18542 <SID to be made>, <number of lines>

18543 If there is more than one named file or if a directory or standard input is named, each pathname
 18544 shall be written before each of the lines shown in one of the preceding formats:

18545 "**\n%s:\n**", <pathname>

18546 If the **-L** option is used, a delta summary shall be written following the format specified below
 18547 for **l-files**.

18548 If the **-i** option is used, included deltas shall be listed following the notation, in the POSIX locale:
18549 "Included:\n"
18550 If the **-x** option is used, excluded deltas shall be listed following the notation, in the POSIX
18551 locale:
18552 "Excluded:\n"
18553 If the **-p** or **-L** options are specified, the standard output shall consist of the text retrieved from
18554 the SCCS file.

18555 **STDERR**

18556 The standard error shall be used only for diagnostic messages, except if the **-p** or **-L** options are
18557 specified, it shall include all informative messages normally sent to standard output.

18558 **OUTPUT FILES**

18559 Several auxiliary files may be created by *get*. These files are known generically as the **g-file**, **l-**
18560 **file**, **p-file**, and **z-file**. The letter before the hyphen is called the *tag*. An auxiliary filename shall
18561 be formed from the SCCS filename: the application shall ensure that the last component of all
18562 SCCS filenames is of the form **s.module-name**; the auxiliary files shall be named by replacing the
18563 leading **s** with the tag. The **g-file** shall be an exception to this scheme: the **g-file** is named by
18564 removing the **s**. prefix. For example, for **s.xyz.c**, the auxiliary filenames would be **xyz.c**, **l.xyz.c**,
18565 **p.xyz.c**, and **z.xyz.c**, respectively.

18566 The **g-file**, which contains the generated text, shall be created in the current directory (unless the
18567 **-p** option is used). A **g-file** shall be created in all cases, whether or not any lines of text were
18568 generated by the *get*. It shall be owned by the real user. If the **-k** option is used or implied, the
18569 **g-file** shall be writable by the owner only (read-only for everyone else); otherwise, it shall be
18570 read-only. Only the real user need have write permission in the current directory.

18571 The **l-file** shall contain a table showing which deltas were applied in generating the retrieved
18572 text. The **l-file** shall be created in the current directory if the **-l** option is used; it shall be read-
18573 only and it is owned by the real user. Only the real user need have write permission in the
18574 current directory.

18575 Lines in the **l-file** shall have the following format:

18576 "%c%c%cΔ%s\t%sΔ%s\n", <code1>, <code2>, <code3>,
18577 <SID>, <date-time>, <login>

18578 where the entries are:

18579 <code1> A <space> if the delta was applied; '*' otherwise.
18580 <code2> A <space> if the delta was applied or was not applied and ignored; '*' if the delta
18581 was not applied and was not ignored.
18582 <code3> A character indicating a special reason why the delta was or was not applied:
18583 **I** Included.
18584 **X** Excluded.
18585 **C** Cut off (by a **-c** option).
18586 <date-time> Date and time (using the format of the *date* utility's %y/%m/%d %T conversion
18587 specification format) of creation.
18588 <login> Login name of person who created *delta*.

18589 The comments and MR data shall follow on subsequent lines, indented one <tab>. A blank line
18590 shall terminate each entry.

18591 The **p-file** shall be used to pass information resulting from a *get* with a **-e** option along to *delta*.
18592 Its contents shall also be used to prevent a subsequent execution of *get* with a **-e** option for the
18593 same SID until *delta* is executed or the joint edit flag, **j**, is set in the SCCS file. The **p-file** shall be
18594 created in the directory containing the SCCS file and the application shall ensure that the
18595 effective user has write permission in that directory. It shall be writable by owner only, and
18596 owned by the effective user. Each line in the **p-file** shall have the following format:

18597 "%sΔ%sΔ%sΔ%s%s\n", <*g-file SID*>,
18598 <*SID of new delta*>, <*login-name of real user*>,
18599 <*date-time*>, <*i-value*>, <*x-value*>

18600 where <*i-value*> uses the format " " if no **-i** option was specified, and shall use the format:

18601 "&Δ-i%s", <-i option *option-argument*>

18602 if a **-i** option was specified and <*x-value*> uses the format " " if no **-x** option was specified, and
18603 shall use the format:

18604 "&Δ-x%s", <-x option *option-argument*>

18605 if a **-x** option was specified. There can be an arbitrary number of lines in the **p-file** at any time;
18606 no two lines shall have the same new delta SID.

18607 The **z-file** shall serve as a lock-out mechanism against simultaneous updates. Its contents shall
18608 be the binary process ID of the command (that is, *get*) that created it. The **z-file** shall be created
18609 in the directory containing the SCCS file for the duration of *get*. The same protection restrictions
18610 as those for the **p-file** shall apply for the **z-file**. The **z-file** shall be created read-only.

18611 EXTENDED DESCRIPTION

Determination of SCCS Identification String					
18612	SID* Specified	-b Keyletter Used†	Other Conditions	SID Retrieved	SID of Delta to be Created
18613	none‡	no	R defaults to mR	mR.mL	mR.(mL+1)
18614	none‡	yes	R defaults to mR	mR.mL	mR.mL.(mB+1).1
18615	R	no	R > mR	mR.mL	R.1***
18616	R	no	R = mR	mR.mL	mR.(mL+1)
18617	R	yes	R > mR	mR.mL	mR.mL.(mB+1).1
18618	R	yes	R = mR	mR.mL	mR.mL.(mB+1).1
18619	R	—	R < mR and R does not exist	hR.mL**	hR.mL.(mB+1).1
18620	R	—	Trunk successor in release > R and R exists	R.mL	R.mL.(mB+1).1
18621	R.L	no	No trunk successor	R.L	R.(L+1)
18622	R.L	yes	No trunk successor	R.L	R.L.(mB+1).1
18623	R.L	—	Trunk successor in release ≥ R	R.L	R.L.(mB+1).1
18624	R.L.B	no	No branch successor	R.L.B.mS	R.L.B.(mS+1)
18625	R.L.B	yes	No branch successor	R.L.B.mS	R.L.(mB+1).1
18626	R.L.B.S	no	No branch successor	R.L.B.S	R.L.B.(S+1)
18627	R.L.B.S	yes	No branch successor	R.L.B.S	R.L.(mB+1).1
18628	R.L.B.S	—	Branch successor	R.L.B.S	R.L.(mB+1).1

* R, L, B, and S are the release, level, branch, and sequence components of the SID, respectively; m means maximum. Thus, for example, R.mL means “the maximum level number within release R”; R.L.(mB+1).1 means “the first sequence number on the new branch (that is, maximum branch number plus one) of level L within release R”. Note that if the SID specified is of the form R.L, R.L.B, or R.L.B.S, each of the specified components shall exist.

** hR is the highest existing release that is lower than the specified, nonexistent, release R.

*** This is used to force creation of the first delta in a new release.

† The –b option is effective only if the b flag is present in the file. An entry of ‘—’ means “irrelevant”.

‡ This case applies if the d (default SID) flag is not present in the file. If the d flag is present in the file, then the SID obtained from the d flag is interpreted as if it had been specified on the command line. Thus, one of the other cases in this table applies.

18647 **System Date and Time**

18648 When a **g-file** is generated, the creation time of deltas in the SCCS file may be taken into
18649 account. If any of these times are apparently in the future, the behavior is unspecified.

18650 **Identification Keywords**

18651 Identifying information shall be inserted into the text retrieved from the SCCS file by replacing
18652 identification keywords with their value wherever they occur. The following keywords may be
18653 used in the text stored in an SCCS file:

18654 %M% Module name: either the value of the **m** flag in the file, or if absent, the name of the
18655 SCCS file with the leading **s**. removed.

18656 %I% SCCS identification (SID) (%R%.%L% or %R%.%L%.%B%.%S%) of the retrieved
18657 text.

18658 %R% Release.

18659 %L% Level.

18660 %B% Branch.

18661 %S% Sequence.

18662 %D% Current date (YY/MM/DD).

18663 %H% Current date (MM/DD/YY).

18664 %T% Current time (HH:MM:SS).

18665 %E% Date newest applied delta was created (YY/MM/DD).

18666 %G% Date newest applied delta was created (MM/DD/YY).

18667 %U% Time newest applied delta was created (HH:MM:SS).

18668 %Y% Module type: value of the **t** flag in the SCCS file.

18669 %F% SCCS filename.

18670 %P% SCCS absolute pathname.

18671 %Q% The value of the **q** flag in the file.

18672 %C% Current line number. This keyword is intended for identifying messages output by
18673 the program, such as "this should not have happened" type errors. It is not
18674 intended to be used on every line to provide sequence numbers.

18675 %Z% The four-character string "@(#)" recognizable by *what*.

18676 %W% A shorthand notation for constructing *what* strings:

18677 % W % = % Z % % M % <tab> % I %

18678 %A% Another shorthand notation for constructing *what* strings:

18679 % A % = % Z % % Y % % M % % I % % Z %

18680 **EXIT STATUS**

18681 The following exit values shall be returned:

18682 0 Successful completion.

18683 >0 An error occurred.

18684 CONSEQUENCES OF ERRORS

18685 Default.

18686 APPLICATION USAGE

18687 Problems can arise if the system date and time have been modified (for example, put forward
18688 and then back again, or unsynchronized clocks across a network) and can also arise when
18689 different values of the *TZ* environment variable are used.

18690 Problems of a similar nature can also arise for the operation of the *delta* utility, which compares
18691 the previous file body against the working file as part of its normal operation.

18692 EXAMPLES

18693 None.

18694 RATIONALE

18695 None.

18696 FUTURE DIRECTIONS

18697 The *-lp* option may be withdrawn in a future version.

18698 SEE ALSO

18699 *admin, delta, prs, what*

18700 CHANGE HISTORY

18701 First released in Issue 2.

18702 Issue 5

18703 A correction is made to the first format string in STDOUT.

18704 The interpretation of the YY component of the *-c cutoff* argument is noted.

18705 Issue 6

18706 The obsolescent SYNOPSIS is removed, removing the *-lp* option.

18707 The normative text is reworded to avoid use of the term “must” for application requirements.

18708 The Open Group Corrigendum U025/5 is applied, correcting text in the OPTIONS section.

18709 The Open Group Corrigendum U048/1 is applied.

18710 The Open Group Interpretation PIN4C.00014 is applied.

18711 The Open Group Base Resolution bwg2001-007 is applied as follows:

- The EXTENDED DESCRIPTION section is updated to make partial SID handling unspecified, reflecting common usage, and to clarify SID ranges.

- New text is added to the EXTENDED DESCRIPTION and APPLICATION USAGE sections regarding how the system date and time may be taken into account.

- The *TZ* environment variable is added to the ENVIRONMENT VARIABLES section.

18717 NAME

18718 getconf — get configuration values

18719 SYNOPSIS

18720 getconf [-v specification] system_var

18721 getconf [-v specification] path_var pathname

18722 DESCRIPTION

18723 In the first synopsis form, the *getconf* utility shall write to the standard output the value of the
18724 variable specified by the *system_var* operand.

18725 In the second synopsis form, the *getconf* utility shall write to the standard output the value of the
18726 variable specified by the *path_var* operand for the path specified by the *pathname* operand.

18727 The value of each configuration variable shall be determined as if it were obtained by calling the
18728 function from which it is defined to be available by this volume of IEEE Std 1003.1-2001 or by the
18729 System Interfaces volume of IEEE Std 1003.1-2001 (see the OPERANDS section). The value shall
18730 reflect conditions in the current operating environment.

18731 OPTIONS

18732 The *getconf* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section
18733 12.2, Utility Syntax Guidelines.

18734 The following option shall be supported:

18735 **-v specification**

18736 Indicate a specific specification and version for which configuration variables shall
18737 be determined. If this option is not specified, the values returned correspond to an
18738 implementation default conforming compilation environment.

18739 If the command:

18740 getconf _POSIX_V6_ILP32_OFF32

18741 does not write "-1\n" or "undefined\n" to standard output, then commands of
18742 the form:

18743 getconf -v POSIX_V6_ILP32_OFF32 ...

18744 determine values for configuration variables corresponding to the
18745 POSIX_V6_ILP32_OFF32 compilation environment specified in *c99*, the
18746 EXTENDED DESCRIPTION.

18747 If the command:

18748 getconf _POSIX_V6_ILP32_OFFBIG

18749 does not write "-1\n" or "undefined\n" to standard output, then commands of
18750 the form:

18751 getconf -v POSIX_V6_ILP32_OFFBIG ...

18752 determine values for configuration variables corresponding to the
18753 POSIX_V6_ILP32_OFFBIG compilation environment specified in *c99*, the
18754 EXTENDED DESCRIPTION.

18755 If the command:

18756 getconf _POSIX_V6_LP64_OFF64

18757 does not write "-1\n" or "undefined\n" to standard output, then commands of
18758 the form:

18759 getconf -v POSIX_V6_LP64_OFF64 ...
 18760 determine values for configuration variables corresponding to the
 18761 POSIX_V6_LP64_OFF64 compilation environment specified in *c99*, the
 18762 EXTENDED DESCRIPTION.
 18763 If the command:
 18764 getconf _POSIX_V6_LPBIG_OFFBIG
 18765 does not write "-1\n" or "undefined\n" to standard output, then commands of
 18766 the form:
 18767 getconf -v POSIX_V6_LPBIG_OFFBIG ...
 18768 determine values for configuration variables corresponding to the
 18769 POSIX_V6_LPBIG_OFFBIG compilation environment specified in *c99*, the
 18770 EXTENDED DESCRIPTION.

18771 OPERANDS

18772 The following operands shall be supported:

18773 <i>path_var</i>	A name of a configuration variable. All of the variables in the Variable column of the table in the DESCRIPTION of the <i>fpathconf()</i> function defined in the System Interfaces volume of IEEE Std 1003.1-2001, without the enclosing braces, shall be supported. The implementation may add other local variables.	1
18777 <i>pathname</i>	A pathname for which the variable specified by <i>path_var</i> is to be determined.	1
18778 <i>system_var</i>	A name of a configuration variable. All of the following variables shall be supported: <ul style="list-style-type: none"> • The names in the Variable column of the table in the DESCRIPTION of the <i>sysconf()</i> function in the System Interfaces volume of IEEE Std 1003.1-2001, except for the entries corresponding to <i>_SC_CLK_TCK</i>, <i>_SC_GETGR_R_SIZE_MAX</i>, and <i>_SC_GETPW_R_SIZE_MAX</i>, without the enclosing braces. For compatibility with earlier versions, the following variable names shall also be supported: <ul style="list-style-type: none"> POSIX2_C_BIND POSIX2_C_DEV POSIX2_CHAR_TERM POSIX2_FORT_DEV POSIX2_FORT_RUN POSIX2_LOCALEDEF POSIX2_SW_DEV POSIX2_UPE POSIX2_VERSION 	1
18796 and shall be equivalent to the same name prefixed with an underscore. This requirement may be removed in a future version.	1	
18798 • The names of the symbolic constants used as the <i>name</i> argument of the <i>confstr()</i> function in the System Interfaces volume of IEEE Std 1003.1-2001, without the <i>_CS_</i> prefix.	1	
18801 • The names of the symbolic constants listed under the headings "Maximum Values" and "Minimum Values" in the description of the <limits.h> header in	1	

18803	the Base Definitions volume of IEEE Std 1003.1-2001, without the enclosing braces.	1
18804		1
18805	For compatibility with earlier versions, the following variable names shall also be supported:	1
18806		1
18807	POSIX2_BC_BASE_MAX	1
18808	POSIX2_BC_DIM_MAX	1
18809	POSIX2_BC_SCALE_MAX	1
18810	POSIX2_BC_STRING_MAX	1
18811	POSIX2_COLL_WEIGHTS_MAX	1
18812	POSIX2_EXPR_NEST_MAX	1
18813	POSIX2_LINE_MAX	1
18814	POSIX2_RE_DUP_MAX	1
18815	and shall be equivalent to the same name prefixed with an underscore. This requirement may be removed in a future version.	1
18816		1
18817	The implementation may add other local values.	1
18818	STDIN	
18819	Not used.	
18820	INPUT FILES	
18821	None.	
18822	ENVIRONMENT VARIABLES	
18823	The following environment variables shall affect the execution of <i>getconf</i> .	
18824	<i>LANG</i>	Provide a default value for the internationalization variables that are unset or null. (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2, Internationalization Variables for the precedence of internationalization variables used to determine the values of locale categories.)
18825		
18826		
18827		
18828	<i>LC_ALL</i>	If set to a non-empty string value, override the values of all the other internationalization variables.
18829		
18830	<i>LC_CTYPE</i>	Determine the locale for the interpretation of sequences of bytes of text data as characters (for example, single-byte as opposed to multi-byte characters in arguments).
18831		
18832		
18833	<i>LC_MESSAGES</i>	
18834	Determine the locale that should be used to affect the format and contents of diagnostic messages written to standard error.	
18835		
18836	<i>XSI NLSPATH</i>	Determine the location of message catalogs for the processing of <i>LC_MESSAGES</i> .
18837	ASYNCHRONOUS EVENTS	
18838	Default.	
18839	STDOUT	
18840	If the specified variable is defined on the system and its value is described to be available from the <i>confstr()</i> function defined in the System Interfaces volume of IEEE Std 1003.1-2001, its value shall be written in the following format:	
18841		
18842		
18843	"%s\n", <value>	
18844	Otherwise, if the specified variable is defined on the system, its value shall be written in the following format:	
18845		

18846 "%d\n" , <value>
18847 If the specified variable is valid, but is undefined on the system, *getconf* shall write using the
18848 following format:
18849 "undefined\n"
18850 If the variable name is invalid or an error occurs, nothing shall be written to standard output.

18851 **STDERR**

18852 The standard error shall be used only for diagnostic messages.

18853 **OUTPUT FILES**

18854 None.

18855 **EXTENDED DESCRIPTION**

18856 None.

18857 **EXIT STATUS**

18858 The following exit values shall be returned:

18859 0 The specified variable is valid and information about its current state was written
18860 successfully.
18861 >0 An error occurred.

18862 **CONSEQUENCES OF ERRORS**

18863 Default.

18864 **APPLICATION USAGE**

18865 None.

18866 **EXAMPLES**

18867 The following example illustrates the value of {NGROUPS_MAX}:

18868 getconf NGROUPS_MAX

18869 The following example illustrates the value of {NAME_MAX} for a specific directory:

18870 getconf NAME_MAX /usr

18871 The following example shows how to deal more carefully with results that might be unspecified:

18872 if value=\$(getconf PATH_MAX /usr); then
18873 if ["\$value" = "undefined"]; then
18874 echo PATH_MAX in /usr is infinite.
18875 else
18876 echo PATH_MAX in /usr is \$value.
18877 fi
18878 else
18879 echo Error in getconf.
18880 fi

18881 Note that:

18882 sysconf(_SC_POSIX_C_BIND);

18883 and:

18884 system("getconf POSIX2_C_BIND");

18885 in a C program could give different answers. The *sysconf()* call supplies a value that corresponds
18886 to the conditions when the program was either compiled or executed, depending on the

18887 implementation; the *system()* call to *getconf* always supplies a value corresponding to conditions
18888 when the program is executed.

18889 RATIONALE

18890 The original need for this utility, and for the *confstr()* function, was to provide a way of finding
18891 the configuration-defined default value for the *PATH* environment variable. Since *PATH* can be
18892 modified by the user to include directories that could contain utilities replacing the standard
18893 utilities, shell scripts need a way to determine the system-supplied *PATH* environment variable
18894 value that contains the correct search path for the standard utilities. It was later suggested that
18895 access to the other variables described in this volume of IEEE Std 1003.1-2001 could also be
18896 useful to applications.

18897 This functionality of *getconf* would not be adequately subsumed by another command such as:

18898 `grep var /etc/conf`

18899 because such a strategy would provide correct values for neither those variables that can vary at
18900 runtime, nor those that can vary depending on the path.

18901 Early proposal versions of *getconf* specified exit status 1 when the specified variable was valid,
18902 but not defined on the system. The output string "undefined" is now used to specify this case
18903 with exit code 0 because so many things depend on an exit code of zero when an invoked utility
18904 is successful.

18905 FUTURE DIRECTIONS

18906 None.

18907 SEE ALSO

18908 *c99*, the Base Definitions volume of IEEE Std 1003.1-2001, <limits.h>, the System Interfaces 1
18909 volume of IEEE Std 1003.1-2001, *confstr()*, *pathconf()*, *sysconf()*, *system()*

18910 CHANGE HISTORY

18911 First released in Issue 4.

18912 Issue 5

18913 In the OPERANDS section:

- 18914 • {NL_MAX} is changed to {NL_NMAX}.
- 18915 • Entries beginning NL_ are deleted from the list of standard configuration variables.
- 18916 • The list of variables previously marked UX is merged with the list marked EX.
- 18917 • Operands are added to support new Option Groups.
- 18918 • Operands are added so that *getconf* can determine supported programming environments.

18919 Issue 6

18920 The Open Group Corrigendum U029/4 is applied, correcting the example command in the last
18921 paragraph of the OPTIONS section.

18922 The following new requirements on POSIX implementations derive from alignment with the
18923 Single UNIX Specification:

- 18924 • Operands are added to determine supported programming environments.

18925 This reference page is updated for alignment with the ISO/IEC 9899:1999 standard. Specifically,
18926 new macros for *c99* programming environments are introduced.

18927 XSI marked *system_var* (XBS5_*) values are marked LEGACY.

18928 IEEE Std 1003.1-2001/Cor 1-2002, item XCU/TC1/D6/27 is applied, correcting the descriptions 1
18929 of *path_var* and *system_var* in the OPERANDS section. 1

18930 NAME

18931 getopts — parse utility options

18932 SYNOPSIS

18933 getopts *optstring name [arg...]*

18934 DESCRIPTION

18935 The *getopts* utility shall retrieve options and option-arguments from a list of parameters. It shall
18936 support the Utility Syntax Guidelines 3 to 10, inclusive, described in the Base Definitions volume
18937 of IEEE Std 1003.1-2001, Section 12.2, Utility Syntax Guidelines.

18938 Each time it is invoked, the *getopts* utility shall place the value of the next option in the shell
18939 variable specified by the *name* operand and the index of the next argument to be processed in the
18940 shell variable *OPTIND*. Whenever the shell is invoked, *OPTIND* shall be initialized to 1.

18941 When the option requires an option-argument, the *getopts* utility shall place it in the shell
18942 variable *OPTARG*. If no option was found, or if the option that was found does not have an
18943 option-argument, *OPTARG* shall be unset.

18944 If an option character not contained in the *optstring* operand is found where an option character
18945 is expected, the shell variable specified by *name* shall be set to the question-mark ('?') character.
18946 In this case, if the first character in *optstring* is a colon (':'), the shell variable *OPTARG* shall be
18947 set to the option character found, but no output shall be written to standard error; otherwise, the
18948 shell variable *OPTARG* shall be unset and a diagnostic message shall be written to standard
18949 error. This condition shall be considered to be an error detected in the way arguments were
18950 presented to the invoking application, but shall not be an error in *getopts* processing.

18951 If an option-argument is missing:

- If the first character of *optstring* is a colon, the shell variable specified by *name* shall be set to
the colon character and the shell variable *OPTARG* shall be set to the option character found.
- Otherwise, the shell variable specified by *name* shall be set to the question-mark character,
the shell variable *OPTARG* shall be unset, and a diagnostic message shall be written to
standard error. This condition shall be considered to be an error detected in the way
arguments were presented to the invoking application, but shall not be an error in *getopts*
processing; a diagnostic message shall be written as stated, but the exit status shall be zero.

18952 When the end of options is encountered, the *getopts* utility shall exit with a return value greater
18953 than zero; the shell variable *OPTIND* shall be set to the index of the first non-option-argument,
18954 where the first "--" argument is considered to be an option-argument if there are no other
18955 non-option-arguments appearing before it, or the value "\$#+1 if there are no non-option-
18956 arguments; the *name* variable shall be set to the question-mark character. Any of the following
18957 shall identify the end of options: the special option "--", finding an argument that does not
18958 begin with a '--', or encountering an error.

18959 The shell variables *OPTIND* and *OPTARG* shall be local to the caller of *getopts* and shall not be
18960 exported by default.

18961 The shell variable specified by the *name* operand, *OPTIND*, and *OPTARG* shall affect the current
18962 shell execution environment; see Section 2.12 (on page 61).

18963 If the application sets *OPTIND* to the value 1, a new set of parameters can be used: either the
18964 current positional parameters or new *arg* values. Any other attempt to invoke *getopts* multiple
18965 times in a single shell execution environment with parameters (positional parameters or *arg*
18966 operands) that are not the same in all invocations, or with an *OPTIND* value modified to be a
18967 value other than 1, produces unspecified results.

18975 **OPTIONS**

18976 None.

18977 **OPERANDS**

18978 The following operands shall be supported:

- 18979 *optstring* A string containing the option characters recognized by the utility invoking *getopts*.
 18980 If a character is followed by a colon, the option shall be expected to have an
 18981 argument, which should be supplied as a separate argument. Applications should
 18982 specify an option character and its option-argument as separate arguments, but
 18983 *getopts* shall interpret the characters following an option character requiring
 18984 arguments as an argument whether or not this is done. An explicit null option-
 18985 argument need not be recognized if it is not supplied as a separate argument when
 18986 *getopts* is invoked. (See also the *getopt()* function defined in the System Interfaces
 18987 volume of IEEE Std 1003.1-2001.) The characters question-mark and colon shall not
 18988 be used as option characters by an application. The use of other option characters
 18989 that are not alphanumeric produces unspecified results. If the option-argument is
 18990 not supplied as a separate argument from the option character, the value in
 18991 *OPTARG* shall be stripped of the option character and the '-'. The first character
 18992 in *optstring* determines how *getopts* behaves if an option character is not known or
 18993 an option-argument is missing.
- 18994 *name* The name of a shell variable that shall be set by the *getopts* utility to the option
 18995 character that was found.
- 18996 The *getopts* utility by default shall parse positional parameters passed to the invoking shell
 18997 procedure. If *args* are given, they shall be parsed instead of the positional parameters.

18998 **STDIN**

18999 Not used.

19000 **INPUT FILES**

19001 None.

19002 **ENVIRONMENT VARIABLES**19003 The following environment variables shall affect the execution of *getopts*:

- 19004 *LANG* Provide a default value for the internationalization variables that are unset or null.
 19005 (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2,
 19006 Internationalization Variables for the precedence of internationalization variables
 19007 used to determine the values of locale categories.)
- 19008 *LC_ALL* If set to a non-empty string value, override the values of all the other
 19009 internationalization variables.
- 19010 *LC_CTYPE* Determine the locale for the interpretation of sequences of bytes of text data as
 19011 characters (for example, single-byte as opposed to multi-byte characters in
 19012 arguments and input files).
- 19013 *LC_MESSAGES* Determine the locale that should be used to affect the format and contents of
 19014 diagnostic messages written to standard error.
- 19016 XSI *NLSPATH* Determine the location of message catalogs for the processing of *LC_MESSAGES*.
- 19017 *OPTIND* This variable shall be used by the *getopts* utility as the index of the next argument
 19018 to be processed.

19019 ASYNCHRONOUS EVENTS

19020 Default.

19021 STDOUT

19022 Not used.

19023 STDERR

19024 Whenever an error is detected and the first character in the *optstring* operand is not a colon (' : '), a diagnostic message shall be written to standard error with the following information in 19025 an unspecified format:

- 19027 • The invoking program name shall be identified in the message. The invoking program name 19028 shall be the value of the shell special parameter 0 (see Section 2.5.2 (on page 34)) at the time 19029 the *getopts* utility is invoked. A name equivalent to:

19030 basename "\$0"

19031 may be used.

- 19032 • If an option is found that was not specified in *optstring*, this error is identified and the invalid 19033 option character shall be identified in the message.
- 19034 • If an option requiring an option-argument is found, but an option-argument is not found, 19035 this error shall be identified and the invalid option character shall be identified in the 19036 message.

19037 OUTPUT FILES

19038 None.

19039 EXTENDED DESCRIPTION

19040 None.

19041 EXIT STATUS

19042 The following exit values shall be returned:

19043 0 An option, specified or unspecified by *optstring*, was found.

19044 >0 The end of options was encountered or an error occurred.

19045 CONSEQUENCES OF ERRORS

19046 Default.

19047 APPLICATION USAGE

19048 Since *getopts* affects the current shell execution environment, it is generally provided as a shell 19049 regular built-in. If it is called in a subshell or separate utility execution environment, such as one 19050 of the following:

19051 (getopts abc value "\$@")
19052 nohup getopts ...
19053 find . -exec getopts ... \;

19054 it does not affect the shell variables in the caller's environment.

19055 Note that shell functions share *OPTIND* with the calling shell even though the positional 19056 parameters are changed. If the calling shell and any of its functions uses *getopts* to parse 19057 arguments, the results are unspecified.

19058 EXAMPLES

19059 The following example script parses and displays its arguments:

19060 aflag=
19061 bflag=

```

19062     while getopts ab: name
19063     do
19064         case $name in
19065             a)      aflag=1;;
19066             b)      bflag=1
19067                 bval="$OPTARG";;
19068         ?)      printf "Usage: %s: [-a] [-b value] args\n" $0
19069                 exit 2;;
19070         esac
19071     done
19072     if [ ! -z "$aflag" ]; then
19073         printf "Option -a specified\n"
19074     fi
19075     if [ ! -z "$bflag" ]; then
19076         printf 'Option -b "%s" specified\n' "$bval"
19077     fi
19078     shift $((OPTIND - 1))
19079     printf "Remaining arguments are: %s\n" "$*"

```

19080 RATIONALE

19081 The *getopts* utility was chosen in preference to the System V *getopt* utility because *getopts* handles
 19082 option-arguments containing <blank>s.

19083 The *OPTARG* variable is not mentioned in the ENVIRONMENT VARIABLES section because it
 19084 does not affect the execution of *getopts*; it is one of the few “output-only” variables used by the
 19085 standard utilities.

19086 The colon is not allowed as an option character because that is not historical behavior, and it
 19087 violates the Utility Syntax Guidelines. The colon is now specified to behave as in the KornShell
 19088 version of the *getopts* utility; when used as the first character in the *optstring* operand, it disables
 19089 diagnostics concerning missing option-arguments and unexpected option characters. This
 19090 replaces the use of the *OPTERR* variable that was specified in an early proposal.

19091 The formats of the diagnostic messages produced by the *getopts* utility and the *getopt()* function
 19092 are not fully specified because implementations with superior (“friendlier”) formats objected to
 19093 the formats used by some historical implementations. The standard developers considered it
 19094 important that the information in the messages used be uniform between *getopts* and *getopt()*.
 19095 Exact duplication of the messages might not be possible, particularly if a utility is built on
 19096 another system that has a different *getopt()* function, but the messages must have specific
 19097 information included so that the program name, invalid option character, and type of error can
 19098 be distinguished by a user.

19099 Only a rare application program intercepts a *getopts* standard error message and wants to parse
 19100 it. Therefore, implementations are free to choose the most usable messages they can devise. The
 19101 following formats are used by many historical implementations:

```

19102 "%s: illegal option -- %c\n", <program name>, <option character>
19103 "%s: option requires an argument -- %c\n", <program name>, \
19104             <option character>

```

19105 Historical shells with built-in versions of *getopt()* or *getopts* have used different formats,
 19106 frequently not even indicating the option character found in error.

19107 FUTURE DIRECTIONS

19108 None.

19109 SEE ALSO

19110 Section 2.5.2 (on page 34), the System Interfaces volume of IEEE Std 1003.1-2001, *getopt()*

19111 CHANGE HISTORY

19112 First released in Issue 4.

19113 Issue 6

19114 The normative text is reworded to avoid use of the term “must” for application requirements.

19115 **NAME**

19116 grep — search a file for a pattern

19117 **SYNOPSIS**

```
19118     grep [-E| -F][-c| -l| -q][-insvx] -e pattern_list...
19119     [-f pattern_file]...[file...]
19120     grep [-E| -F][-c| -l| -q][-insvx][-e pattern_list]...
19121     -f pattern_file...[file...]
19122     grep [-E| -F][-c| -l| -q][-insvx] pattern_list[file...]
```

19123 **DESCRIPTION**

19124 The *grep* utility shall search the input files, selecting lines matching one or more patterns; the
19125 types of patterns are controlled by the options specified. The patterns are specified by the **-e**
19126 option, **-f** option, or the *pattern_list* operand. The *pattern_list*'s value shall consist of one or more
19127 patterns separated by <newline>s; the *pattern_file*'s contents shall consist of one or more
19128 patterns terminated by <newline>. By default, an input line shall be selected if any pattern,
19129 treated as an entire basic regular expression (BRE) as described in the Base Definitions volume of
19130 IEEE Std 1003.1-2001, Section 9.3, Basic Regular Expressions, matches any part of the line
19131 excluding the terminating <newline>; a null BRE shall match every line. By default, each selected
19132 input line shall be written to the standard output.

19133 Regular expression matching shall be based on text lines. Since a <newline> separates or
19134 terminates patterns (see the **-e** and **-f** options below), regular expressions cannot contain a
19135 <newline>. Similarly, since patterns are matched against individual lines (excluding the
19136 terminating <newline>s) of the input, there is no way for a pattern to match a <newline> found
19137 in the input.

19138 **OPTIONS**

19139 The *grep* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section
19140 12.2, Utility Syntax Guidelines.

19141 The following options shall be supported:

19142 **-E** Match using extended regular expressions. Treat each pattern specified as an ERE,
19143 as described in the Base Definitions volume of IEEE Std 1003.1-2001, Section 9.4,
19144 Extended Regular Expressions. If any entire ERE pattern matches some part of an
19145 input line excluding the terminating <newline>, the line shall be matched. A null
19146 ERE shall match every line.

19147 **-F** Match using fixed strings. Treat each pattern specified as a string instead of a
19148 regular expression. If an input line contains any of the patterns as a contiguous
19149 sequence of bytes, the line shall be matched. A null string shall match every line.

19150 **-c** Write only a count of selected lines to standard output.

19151 **-e** *pattern_list*

19152 Specify one or more patterns to be used during the search for input. The
19153 application shall ensure that patterns in *pattern_list* are separated by a <newline>.
19154 A null pattern can be specified by two adjacent <newline>s in *pattern_list*. Unless
19155 the **-E** or **-F** option is also specified, each pattern shall be treated as a BRE, as
19156 described in the Base Definitions volume of IEEE Std 1003.1-2001, Section 9.3, Basic
19157 Regular Expressions. Multiple **-e** and **-f** options shall be accepted by the *grep*
19158 utility. All of the specified patterns shall be used when matching lines, but the
19159 order of evaluation is unspecified.

19160	-f <i>pattern_file</i>	Read one or more patterns from the file named by the pathname <i>pattern_file</i> . Patterns in <i>pattern_file</i> shall be terminated by a <newline>. A null pattern can be specified by an empty line in <i>pattern_file</i> . Unless the -E or -F option is also specified, each pattern shall be treated as a BRE, as described in the Base Definitions volume of IEEE Std 1003.1-2001, Section 9.3, Basic Regular Expressions.
19166	-i	Perform pattern matching in searches without regard to case; see the Base Definitions volume of IEEE Std 1003.1-2001, Section 9.2, Regular Expression General Requirements.
19169	-l	(The letter ell.) Write only the names of files containing selected lines to standard output. Pathnames shall be written once per file searched. If the standard input is searched, a pathname of "(standard input)" shall be written, in the POSIX locale. In other locales, "standard input" may be replaced by something more appropriate in those locales.
19174	-n	Precede each output line by its relative line number in the file, each file starting at line 1. The line number counter shall be reset for each file processed.
19176	-q	Quiet. Nothing shall be written to the standard output, regardless of matching lines. Exit with zero status if an input line is selected.
19178	-s	Suppress the error messages ordinarily written for nonexistent or unreadable files. Other error messages shall not be suppressed.
19180	-v	Select lines not matching any of the specified patterns. If the -v option is not specified, selected lines shall be those that match any of the specified patterns.
19182	-x	Consider only input lines that use all characters in the line excluding the terminating <newline> to match an entire fixed string or regular expression to be matching lines.

19185 OPERANDS

19186 The following operands shall be supported:

19187	<i>pattern_list</i>	Specify one or more patterns to be used during the search for input. This operand shall be treated as if it were specified as -e <i>pattern_list</i> .
19189	<i>file</i>	A pathname of a file to be searched for the patterns. If no <i>file</i> operands are specified, the standard input shall be used.

19191 STDIN

19192 The standard input shall be used only if no *file* operands are specified. See the INPUT FILES section.

19194 INPUT FILES

19195 The input files shall be text files.

19196 ENVIRONMENT VARIABLES

19197 The following environment variables shall affect the execution of *grep*:

19198	<i>LANG</i>	Provide a default value for the internationalization variables that are unset or null. (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2, Internationalization Variables for the precedence of internationalization variables used to determine the values of locale categories.)
19202	<i>LC_ALL</i>	If set to a non-empty string value, override the values of all the other internationalization variables.

19204	<i>LC_COLLATE</i>	Determine the locale for the behavior of ranges, equivalence classes, and multi-character collating elements within regular expressions.
19205		
19206		
19207	<i>LC_CTYPE</i>	Determine the locale for the interpretation of sequences of bytes of text data as characters (for example, single-byte as opposed to multi-byte characters in arguments and input files) and the behavior of character classes within regular expressions.
19208		
19209		
19210		
19211	<i>LC_MESSAGES</i>	Determine the locale that should be used to affect the format and contents of diagnostic messages written to standard error.
19212		
19213		
19214 XSI	<i>NLSPATH</i>	Determine the location of message catalogs for the processing of <i>LC_MESSAGES</i> .
19215	ASYNCHRONOUS EVENTS	
19216		Default.
19217	STDOUT	
19218		If the -l option is in effect, and the -q option is not, the following shall be written for each file containing at least one selected input line:
19219		
19220		"%s\n", <file>
19221		Otherwise, if more than one <i>file</i> argument appears, and -q is not specified, the <i>grep</i> utility shall prefix each output line by:
19222		
19223		"%s:", <file>
19224		The remainder of each output line shall depend on the other options specified:
19225		<ul style="list-style-type: none">• If the -c option is in effect, the remainder of each output line shall contain:
19226		"%d\n", <count>
19227		<ul style="list-style-type: none">• Otherwise, if -c is not in effect and the -n option is in effect, the following shall be written to standard output:
19228		
19229		"%d:", <line number>
19230		<ul style="list-style-type: none">• Finally, the following shall be written to standard output:
19231		"%s", <selected-line contents>
19232	STDERR	
19233		The standard error shall be used only for diagnostic messages.
19234	OUTPUT FILES	
19235		None.
19236	EXTENDED DESCRIPTION	
19237		None.
19238	EXIT STATUS	
19239		The following exit values shall be returned:
19240		0 One or more lines were selected.
19241		1 No lines were selected.
19242		>1 An error occurred.

19243 CONSEQUENCES OF ERRORS

19244 If the **-q** option is specified, the exit status shall be zero if an input line is selected, even if an
19245 error was detected. Otherwise, default actions shall be performed.

19246 APPLICATION USAGE

19247 Care should be taken when using characters in *pattern_list* that may also be meaningful to the
19248 command interpreter. It is safest to enclose the entire *pattern_list* argument in single quotes:

19249 ' . . . '

19250 The **-e** *pattern_list* option has the same effect as the *pattern_list* operand, but is useful when
19251 *pattern_list* begins with the hyphen delimiter. It is also useful when it is more convenient to
19252 provide multiple patterns as separate arguments.

19253 Multiple **-e** and **-f** options are accepted and *grep* uses all of the patterns it is given while
19254 matching input text lines. (Note that the order of evaluation is not specified. If an
19255 implementation finds a null string as a pattern, it is allowed to use that pattern first, matching
19256 every line, and effectively ignore any other patterns.)

19257 The **-q** option provides a means of easily determining whether or not a pattern (or string) exists
19258 in a group of files. When searching several files, it provides a performance improvement
19259 (because it can quit as soon as it finds the first match) and requires less care by the user in
19260 choosing the set of files to supply as arguments (because it exits zero if it finds a match even if
19261 *grep* detected an access or read error on earlier *file* operands).

19262 EXAMPLES

- 19263 1. To find all uses of the word "Posix" (in any case) in file **text.mm** and write with line
19264 numbers:

19265 `grep -i -n posix text.mm`

- 19266 2. To find all empty lines in the standard input:

19267 `grep ^$`

19268 or:

19269 `grep -v .`

- 19270 3. Both of the following commands print all lines containing strings "abc" or "def" or both:

19271 `grep -E 'abc|def'`

19272 `grep -F 'abc`

1

19273 `def'`

1

- 19274 4. Both of the following commands print all lines matching exactly "abc" or "def":

19275 `grep -E '^abc$|^def$'`

19276 `grep -F -x 'abc`

1

19277 `def'`

1

19278 RATIONALE

19279 This *grep* has been enhanced in an upwards-compatible way to provide the exact functionality of
19280 the historical *egrep* and *fgrep* commands as well. It was the clear intention of the standard
19281 developers to consolidate the three *greps* into a single command.

19282 The old *egrep* and *fgrep* commands are likely to be supported for many years to come as
19283 implementation extensions, allowing historical applications to operate unmodified.

19284 Historical implementations usually silently ignored all but one of multiply-specified **-e** and **-f**
19285 options, but were not consistent as to which specification was actually used.

19286 The **-b** option was omitted from the OPTIONS section because block numbers are
19287 implementation-defined.

19288 The System V restriction on using **-** to mean standard input was omitted.

19289 A definition of action taken when given a null BRE or ERE is specified. This is an error condition
19290 in some historical implementations.

19291 The **-l** option previously indicated that its use was undefined when no files were explicitly
19292 named. This behavior was historical and placed an unnecessary restriction on future
19293 implementations. It has been removed.

19294 The historical BSD *grep -s* option practice is easily duplicated by redirecting standard output to
19295 **/dev/null**. The **-s** option required here is from System V.

19296 The **-x** option, historically available only with *fgrep*, is available here for all of the non-
19297 obsolescent versions.

19298 FUTURE DIRECTIONS

19299 None.

19300 SEE ALSO

19301 *sed*

19302 CHANGE HISTORY

19303 First released in Issue 2.

19304 Issue 6

19305 The Open Group Corrigendum U029/5 is applied, correcting the SYNOPSIS.

19306 The normative text is reworded to avoid use of the term “must” for application requirements.

19307 IEEE Std 1003.1-2001/Cor 1-2002, item XCU/TC1/D6/28 is applied, correcting the examples 1
19308 using the *grep -F* option which did not match the normative description of the **-F** option. 1

19309 **NAME**

19310 hash — remember or report utility locations

19311 **SYNOPSIS**

19312 XSI hash [*utility...*]

19313 hash -r

19314

19315 **DESCRIPTION**

19316 The *hash* utility shall affect the way the current shell environment remembers the locations of
19317 utilities found as described in Section 2.9.1.1 (on page 48). Depending on the arguments
19318 specified, it shall add utility locations to its list of remembered locations or it shall purge the
19319 contents of the list. When no arguments are specified, it shall report on the contents of the list.

19320 Utilities provided as built-ins to the shell shall not be reported by *hash*.

19321 **OPTIONS**

19322 The *hash* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section
19323 12.2, Utility Syntax Guidelines.

19324 The following option shall be supported:

19325 -r Forget all previously remembered utility locations.

19326 **OPERANDS**

19327 The following operand shall be supported:

19328 *utility* The name of a utility to be searched for and added to the list of remembered
19329 locations. If *utility* contains one or more slashes, the results are unspecified.

19330 **STDIN**

19331 Not used.

19332 **INPUT FILES**

19333 None.

19334 **ENVIRONMENT VARIABLES**

19335 The following environment variables shall affect the execution of *hash*:

19336 *LANG* Provide a default value for the internationalization variables that are unset or null.
19337 (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2,
19338 Internationalization Variables for the precedence of internationalization variables
19339 used to determine the values of locale categories.)

19340 *LC_ALL* If set to a non-empty string value, override the values of all the other
19341 internationalization variables.

19342 *LC_CTYPE* Determine the locale for the interpretation of sequences of bytes of text data as
19343 characters (for example, single-byte as opposed to multi-byte characters in
19344 arguments).

19345 *LC_MESSAGES*

19346 Determine the locale that should be used to affect the format and contents of
19347 diagnostic messages written to standard error.

19348 *NLSPATH* Determine the location of message catalogs for the processing of *LC_MESSAGES*.

19349 *PATH* Determine the location of *utility*, as described in the Base Definitions volume of
19350 IEEE Std 1003.1-2001, Chapter 8, Environment Variables.

19351 ASYNCHRONOUS EVENTS

19352 Default.

19353 STDOUT

19354 The standard output of *hash* shall be used when no arguments are specified. Its format is
19355 unspecified, but includes the pathname of each utility in the list of remembered locations for the
19356 current shell environment. This list shall consist of those utilities named in previous *hash*
19357 invocations that have been invoked, and may contain those invoked and found through the
19358 normal command search process.

19359 STDERR

19360 The standard error shall be used only for diagnostic messages.

19361 OUTPUT FILES

19362 None.

19363 EXTENDED DESCRIPTION

19364 None.

19365 EXIT STATUS

19366 The following exit values shall be returned:

19367 0 Successful completion.

19368 >0 An error occurred.

19369 CONSEQUENCES OF ERRORS

19370 Default.

19371 APPLICATION USAGE

19372 Since *hash* affects the current shell execution environment, it is always provided as a shell
19373 regular built-in. If it is called in a separate utility execution environment, such as one of the
19374 following:

19375 nohup hash -r
19376 find . -type f | xargs hash

19377 it does not affect the command search process of the caller's environment.

19378 The *hash* utility may be implemented as an alias—for example, *alias -t -*, in which case utilities
19379 found through normal command search are not listed by the *hash* command.

19380 The effects of *hash -r* can also be achieved portably by resetting the value of *PATH*; in the
19381 simplest form, this can be:

19382 PATH= "\$PATH"

19383 The use of *hash* with *utility* names is unnecessary for most applications, but may provide a
19384 performance improvement on a few implementations; normally, the hashing process is included
19385 by default.

19386 EXAMPLES

19387 None.

19388 RATIONALE

19389 None.

19390 FUTURE DIRECTIONS

19391 None.

19392 **SEE ALSO**

19393 Section 2.9.1.1 (on page 48)

19394 **CHANGE HISTORY**

19395 First released in Issue 2.

19396 NAME

19397 head — copy the first part of files

19398 SYNOPSIS

19399 head [-n *number*] [*file...*]

19400 DESCRIPTION

19401 The *head* utility shall copy its input files to the standard output, ending the output for each file at
19402 a designated point.

19403 Copying shall end at the point in each input file indicated by the **-n** *number* option. The option-
19404 argument *number* shall be counted in units of lines.

19405 OPTIONS

19406 The *head* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section
19407 12.2, Utility Syntax Guidelines.

19408 The following option shall be supported:

19409 **-n** *number* The first *number* lines of each input file shall be copied to standard output. The
19410 application shall ensure that the *number* option-argument is a positive decimal
19411 integer.

19412 When a file contains less than *number* lines, it shall be copied to standard output in its entirety.
19413 This shall not be an error.

19414 If no options are specified, *head* shall act as if **-n 10** had been specified.

19415 OPERANDS

19416 The following operand shall be supported:

19417 *file* A pathname of an input file. If no *file* operands are specified, the standard input
19418 shall be used.

19419 STDIN

19420 The standard input shall be used only if no *file* operands are specified. See the INPUT FILES
19421 section.

19422 INPUT FILES

19423 Input files shall be text files, but the line length is not restricted to {LINE_MAX} bytes.

19424 ENVIRONMENT VARIABLES

19425 The following environment variables shall affect the execution of *head*:

19426 **LANG** Provide a default value for the internationalization variables that are unset or null.
19427 (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2,
19428 Internationalization Variables for the precedence of internationalization variables
19429 used to determine the values of locale categories.)

19430 **LC_ALL** If set to a non-empty string value, override the values of all the other
19431 internationalization variables.

19432 **LC_CTYPE** Determine the locale for the interpretation of sequences of bytes of text data as
19433 characters (for example, single-byte as opposed to multi-byte characters in
19434 arguments and input files).

19435 LC_MESSAGES

19436 Determine the locale that should be used to affect the format and contents of
19437 diagnostic messages written to standard error.

19438 XSI **NLSPATH** Determine the location of message catalogs for the processing of *LC_MESSAGES*.

19439 ASYNCHRONOUS EVENTS

19440 Default.

19441 STDOUT

19442 The standard output shall contain designated portions of the input files.

19443 If multiple *file* operands are specified, *head* shall precede the output for each with the header:

19444 "*\n==> %s <=\n*" , <*pathname*>

19445 except that the first header written shall not include the initial <newline>.

19446 STDERR

19447 The standard error shall be used only for diagnostic messages.

19448 OUTPUT FILES

19449 None.

19450 EXTENDED DESCRIPTION

19451 None.

19452 EXIT STATUS

19453 The following exit values shall be returned:

19454 0 Successful completion.

19455 >0 An error occurred.

19456 CONSEQUENCES OF ERRORS

19457 Default.

19458 APPLICATION USAGE

19459 The obsolescent *-number* form is withdrawn in this version. Applications should use the *-n* *number* option.

19461 EXAMPLES

19462 To write the first ten lines of all files (except those with a leading period) in the directory:

19463 head *

19464 RATIONALE

19465 Although it is possible to simulate *head* with *sed 10q* for a single file, the standard developers
19466 decided that the popularity of *head* on historical BSD systems warranted its inclusion alongside
19467 *tail*.

19468 This standard version of *head* follows the Utility Syntax Guidelines. The *-n* option was added to
19469 this new interface so that *head* and *tail* would be more logically related.

19470 There is no *-c* option (as there is in *tail*) because it is not historical practice and because other
19471 utilities in this volume of IEEE Std 1003.1-2001 provide similar functionality.

19472 FUTURE DIRECTIONS

19473 None.

19474 SEE ALSO

19475 *sed, tail*

19476 CHANGE HISTORY

19477 First released in Issue 4.

19478 Issue 6

19479 The obsolescent **-number** form is withdrawn.

19480 The normative text is reworded to avoid use of the term “must” for application requirements.

19481 The DESCRIPTION is updated to clarify that when a file contains less than the number of lines requested, the entire file is copied to standard output.
19482

19483 NAME

19484 iconv — codeset conversion

19485 SYNOPSIS

19486 iconv [-cs] -f <i>frommap</i> -t <i>tomap</i> [<i>file</i> ...]	1
19487 iconv -f <i>fromcode</i> [-cs] [-t <i>tocode</i>] [<i>file</i> ...]	1
19488 iconv -t <i>tocode</i> [-cs] [-f <i>fromcode</i>] [<i>file</i> ...]	1
19489 iconv -l	1

19490 DESCRIPTION

19491 The *iconv* utility shall convert the encoding of characters in *file* from one codeset to another and
 19492 write the results to standard output.

19493 When the options indicate that charmap files are used to specify the codesets (see OPTIONS),
 19494 the codeset conversion shall be accomplished by performing a logical join on the symbolic
 19495 character names in the two charmaps. The implementation need not support the use of charmap
 19496 files for codeset conversion unless the POSIX2_LOCALEDEF symbol is defined on the system.

19497 OPTIONS

19498 The *iconv* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section
 19499 12.2, Utility Syntax Guidelines.

19500 The following options shall be supported:

19501 -c	Omit any characters that are invalid in the codeset of the input file from the 19502 output. When -c is not used, the results of encountering invalid characters in the 19503 input stream (either those that are not characters in the codeset of the input file or 19504 that have no corresponding character in the codeset of the output file) shall be 19505 specified in the system documentation. The presence or absence of -c shall not 19506 affect the exit status of <i>iconv</i> .	1
19507 -f <i>fromcodeset</i>	Identify the codeset of the input file. The implementation shall recognize the 19508 following two forms of the <i>fromcodeset</i> option-argument: 19509	1
19510 <i>fromcode</i>	The <i>fromcode</i> option-argument must not contain a slash character. It 19511 shall be interpreted as the name of one of the codeset descriptions 19512 provided by the implementation in an unspecified format. Valid 19513 values of <i>fromcode</i> are implementation-defined.	1
19514 <i>frommap</i>	The <i>frommap</i> option-argument must contain a slash character. It shall 19515 be interpreted as the pathname of a charmap file as defined in the 19516 Base Definitions volume of IEEE Std 1003.1-2001, Section 6.4, 19517 Character Set Description File. If the pathname does not represent a 19518 valid, readable charmap file, the results are undefined.	1
19519	If this option is omitted, the codeset of the current locale shall be used.	1
19520 -l	Write all supported <i>fromcode</i> and <i>tocode</i> values to standard output in an unspecified 19521 format.	1
19522 -s	Suppress any messages written to standard error concerning invalid characters. 19523 When -s is not used, the results of encountering invalid characters in the input 19524 stream (either those that are not valid characters in the codeset of the input file or 19525 that have no corresponding character in the codeset of the output file) shall be 19526 specified in the system documentation. The presence or absence of -s shall not 19527 affect the exit status of <i>iconv</i> .	1

19528	-t <i>tocodeset</i>	Identify the codeset to be used for the output file. The implementation shall recognize the following two forms of the <i>tocodeset</i> option-argument:	1
19529			1
19530	<i>tocode</i>	The semantics shall be equivalent to the -f <i>fromcode</i> option.	1
19531	<i>tomap</i>	The semantics shall be equivalent to the <i>tomap</i> option.	1
19532		If this option is omitted, the codeset of the current locale shall be used.	1
19533		If either -f or -t represents a charmap file, but the other does not (or is omitted), or both -f and -t are omitted, the results are undefined.	1
19534			
19535	OPERANDS		
19536		The following operand shall be supported:	
19537	<i>file</i>	A pathname of an input file. If no <i>file</i> operands are specified, or if a <i>file</i> operand is ' - ', the standard input shall be used.	
19538			
19539	STDIN		
19540		The standard input shall be used only if no <i>file</i> operands are specified, or if a <i>file</i> operand is ' - '.	
19541	INPUT FILES		
19542		The input file shall be a text file.	
19543	ENVIRONMENT VARIABLES		
19544		The following environment variables shall affect the execution of <i>iconv</i> :	
19545	<i>LANG</i>	Provide a default value for the internationalization variables that are unset or null. (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2, Internationalization Variables for the precedence of internationalization variables used to determine the values of locale categories.)	
19546			
19547			
19548			
19549	<i>LC_ALL</i>	If set to a non-empty string value, override the values of all the other internationalization variables.	
19550			
19551	<i>LC_CTYPE</i>	Determine the locale for the interpretation of sequences of bytes of text data as characters (for example, single-byte as opposed to multi-byte characters in arguments). During translation of the file, this variable is superseded by the use of the <i>fromcode</i> option-argument.	
19552			
19553			
19554			
19555	<i>LC_MESSAGES</i>		
19556		Determine the locale that should be used to affect the format and contents of diagnostic messages written to standard error.	
19557			
19558	XSI <i>NLSPATH</i>	Determine the location of message catalogs for the processing of <i>LC_MESSAGES</i> .	
19559	ASYNCHRONOUS EVENTS		
19560		Default.	
19561	STDOUT		
19562		When the -l option is used, the standard output shall contain all supported <i>fromcode</i> and <i>tocode</i> values, written in an unspecified format.	
19563			
19564		When the -l option is not used, the standard output shall contain the sequence of characters read from the input files, translated to the specified codeset. Nothing else shall be written to the standard output.	
19565			
19566			
19567	STDERR		
19568		The standard error shall be used only for diagnostic messages.	

19569 OUTPUT FILES

19570 None.

19571 EXTENDED DESCRIPTION

19572 None.

19573 EXIT STATUS

19574 The following exit values shall be returned:

19575 0 Successful completion.

19576 >0 An error occurred.

19577 CONSEQUENCES OF ERRORS

19578 Default.

19579 APPLICATION USAGE

19580 The user must ensure that both charmap files use the same symbolic names for characters the
19581 two codesets have in common.

19582 EXAMPLES

19583 The following example converts the contents of file **mail.x400** from the ISO/IEC 6937:2001 1
19584 standard codeset to the ISO/IEC 8859-1:1998 standard codeset, and stores the results in file 1
19585 **mail.local**:

19586 `iconv -f IS6937 -t IS8859 mail.x400 > mail.local`

19587 RATIONALE

19588 The *iconv* utility can be used portably only when the user provides two charmap files as option-
19589 arguments. This is because a single charmap provided by the user cannot reliably be joined with
19590 the names in a system-provided character set description. The valid values for *fromcode* and
19591 *tocode* are implementation-defined and do not have to have any relation to the charmap
19592 mechanisms. As an aid to interactive users, the **-l** option was adopted from the Plan 9 operating
19593 system. It writes information concerning these implementation-defined values. The format is
19594 unspecified because there are many possible useful formats that could be chosen, such as a
19595 matrix of valid combinations of *fromcode* and *tocode*. The **-l** option is not intended for shell script
19596 usage; conforming applications will have to use charmaps.

19597 FUTURE DIRECTIONS

19598 None.

19599 SEE ALSO

19600 *gencat*

19601 CHANGE HISTORY

19602 First released in Issue 3.

19603 Issue 6

19604 This utility has been rewritten to align with the IEEE P1003.2b draft standard. Specifically, the
19605 ability to use charmap files for conversion has been added.

19606 IEEE Std 1003.1-2001/Cor 1-2002, item XCU/TC1/D6/29 is applied, making changes to address 1
19607 inconsistencies with the *iconv()* function in the System Interfaces volume of 1
19608 IEEE Std 1003.1-2001. 1

19609 NAME

19610 *id* — return user identity

19611 SYNOPSIS

19612 *id* [*user*]

19613 *id* **-G**[**-n**] [*user*]

19614 *id* **-g**[**-nr**] [*user*]

19615 *id* **-u**[**-nr**] [*user*]

19616 DESCRIPTION

19617 If no *user* operand is provided, the *id* utility shall write the user and group IDs and the corresponding user and group names of the invoking process to standard output. If the effective and real IDs do not match, both shall be written. If multiple groups are supported by the underlying system (see the description of {NGROUPS_MAX} in the System Interfaces volume of IEEE Std 1003.1-2001), the supplementary group affiliations of the invoking process shall also be written.

19623 If a *user* operand is provided and the process has the appropriate privileges, the user and group IDs of the selected user shall be written. In this case, effective IDs shall be assumed to be identical to real IDs. If the selected user has more than one allowable group membership listed in the group database, these shall be written in the same manner as the supplementary groups described in the preceding paragraph.

19628 OPTIONS

19629 The *id* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section 12.2, Utility Syntax Guidelines.

19631 The following options shall be supported:

19632 **-G** Output all different group IDs (effective, real, and supplementary) only, using the format "%u\n". If there is more than one distinct group affiliation, output each such affiliation, using the format "%u", before the <newline> is output.

19635 **-g** Output only the effective group ID, using the format "%u\n".

19636 **-n** Output the name in the format "%s" instead of the numeric ID using the format "%u".

19638 **-r** Output the real ID instead of the effective ID.

19639 **-u** Output only the effective user ID, using the format "%u\n".

19640 OPERANDS

19641 The following operand shall be supported:

19642 *user* The login name for which information is to be written.

19643 STDIN

19644 Not used.

19645 INPUT FILES

19646 None.

19647 ENVIRONMENT VARIABLES

19648 The following environment variables shall affect the execution of *id*:

19649 **LANG** Provide a default value for the internationalization variables that are unset or null.
19650 (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2,
19651 Internationalization Variables for the precedence of internationalization variables

19652 used to determine the values of locale categories.)

19653 *LC_ALL* If set to a non-empty string value, override the values of all the other
19654 internationalization variables.

19655 *LC_CTYPE* Determine the locale for the interpretation of sequences of bytes of text data as
19656 characters (for example, single-byte as opposed to multi-byte characters in
19657 arguments).

19658 *LC_MESSAGES*

19659 Determine the locale that should be used to affect the format and contents of
19660 diagnostic messages written to standard error and informative messages written to
19661 standard output.

19662 XSI *NLSPATH* Determine the location of message catalogs for the processing of *LC_MESSAGES*.

19663 **ASYNCHRONOUS EVENTS**

19664 Default.

19665 **STDOUT**

19666 The following formats shall be used when the *LC_MESSAGES* locale category specifies the
19667 POSIX locale. In other locales, the strings *uid*, *gid*, *euid*, *egid*, and *groups* may be replaced with
19668 more appropriate strings corresponding to the locale.

19669 "uid=%u(%s) gid=%u(%s)\n", <real user ID>, <user-name>,
19670 <real group ID>, <group-name>

19671 If the effective and real user IDs do not match, the following shall be inserted immediately
19672 before the '\n' character in the previous format:

19673 " euid=%u(%s)"

19674 with the following arguments added at the end of the argument list:

19675 <effective user ID>, <effective user-name>

19676 If the effective and real group IDs do not match, the following shall be inserted directly before
19677 the '\n' character in the format string (and after any addition resulting from the effective and
19678 real user IDs not matching):

19679 " egid=%u(%s)"

19680 with the following arguments added at the end of the argument list:

19681 <effective group-ID>, <effective group name>

19682 If the process has supplementary group affiliations or the selected user is allowed to belong to
19683 multiple groups, the first shall be added directly before the <newline> in the format string:

19684 " groups=%u(%s)"

19685 with the following arguments added at the end of the argument list:

19686 <supplementary group ID>, <supplementary group name>

19687 and the necessary number of the following added after that for any remaining supplementary
19688 group IDs:

19689 " ,%u(%s)"

19690 and the necessary number of the following arguments added at the end of the argument list:

19691 <supplementary group ID>, <supplementary group name>

19692 If any of the user ID, group ID, effective user ID, effective group ID, or supplementary/multiple
19693 group IDs cannot be mapped by the system into printable user or group names, the
19694 corresponding "(%s)" and *name* argument shall be omitted from the corresponding format
19695 string.

19696 When any of the options are specified, the output format shall be as described in the OPTIONS
19697 section.

19698 **STDERR**

19699 The standard error shall be used only for diagnostic messages.

19700 **OUTPUT FILES**

19701 None.

19702 **EXTENDED DESCRIPTION**

19703 None.

19704 **EXIT STATUS**

19705 The following exit values shall be returned:

19706 0 Successful completion.

19707 >0 An error occurred.

19708 **CONSEQUENCES OF ERRORS**

19709 Default.

19710 **APPLICATION USAGE**

19711 Output produced by the -G option and by the default case could potentially produce very long
19712 lines on systems that support large numbers of supplementary groups. (On systems with user
19713 and group IDs that are 32-bit integers and with group names with a maximum of 8 bytes per
19714 name, 93 supplementary groups plus distinct effective and real group and user IDs could
19715 theoretically overflow the 2 048-byte {LINE_MAX} text file line limit on the default output case.
19716 It would take about 186 supplementary groups to overflow the 2 048-byte barrier using *id* -G).
19717 This is not expected to be a problem in practice, but in cases where it is a concern, applications
19718 should consider using *fold* -s before postprocessing the output of *id*.

19719 **EXAMPLES**

19720 None.

19721 **RATIONALE**

19722 The functionality provided by the 4 BSD *groups* utility can be simulated using:

19723 `id -Gn [user]`

19724 The 4 BSD command *groups* was considered, but it was not included because it did not provide
19725 the functionality of the *id* utility of the SVID. Also, it was thought that it would be easier to
19726 modify *id* to provide the additional functionality necessary to systems with multiple groups
19727 than to invent another command.

19728 The options -u, -g, -n, and -r were added to ease the use of *id* with shell commands
19729 substitution. Without these options it is necessary to use some preprocessor such as *sed* to select
19730 the desired piece of information. Since output such as that produced by:

19731 `id -u -n`

19732 is frequently wanted, it seemed desirable to add the options.

19733 **FUTURE DIRECTIONS**

19734 None.

19735 **SEE ALSO**19736 *fold, logname, who*, the System Interfaces volume of IEEE Std 1003.1-2001, *getgid()*, *getgroups()*,
19737 *getuid()*19738 **CHANGE HISTORY**

19739 First released in Issue 2.

19740 NAME

19741 ipcrm — remove an XSI message queue, semaphore set, or shared memory segment identifier

19742 SYNOPSIS

19743 XSI ipcrm [-q msgid | -Q msgkey | -s semid | -S semkey |
 19744 -m shmid | -M shmkey] ...
 19745

19746 DESCRIPTION

19747 The *ipcrm* utility shall remove zero or more message queues, semaphore sets, or shared memory
 19748 segments. The interprocess communication facilities to be removed are specified by the options.

19749 Only a user with appropriate privilege shall be allowed to remove an interprocess
 19750 communication facility that was not created by or owned by the user invoking *ipcrm*.

19751 OPTIONS

19752 The *ipcrm* facility supports the Base Definitions volume of IEEE Std 1003.1-2001, Section 12.2,
 19753 Utility Syntax Guidelines.

19754 The following options shall be supported:

- 19755 **-q msgid** Remove the message queue identifier *msgid* from the system and destroy the
 19756 message queue and data structure associated with it.
- 19757 **-m shmid** Remove the shared memory identifier *shmid* from the system. The shared memory
 19758 segment and data structure associated with it shall be destroyed after the last
 19759 detach.
- 19760 **-s semid** Remove the semaphore identifier *semid* from the system and destroy the set of
 19761 semaphores and data structure associated with it.
- 19762 **-Q msgkey** Remove the message queue identifier, created with key *msgkey*, from the system
 19763 and destroy the message queue and data structure associated with it.
- 19764 **-M shmkey** Remove the shared memory identifier, created with key *shmkey*, from the system.
 19765 The shared memory segment and data structure associated with it shall be
 19766 destroyed after the last detach.
- 19767 **-S semkey** Remove the semaphore identifier, created with key *semkey*, from the system and
 19768 destroy the set of semaphores and data structure associated with it.

19769 OPERANDS

19770 None.

19771 STDIN

19772 Not used.

19773 INPUT FILES

19774 None.

19775 ENVIRONMENT VARIABLES

19776 The following environment variables shall affect the execution of *ipcrm*:

- 19777 **LANG** Provide a default value for the internationalization variables that are unset or null.
 19778 (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2,
 19779 Internationalization Variables for the precedence of internationalization variables
 19780 used to determine the values of locale categories.)
- 19781 **LC_ALL** If set to a non-empty string value, override the values of all the other
 19782 internationalization variables.

19783	<i>LC_CTYPE</i>	Determine the locale for the interpretation of sequences of bytes of text data as characters (for example, single-byte as opposed to multi-byte characters in arguments).
19786	<i>LC_MESSAGES</i>	Determine the locale that should be used to affect the format and contents of diagnostic messages written to standard error.
19789	<i>NLSPATH</i>	Determine the location of message catalogs for the processing of <i>LC_MESSAGES</i> .
19790	ASYNCHRONOUS EVENTS	
19791		Default.
19792	STDOUT	
19793		Not used.
19794	STDERR	
19795		The standard error shall be used only for diagnostic messages.
19796	OUTPUT FILES	
19797		None.
19798	EXTENDED DESCRIPTION	
19799		None.
19800	EXIT STATUS	
19801		The following exit values shall be returned:
19802	0	Successful completion.
19803	>0	An error occurred.
19804	CONSEQUENCES OF ERRORS	
19805		Default.
19806	APPLICATION USAGE	
19807		None.
19808	EXAMPLES	
19809		None.
19810	RATIONALE	
19811		None.
19812	FUTURE DIRECTIONS	
19813		None.
19814	SEE ALSO	
19815		<i>ipcs</i> , the System Interfaces volume of IEEE Std 1003.1-2001, <i>msgctl()</i> , <i>semctl()</i> , <i>shmctl()</i>
19816	CHANGE HISTORY	
19817		First released in Issue 5.

19818 NAME

19819 ipcs — report XSI interprocess communication facilities status

19820 SYNOPSIS

19821 XSI ipcs [-qms][-a | -bcopt]

19822

19823 DESCRIPTION

19824 The *ipcs* utility shall write information about active interprocess communication facilities.

19825 Without options, information shall be written in short format for message queues, shared
19826 memory segments, and semaphore sets that are currently active in the system. Otherwise, the
19827 information that is displayed is controlled by the options specified.

19828 OPTIONS

19829 The *ipcs* facility supports the Base Definitions volume of IEEE Std 1003.1-2001, Section 12.2,
19830 Utility Syntax Guidelines.

19831 The *ipcs* utility accepts the following options:

19832 **-q** Write information about active message queues.

19833 **-m** Write information about active shared memory segments.

19834 **-s** Write information about active semaphore sets.

19835 If **-q**, **-m**, or **-s** are specified, only information about those facilities shall be written. If none of
19836 these three are specified, information about all three shall be written subject to the following
19837 options:

19838 **-a** Use all print options. (This is a shorthand notation for **-b**, **-c**, **-o**, **-p**, and **-t**.)

19839 **-b** Write information on maximum allowable size. (Maximum number of bytes in
19840 messages on queue for message queues, size of segments for shared memory, and
19841 number of semaphores in each set for semaphores.)

19842 **-c** Write creator's user name and group name; see below.

19843 **-o** Write information on outstanding usage. (Number of messages on queue and total
19844 number of bytes in messages on queue for message queues, and number of
19845 processes attached to shared memory segments.)

19846 **-p** Write process number information. (Process ID of the last process to send a
19847 message and process ID of the last process to receive a message on message
19848 queues, process ID of the creating process, and process ID of the last process to
19849 attach or detach on shared memory segments.)

19850 **-t** Write time information. (Time of the last control operation that changed the access
19851 permissions for all facilities, time of the last *msgsnd()* and *msgrcv()* operations on
19852 message queues, time of the last *shmat()* and *shmdt()* operations on shared
19853 memory, and time of the last *semop()* operation on semaphores.)

19854 OPERANDS

19855 None.

19856 STDIN

19857 Not used.

19858 INPUT FILES

- 19859 • The group database
19860 • The user database

19861 ENVIRONMENT VARIABLES

19862 The following environment variables shall affect the execution of *ipcs*:

- 19863 **LANG** Provide a default value for the internationalization variables that are unset or null.
19864 (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2,
19865 Internationalization Variables for the precedence of internationalization variables
19866 used to determine the values of locale categories.)
- 19867 **LC_ALL** If set to a non-empty string value, override the values of all the other
19868 internationalization variables.
- 19869 **LC_CTYPE** Determine the locale for the interpretation of sequences of bytes of text data as
19870 characters (for example, single-byte as opposed to multi-byte characters in
19871 arguments).
- 19872 **LC_MESSAGES**
19873 Determine the locale that should be used to affect the format and contents of
19874 diagnostic messages written to standard error.
- 19875 **NLSPATH** Determine the location of message catalogs for the processing of *LC_MESSAGES*.
- 19876 **TZ** Determine the timezone for the date and time strings written by *ipcs*. If *TZ* is unset
19877 or null, an unspecified default timezone shall be used.

19878 ASYNCHRONOUS EVENTS

19879 Default.

19880 STDOUT

19881 An introductory line shall be written with the format:

19882 "IPC status from %s as of %s\n", <source>, <date>

19883 where <source> indicates the source used to gather the statistics and <date> is the information
19884 that would be produced by the *date* command when invoked in the POSIX locale.

19885 The *ipcs* utility then shall create up to three reports depending upon the **-q**, **-m**, and **-s** options.
19886 The first report shall indicate the status of message queues, the second report shall indicate the
19887 status of shared memory segments, and the third report shall indicate the status of semaphore
19888 sets.

19889 If the corresponding facility is not installed or has not been used since the last reboot, then the
19890 report shall be written out in the format:

19891 "%s facility not in system.\n", <facility>

19892 where <facility> is *Message Queue*, *Shared Memory*, or *Semaphore*, as appropriate. If the facility has
19893 been installed and has been used since the last reboot, column headings separated by one or
19894 more spaces and followed by a <newline> shall be written as indicated below followed by the
19895 facility name written out using the format:

19896 "%s:\n", <facility>

19897 where <facility> is *Message Queues*, *Shared Memory*, or *Semaphores*, as appropriate. On the second
19898 and third reports the column headings need not be written if the last column headings written
19899 already provide column headings for all information in that report.

19900 The column headings provided in the first column below and the meaning of the information in
19901 those columns shall be given in order below; the letters in parentheses indicate the options that
19902 shall cause the corresponding column to appear; "all" means that the column shall always
19903 appear. Each column is separated by one or more <space>s. Note that these options only
19904 determine what information is provided for each report; they do not determine which reports
19905 are written.

19906 T (all) Type of facility:
19907 q Message queue.
19908 m Shared memory segment.
19909 s Semaphore.
19910 This field is a single character written using the format %c.
19911 ID (all) The identifier for the facility entry. This field shall be written using the format
19912 %d.
19913 KEY (all) The key used as an argument to *msgget()*, *semget()*, or *shmget()* to create the
19914 facility entry.
19915 **Note:** The key of a shared memory segment is changed to IPC_PRIVATE when
19916 the segment has been removed until all processes attached to the segment
19917 detach it.
19918 This field shall be written using the format 0x%
19919 MODE (all) The facility access modes and flags. The mode shall consist of 11 characters
19920 that are interpreted as follows.
19921 The first character shall be:
19922 S If a process is waiting on a *msgsnd()* operation.
19923 – If the above is not true.
19924 The second character shall be:
19925 R If a process is waiting on a *msgrecv()* operation.
19926 C or – If the associated shared memory segment is to be cleared when the
19927 first attach operation is executed.
19928 – If none of the above is true.
19929 The next nine characters shall be interpreted as three sets of three bits each.
19930 The first set refers to the owner's permissions; the next to permissions of
19931 others in the usergroup of the facility entry; and the last to all others. Within
19932 each set, the first character indicates permission to read, the second character
19933 indicates permission to write or alter the facility entry, and the last character is
19934 a minus sign ('-').
19935 The permissions shall be indicated as follows:
19936 r If read permission is granted.
19937 w If write permission is granted.
19938 a If alter permission is granted.
19939 – If the indicated permission is not granted.

19940		The first character following the permissions specifies if there is an alternate or additional access control method associated with the facility. If there is no alternate or additional access control method associated with the facility, a single <space> shall be written; otherwise, another printable character is written.
19945	OWNER (all)	The user name of the owner of the facility entry. If the user name of the owner is found in the user database, at least the first eight column positions of the name shall be written using the format %s. Otherwise, the user ID of the owner shall be written using the format %d.
19949	GROUP (all)	The group name of the owner of the facility entry. If the group name of the owner is found in the group database, at least the first eight column positions of the name shall be written using the format %s. Otherwise, the group ID of the owner shall be written using the format %d.
The following nine columns shall be only written out for message queues:		
19954	CREATOR (a,c)	The user name of the creator of the facility entry. If the user name of the creator is found in the user database, at least the first eight column positions of the name shall be written using the format %s. Otherwise, the user ID of the creator shall be written using the format %d.
19958	CGROUP (a,c)	The group name of the creator of the facility entry. If the group name of the creator is found in the group database, at least the first eight column positions of the name shall be written using the format %s. Otherwise, the group ID of the creator shall be written using the format %d.
19962	CBYTES (a,o)	The number of bytes in messages currently outstanding on the associated message queue. This field shall be written using the format %d.
19964	QNUM (a,o)	The number of messages currently outstanding on the associated message queue. This field shall be written using the format %d.
19966	QBYTES (a,b)	The maximum number of bytes allowed in messages outstanding on the associated message queue. This field shall be written using the format %d.
19968	LSPID (a,p)	The process ID of the last process to send a message to the associated queue. This field shall be written using the format: "%d", <pid> where <pid> is 0 if no message has been sent to the corresponding message queue; otherwise, <pid> shall be the process ID of the last process to send a message to the queue.
19974	LRPID (a,p)	The process ID of the last process to receive a message from the associated queue. This field shall be written using the format: "%d", <pid> where <pid> is 0 if no message has been received from the corresponding message queue; otherwise, <pid> shall be the process ID of the last process to receive a message from the queue.
19980	STIME (a,t)	The time the last message was sent to the associated queue. If a message has been sent to the corresponding message queue, the hour, minute, and second of the last time a message was sent to the queue shall be written using the format %d.%2.2d.%2.2d. Otherwise, the format "no-entry" shall be written.

19985	RTIME	(a,t)	The time the last message was received from the associated queue. If a message has been received from the corresponding message queue, the hour, minute, and second of the last time a message was received from the queue shall be written using the format %d:%2.2d:%2.2d. Otherwise, the format " no-entry" shall be written.
19986	The following eight columns shall be only written out for shared memory segments.		
19987	CREATOR	(a,c)	The user of the creator of the facility entry. If the user name of the creator is found in the user database, at least the first eight column positions of the name shall be written using the format %s. Otherwise, the user ID of the creator shall be written using the format %d.
19988	CGROUP	(a,c)	The group name of the creator of the facility entry. If the group name of the creator is found in the group database, at least the first eight column positions of the name shall be written using the format %s. Otherwise, the group ID of the creator shall be written using the format %d.
19989	NATTCH	(a,o)	The number of processes attached to the associated shared memory segment. This field shall be written using the format %d.
19990	SEGSZ	(a,b)	The size of the associated shared memory segment. This field shall be written using the format %d.
19991	CPID	(a,p)	The process ID of the creator of the shared memory entry. This field shall be written using the format %d.
19992	LPID	(a,p)	The process ID of the last process to attach or detach the shared memory segment. This field shall be written using the format: "%d" , <pid> where <pid> is 0 if no process has attached the corresponding shared memory segment; otherwise, <pid> shall be the process ID of the last process to attach or detach the segment.
19993	ATIME	(a,t)	The time the last attach on the associated shared memory segment was completed. If the corresponding shared memory segment has ever been attached, the hour, minute, and second of the last time the segment was attached shall be written using the format %d:%2.2d:%2.2d. Otherwise, the format " no-entry" shall be written.
19994	DTIME	(a,t)	The time the last detach on the associated shared memory segment was completed. If the corresponding shared memory segment has ever been detached, the hour, minute, and second of the last time the segment was detached shall be written using the format %d:%2.2d:%2.2d. Otherwise, the format " no-entry" shall be written.
19995	The following four columns shall be only written out for semaphore sets:		
19996	CREATOR	(a,c)	The user of the creator of the facility entry. If the user name of the creator is found in the user database, at least the first eight column positions of the name shall be written using the format %s. Otherwise, the user ID of the creator shall be written using the format %d.
19997	CGROUP	(a,c)	The group name of the creator of the facility entry. If the group name of the creator is found in the group database, at least the first eight column positions of the name shall be written using the format %s. Otherwise, the group ID of the creator shall be written using the format %d.
19998			

20030	NSEMS	(a,b)	The number of semaphores in the set associated with the semaphore entry.
20031			This field shall be written using the format %d.
20032	OTIME	(a,t)	The time the last semaphore operation on the set associated with the semaphore entry was completed. If a semaphore operation has ever been performed on the corresponding semaphore set, the hour, minute, and second of the last semaphore operation on the semaphore set shall be written using the format %d:%2.2d:%2.2d. Otherwise, the format " no-entry" shall be written.
20033			
20034			
20035			
20036			
20037			
20038			The following column shall be written for all three reports when it is requested:
20039	CTIME	(a,t)	The time the associated entry was created or changed. The hour, minute, and second of the time when the associated entry was created shall be written using the format %d:%2.2d:%2.2d.
20040			
20041			

20042 STDERR

20043 The standard error shall be used only for diagnostic messages.

20044 OUTPUT FILES

20045 None.

20046 EXTENDED DESCRIPTION

20047 None.

20048 EXIT STATUS

20049 The following exit values shall be returned:

20050 0 Successful completion.

20051 >0 An error occurred.

20052 CONSEQUENCES OF ERRORS

20053 Default.

20054 APPLICATION USAGE

20055 Things can change while *ipcs* is running; the information it gives is guaranteed to be accurate only when it was retrieved.

20057 EXAMPLES

20058 None.

20059 RATIONALE

20060 None.

20061 FUTURE DIRECTIONS

20062 None.

20063 SEE ALSO

20064 The System Interfaces volume of IEEE Std 1003.1-2001, *msgrecv()*, *msgsnd()*, *semget()*, *semop()*,
20065 *shmat()*, *shmdt()*, *shmget()*

20066 CHANGE HISTORY

20067 First released in Issue 5.

20068 Issue 6

20069 The Open Group Corrigendum U020/1 is applied, correcting the SYNOPSIS.

20070 The Open Group Corrigenda U032/1 and U032/2 are applied, clarifying the output format.

20071 The Open Group Base Resolution bwg98-004 is applied.

20072 NAME

20073 *jobs* — display status of jobs in the current session

20074 SYNOPSIS

20075 UP *jobs [-l | -p][job_id...]*

20076

20077 DESCRIPTION

20078 The *jobs* utility shall display the status of jobs that were started in the current shell environment; see Section 2.12 (on page 61).

20080 When *jobs* reports the termination status of a job, the shell shall remove its process ID from the list of those “known in the current shell execution environment”; see Section 2.9.3.1 (on page 50).

20083 OPTIONS

20084 The *jobs* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section 12.2, Utility Syntax Guidelines.

20086 The following options shall be supported:

20087 **-l** (The letter ell.) Provide more information about each job listed. This information shall include the job number, current job, process group ID, state, and the command that formed the job.

20090 **-p** Display only the process IDs for the process group leaders of the selected jobs.

20091 By default, the *jobs* utility shall display the status of all stopped jobs, running background jobs and all jobs whose status has changed and have not been reported by the shell.

20093 OPERANDS

20094 The following operand shall be supported:

20095 *job_id* Specifies the jobs for which the status is to be displayed. If no *job_id* is given, the status information for all jobs shall be displayed. The format of *job_id* is described in the Base Definitions volume of IEEE Std 1003.1-2001, Section 3.203, Job Control Job ID.

20099 STDIN

20100 Not used.

20101 INPUT FILES

20102 None.

20103 ENVIRONMENT VARIABLES

20104 The following environment variables shall affect the execution of *jobs*:

20105 **LANG** Provide a default value for the internationalization variables that are unset or null. (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2, Internationalization Variables for the precedence of internationalization variables used to determine the values of locale categories.)

20109 **LC_ALL** If set to a non-empty string value, override the values of all the other internationalization variables.

20111 **LC_CTYPE** Determine the locale for the interpretation of sequences of bytes of text data as characters (for example, single-byte as opposed to multi-byte characters in arguments).

20114 LC_MESSAGES

20115 Determine the locale that should be used to affect the format and contents of

20116 diagnostic messages written to standard error and informative messages written to
20117 standard output.

20118 XSI **NLSPATH** Determine the location of message catalogs for the processing of *LC_MESSAGES*.

20119 **ASYNCHRONOUS EVENTS**

20120 Default.

20121 **STDOUT**

20122 If the **-p** option is specified, the output shall consist of one line for each process ID:

20123 "*%d\n*", <process ID>

20124 Otherwise, if the **-l** option is not specified, the output shall be a series of lines of the form:

20125 "*[%d] %c %s %s\n*", <job-number>, <current>, <state>, <command>

20126 where the fields shall be as follows:

20127 <current> The character '+' identifies the job that would be used as a default for the *fg* or *bg*
20128 utilities; this job can also be specified using the *job_id* %+ or "%". The character
20129 '-' identifies the job that would become the default if the current default job were
20130 to exit; this job can also be specified using the *job_id* %- . For other jobs, this field is
20131 a <space>. At most one job can be identified with '+' and at most one job can be
20132 identified with '-'. If there is any suspended job, then the current job shall be a
20133 suspended job. If there are at least two suspended jobs, then the previous job also
20134 shall be a suspended job.

20135 <job-number> A number that can be used to identify the process group to the *wait*, *fg*, *bg*, and *kill*
20136 utilities. Using these utilities, the job can be identified by prefixing the job number
20137 with '%'.

20138 <state> One of the following strings (in the POSIX locale):

20139 **Running** Indicates that the job has not been suspended by a signal and has not
20140 exited.

20141 **Done** Indicates that the job completed and returned exit status zero.

20142 **Done(code)** Indicates that the job completed normally and that it exited with the
20143 specified non-zero exit status, *code*, expressed as a decimal number.

20144 **Stopped** Indicates that the job was suspended by the SIGTSTP signal.

20145 **Stopped (SIGTSTP)**
20146 Indicates that the job was suspended by the SIGTSTP signal.

20147 **Stopped (SIGSTOP)**
20148 Indicates that the job was suspended by the SIGSTOP signal.

20149 **Stopped (SIGTTIN)**
20150 Indicates that the job was suspended by the SIGTTIN signal.

20151 **Stopped (SIGTTOU)**
20152 Indicates that the job was suspended by the SIGTTOU signal.

20153 The implementation may substitute the string **Suspended** in place of **Stopped**. If
20154 the job was terminated by a signal, the format of <state> is unspecified, but it shall
20155 be visibly distinct from all of the other <state> formats shown here and shall
20156 indicate the name or description of the signal causing the termination.

- 20157 *<command>* The associated command that was given to the shell.
- 20158 If the **-l** option is specified, a field containing the process group ID shall be inserted before the
20159 *<state>* field. Also, more processes in a process group may be output on separate lines, using
20160 only the process ID and *<command>* fields.
- 20161 **STDERR**
- 20162 The standard error shall be used only for diagnostic messages.
- 20163 **OUTPUT FILES**
- 20164 None.
- 20165 **EXTENDED DESCRIPTION**
- 20166 None.
- 20167 **EXIT STATUS**
- 20168 The following exit values shall be returned:
- 20169 0 Successful completion.
- 20170 >0 An error occurred.
- 20171 **CONSEQUENCES OF ERRORS**
- 20172 Default.
- 20173 **APPLICATION USAGE**
- 20174 The **-p** option is the only portable way to find out the process group of a job because different
20175 implementations have different strategies for defining the process group of the job. Usage such
20176 as *\$jobs -p* provides a way of referring to the process group of the job in an implementation-
20177 independent way.
- 20178 The *jobs* utility does not work as expected when it is operating in its own utility execution
20179 environment because that environment has no applicable jobs to manipulate. See the
20180 APPLICATION USAGE section for *bg*. For this reason, *jobs* is generally implemented as a shell
20181 regular built-in.
- 20182 **EXAMPLES**
- 20183 None.
- 20184 **RATIONALE**
- 20185 Both "`%%`" and "`%+`" are used to refer to the current job. Both forms are of equal validity—the
20186 "`%%`" mirroring "`$$`" and "`%+`" mirroring the output of *jobs*. Both forms reflect historical
20187 practice of the KornShell and the C shell with job control.
- 20188 The job control features provided by *bg*, *fg*, and *jobs* are based on the KornShell. The standard
20189 developers examined the characteristics of the C shell versions of these utilities and found that
20190 differences exist. Despite widespread use of the C shell, the KornShell versions were selected for
20191 this volume of IEEE Std 1003.1-2001 to maintain a degree of uniformity with the rest of the
20192 KornShell features selected (such as the very popular command line editing features).
- 20193 The *jobs* utility is not dependent on the job control option, as are the seemingly related *bg* and *fg*
20194 utilities because *jobs* is useful for examining background jobs, regardless of the condition of job
20195 control. When the user has invoked a *set +m* command and job control has been turned off, *jobs*
20196 can still be used to examine the background jobs associated with that current session. Similarly,
20197 *kill* can then be used to kill background jobs with *kill% <background job number>*.
- 20198 The output for terminated jobs is left unspecified to accommodate various historical systems.
20199 The following formats have been witnessed:

- 20200 1. **Killed**(*signal name*)
- 20201 2. *signal name*
- 20202 3. *signal name*(**coredump**)
- 20203 4. *signal description*– core dumped
- 20204 Most users should be able to understand these formats, although it means that applications have trouble parsing them.
- 20205
- 20206 The calculation of job IDs was not described since this would suggest an implementation, which may impose unnecessary restrictions.
- 20207
- 20208 In an early proposal, a **-n** option was included to “Display the status of jobs that have changed, exited, or stopped since the last status report”. It was removed because the shell always writes any changed status of jobs before each prompt.
- 20209
- 20210
- 20211 **FUTURE DIRECTIONS**
- 20212 None.
- 20213 **SEE ALSO**
- 20214 Section 2.12 (on page 61), *bg*, *fg*, *kill*, *wait*
- 20215 **CHANGE HISTORY**
- 20216 First released in Issue 4.
- 20217 **Issue 6**
- 20218 This utility is marked as part of the User Portability Utilities option.
- 20219 The JC shading is removed as job control is mandatory in this issue.

20220 NAME

20221 join — relational database operator

20222 SYNOPSIS

20223 join [-a *file_number* | -v *file_number*] [-e *string*] [-o *list*] [-t *char*]
20224 [-1 *field*] [-2 *field*] *file1* *file2*

20225 DESCRIPTION

20226 The *join* utility shall perform an equality join on the files *file1* and *file2*. The joined files shall be
20227 written to the standard output.

20228 The join field is a field in each file on which the files are compared. The *join* utility shall write
20229 one line in the output for each pair of lines in *file1* and *file2* that have identical join fields. The
20230 output line by default shall consist of the join field, then the remaining fields from *file1*, then the
20231 remaining fields from *file2*. This format can be changed by using the **-o** option (see below). The
20232 **-a** option can be used to add unmatched lines to the output. The **-v** option can be used to output
20233 only unmatched lines.

20234 The files *file1* and *file2* shall be ordered in the collating sequence of *sort -b* on the fields on which
20235 they shall be joined, by default the first in each line. All selected output shall be written in the
20236 same collating sequence.

20237 The default input field separators shall be <blank>s. In this case, multiple separators shall count
20238 as one field separator, and leading separators shall be ignored. The default output field separator
20239 shall be a <space>.

20240 The field separator and collating sequence can be changed by using the **-t** option (see below).

20241 If the same key appears more than once in either file, all combinations of the set of remaining
20242 fields in *file1* and the set of remaining fields in *file2* are output in the order of the lines
20243 encountered.

20244 If the input files are not in the appropriate collating sequence, the results are unspecified.

20245 OPTIONS

20246 The *join* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section
20247 12.2, Utility Syntax Guidelines.

20248 The following options shall be supported:

20249 **-a** *file_number*

20250 Produce a line for each unpairable line in file *file_number*, where *file_number* is 1 or
20251 2, in addition to the default output. If both **-a1** and **-a2** are specified, all unpairable
20252 lines shall be output.

20253 **-e** *string*

Replace empty output fields in the list selected by **-o** with the string *string*.

20254 **-o** *list*

20255 Construct the output line to comprise the fields specified in *list*, each element of
which shall have one of the following two forms:

- 20256 1. *file_number.field*, where *file_number* is a file number and *field* is a decimal
20257 integer field number

- 20258 2. 0 (zero), representing the join field

20259 The elements of *list* shall be either comma-separated or <blank>-separated, as
20260 specified in Guideline 8 of the Base Definitions volume of IEEE Std 1003.1-2001,
20261 Section 12.2, Utility Syntax Guidelines. The fields specified by *list* shall be written
20262 for all selected output lines. Fields selected by *list* that do not appear in the input
20263 shall be treated as empty output fields. (See the **-e** option.) Only specifically

20264 requested fields shall be written. The application shall ensure that *list* is a single
20265 command line argument.

20266 **-t char** Use character *char* as a separator, for both input and output. Every appearance of
20267 *char* in a line shall be significant. When this option is specified, the collating
20268 sequence shall be the same as *sort* without the **-b** option.

20269 **-v file_number**
20270 Instead of the default output, produce a line only for each unpairable line in
20271 *file_number*, where *file_number* is 1 or 2. If both **-v1** and **-v2** are specified, all
20272 unpairable lines shall be output.

20273 **-1 field** Join on the *field*th field of file 1. Fields are decimal integers starting with 1.
20274 **-2 field** Join on the *field*th field of file 2. Fields are decimal integers starting with 1.

20275 OPERANDS

20276 The following operands shall be supported:

20277 *file1*, *file2* A pathname of a file to be joined. If either of the *file1* or *file2* operands is '**-**', the
20278 standard input shall be used in its place.

20279 STDIN

20280 The standard input shall be used only if the *file1* or *file2* operand is '**-**'. See the INPUT FILES
20281 section.

20282 INPUT FILES

20283 The input files shall be text files.

20284 ENVIRONMENT VARIABLES

20285 The following environment variables shall affect the execution of *join*:

20286 **LANG** Provide a default value for the internationalization variables that are unset or null.
20287 (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2,
20288 Internationalization Variables for the precedence of internationalization variables
20289 used to determine the values of locale categories.)

20290 **LC_ALL** If set to a non-empty string value, override the values of all the other
20291 internationalization variables.

20292 **LC_COLLATE** Determine the locale of the collating sequence *join* expects to have been used when
20293 the input files were sorted.

20295 **LC_CTYPE** Determine the locale for the interpretation of sequences of bytes of text data as
20296 characters (for example, single-byte as opposed to multi-byte characters in
20297 arguments and input files).

20298 **LC_MESSAGES** Determine the locale that should be used to affect the format and contents of
20299 diagnostic messages written to standard error.

20301 XSI **NLSPATH** Determine the location of message catalogs for the processing of **LC_MESSAGES**.

20302 ASYNCHRONOUS EVENTS

20303 Default.

20304 **STDOUT**

20305 The *join* utility output shall be a concatenation of selected character fields. When the **-o** option
 20306 is not specified, the output shall be:

```
20307        "%s%s\n", <join field>, <other file1 fields>,  

20308           <other file2 fields>
```

20309 If the join field is not the first field in a file, the <other file fields> for that file shall be:

```
20310        <fields preceding join field>, <fields following join field>
```

20311 When the **-o** option is specified, the output format shall be:

```
20312        ">%s\n", <concatenation of fields>
```

20313 where the concatenation of fields is described by the **-o** option, above.

20314 For either format, each field (except the last) shall be written with its trailing separator character.
 20315 If the separator is the default (<blank>s), a single <space> shall be written after each field
 20316 (except the last).

20317 **STDERR**

20318 The standard error shall be used only for diagnostic messages.

20319 **OUTPUT FILES**

20320 None.

20321 **EXTENDED DESCRIPTION**

20322 None.

20323 **EXIT STATUS**

20324 The following exit values shall be returned:

20325 0 All input files were output successfully.

20326 >0 An error occurred.

20327 **CONSEQUENCES OF ERRORS**

20328 Default.

20329 **APPLICATION USAGE**

20330 Pathnames consisting of numeric digits or of the form *string.string* should not be specified
 20331 directly following the **-o** list.

20332 **EXAMPLES**

20333 The **-o 0** field essentially selects the union of the join fields. For example, given file **phone**:

20334	!Name	Phone Number
20335	Don	+1 123-456-7890
20336	Hal	+1 234-567-8901
20337	Yasushi	+2 345-678-9012

20338 and file **fax**:

20339	!Name	Fax Number
20340	Don	+1 123-456-7899
20341	Keith	+1 456-789-0122
20342	Yasushi	+2 345-678-9011

20343 (where the large expanses of white space are meant to each represent a single <tab>), the
 20344 command:

20345 join -t "<tab>" -a 1 -a 2 -e '(unknown)' -o 0,1.2,2.2 phone fax
20346 would produce:

20347 ! Name	20348 Phone Number	20349 Fax Number
Don	+1 123-456-7890	+1 123-456-7899
Hal	+1 234-567-8901	(unknown)
Keith	(unknown)	+1 456-789-0122
Yasushi	+2 345-678-9012	+2 345-678-9011

20352 Multiple instances of the same key will produce combinatorial results. The following:

20353 fa:
20354 a x
20355 a y
20356 a z
20357 fb:
20358 a p

20359 will produce:

20360 a x p
20361 a y p
20362 a z p

20363 And the following:

20364 fa:
20365 a b c
20366 a d e
20367 fb:
20368 a w x
20369 a y z
20370 a o p

20371 will produce:

20372 a b c w x
20373 a b c y z
20374 a b c o p
20375 a d e w x
20376 a d e y z
20377 a d e o p

20378 RATIONALE

20379 The -e option is only effective when used with -o because, unless specific fields are identified
20380 using -o, join is not aware of what fields might be empty. The exception to this is the join field,
20381 but identifying an empty join field with the -e string is not historical practice and some scripts
20382 might break if this were changed.

20383 The 0 field in the -o list was adopted from the Tenth Edition version of join to satisfy
20384 international objections that the join in the base documents does not support the “full join” or
20385 “outer join” described in relational database literature. Although it has been possible to include
20386 a join field in the output (by default, or by field number using -o), the join field could not be
20387 included for an unpaired line selected by -a. The -o 0 field essentially selects the union of the
20388 join fields.

20389 This sort of outer join was not possible with the join commands in the base documents. The -o 0
20390 field was chosen because it is an upwards-compatible change for applications. An alternative

20391 was considered: have the join field represent the union of the fields in the files (where they are
20392 identical for matched lines, and one or both are null for unmatched lines). This was not adopted
20393 because it would break some historical applications.

20394 The ability to specify *file2* as – is not historical practice; it was added for completeness.

20395 The –v option is not historical practice, but was considered necessary because it permitted the
20396 writing of *only* those lines that do not match on the join field, as opposed to the –a option, which
20397 prints both lines that do and do not match. This additional facility is parallel with the –v option
20398 of *grep*.

20399 Some historical implementations have been encountered where a blank line in one of the input
20400 files was considered to be the end of the file; the description in this volume of
20401 IEEE Std 1003.1-2001 does not cite this as an allowable case.

20402 FUTURE DIRECTIONS

20403 None.

20404 SEE ALSO

20405 *awk, comm, sort, uniq*

20406 CHANGE HISTORY

20407 First released in Issue 2.

20408 Issue 6

20409 The obsolescent –j options and the multi-argument –o option are withdrawn in this issue.

20410 The normative text is reworded to avoid use of the term “must” for application requirements.

20411 NAME

20412 kill — terminate or signal processes

20413 SYNOPSIS

20414 kill -s *signal_name* *pid* ...20415 kill -l [*exit_status*]20416 XSI kill [-*signal_name*] *pid* ...20417 kill [-*signal_number*] *pid* ...

20418

20419 DESCRIPTION

20420 The *kill* utility shall send a signal to the process or processes specified by each *pid* operand.20421 For each *pid* operand, the *kill* utility shall perform actions equivalent to the *kill()* function
20422 defined in the System Interfaces volume of IEEE Std 1003.1-2001 called with the following
20423 arguments:

- 20424 • The value of the *pid* operand shall be used as the *pid* argument.
- 20425 • The *sig* argument is the value specified by the **-s** option, **-signal_number** option, or the
20426 **-signal_name** option, or by SIGTERM, if none of these options is specified.

20427 OPTIONS

20428 The *kill* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section
20429 XSI 12.2, Utility Syntax Guidelines, except that in the last two SYNOPSIS forms, the **-signal_number**
20430 and **-signal_name** options are usually more than a single character.

20431 The following options shall be supported:

20432 **-l** (The letter ell.) Write all values of *signal_name* supported by the implementation, if
20433 no operand is given. If an *exit_status* operand is given and it is a value of the '?'
20434 shell special parameter (see Section 2.5.2 (on page 34) and *wait*) corresponding to a
20435 process that was terminated by a signal, the *signal_name* corresponding to the
20436 signal that terminated the process shall be written. If an *exit_status* operand is
20437 given and it is the unsigned decimal integer value of a signal number, the
20438 *signal_name* (the symbolic constant name without the SIG prefix defined in the
20439 Base Definitions volume of IEEE Std 1003.1-2001) corresponding to that signal
20440 shall be written. Otherwise, the results are unspecified.

20441 **-s** *signal_name*

20442 Specify the signal to send, using one of the symbolic names defined in the
20443 **<signal.h>** header. Values of *signal_name* shall be recognized in a case-independent
20444 fashion, without the SIG prefix. In addition, the symbolic name 0 shall be
20445 recognized, representing the signal value zero. The corresponding signal shall be
20446 sent instead of SIGTERM.

20447 XSI **-signal_name**20448 Equivalent to **-s** *signal_name*.20449 XSI **-signal_number**

20450 Specify a non-negative decimal integer, *signal_number*, representing the signal to
20451 be used instead of SIGTERM, as the *sig* argument in the effective call to *kill()*. The
20452 correspondence between integer values and the *sig* value used is shown in the
20453 following list.

20454	The effects of specifying any <i>signal_number</i> other than those listed below are	1
20455	undefined.	1

20456	0	0	1
20457	1	SIGHUP	1
20458	2	SIGINT	1
20459	3	SIGQUIT	1
20460	6	SIGABRT	1
20461	9	SIGKILL	1
20462	14	SIGALRM	1
20463	15	SIGTERM	1
20464	If the first argument is a negative integer, it shall be interpreted as a <i>signal_number</i> option, not as a negative <i>pid</i> operand specifying a process group.		
20465			

20466 OPERANDS

20467 The following operands shall be supported:

20468 *pid* One of the following:

- 20469 1. A decimal integer specifying a process or process group to be signaled. The process or processes selected by positive, negative, and zero values of the *pid* operand shall be as described for the *kill()* function. If process number 0 is specified, all processes in the current process group shall be signaled. For the effects of negative *pid* numbers, see the *kill()* function defined in the System Interfaces volume of IEEE Std 1003.1-2001. If the first *pid* operand is negative, it should be preceded by "--" to keep it from being interpreted as an option.
- 20470 2. A job control job ID (see the Base Definitions volume of IEEE Std 1003.1-2001, Section 3.203, Job Control Job ID) that identifies a background process group to be signaled. The job control job ID notation is applicable only for invocations of *kill* in the current shell execution environment; see Section 2.12 (on page 61).

20481 *exit_status* A decimal integer specifying a signal number or the exit status of a process terminated by a signal.

20483 STDIN

20484 Not used.

20485 INPUT FILES

20486 None.

20487 ENVIRONMENT VARIABLES

20488 The following environment variables shall affect the execution of *kill*:

- 20489 *LANG* Provide a default value for the internationalization variables that are unset or null. (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2, Internationalization Variables for the precedence of internationalization variables used to determine the values of locale categories.)
- 20490 *LC_ALL* If set to a non-empty string value, override the values of all the other internationalization variables.
- 20491 *LC_CTYPE* Determine the locale for the interpretation of sequences of bytes of text data as characters (for example, single-byte as opposed to multi-byte characters in arguments).

20498	LC_MESSAGES	Determine the locale that should be used to affect the format and contents of diagnostic messages written to standard error.
20501 XSI	NLSPATH	Determine the location of message catalogs for the processing of <i>LC_MESSAGES</i> .
20502	ASYNCHRONOUS EVENTS	
20503		Default.
20504	STDOUT	
20505		When the -l option is not specified, the standard output shall not be used.
20506		When the -l option is specified, the symbolic name of each signal shall be written in the following format:
20507		"%s%c", <signal_name>, <separator>
20509		where the <signal_name> is in uppercase, without the SIG prefix, and the <separator> shall be either a <newline> or a <space>. For the last signal written, <separator> shall be a <newline>.
20511		When both the -l option and <i>exit_status</i> operand are specified, the symbolic name of the corresponding signal shall be written in the following format:
20512		"%s\n", <signal_name>
20513		
20514	STDERR	
20515		The standard error shall be used only for diagnostic messages.
20516	OUTPUT FILES	
20517		None.
20518	EXTENDED DESCRIPTION	
20519		None.
20520	EXIT STATUS	
20521		The following exit values shall be returned:
20522	0	At least one matching process was found for each <i>pid</i> operand, and the specified signal was successfully processed for at least one matching process.
20523	>0	An error occurred.
20524		
20525	CONSEQUENCES OF ERRORS	
20526		Default.
20527	APPLICATION USAGE	
20528		Process numbers can be found by using <i>ps</i> .
20529		The job control job ID notation is not required to work as expected when <i>kill</i> is operating in its own utility execution environment. In either of the following examples:
20530		
20531	nohup kill %1 &	
20532	system("kill %1");	
20533		the <i>kill</i> operates in a different environment and does not share the shell's understanding of job numbers.
20534		
20535	EXAMPLES	
20536		Any of the commands:
20537	kill -9 100 -165	
20538	kill -s kill 100 -165	
20539	kill -s KILL 100 -165	

20540 sends the SIGKILL signal to the process whose process ID is 100 and to all processes whose
20541 process group ID is 165, assuming the sending process has permission to send that signal to the
20542 specified processes, and that they exist.

20543 The System Interfaces volume of IEEE Std 1003.1-2001 and this volume of IEEE Std 1003.1-2001
20544 do not require specific signal numbers for any *signal_names*. Even the **-signal_number** option
20545 provides symbolic (although numeric) names for signals. If a process is terminated by a signal,
20546 its exit status indicates the signal that killed it, but the exact values are not specified. The **kill -l**
20547 option, however, can be used to map decimal signal numbers and exit status values into the
20548 name of a signal. The following example reports the status of a terminated job:

```
20549 job
20550 stat=$?
20551 if [ $stat -eq 0 ]
20552 then
20553     echo job completed successfully.
20554 elif [ $stat -gt 128 ]
20555 then
20556     echo job terminated by signal SIG$(kill -l $stat).
20557 else
20558     echo job terminated with error code $stat.
20559 fi
```

20560 To send the default signal to a process group (say 123), an application should use a command
20561 similar to one of the following:

```
20562 kill -TERM -123
20563 kill -- -123
```

20564 RATIONALE

20565 The **-l** option originated from the C shell, and is also implemented in the KornShell. The C shell
20566 output can consist of multiple output lines because the signal names do not always fit on a
20567 single line on some terminal screens. The KornShell output also included the implementation-
20568 defined signal numbers and was considered by the standard developers to be too difficult for
20569 scripts to parse conveniently. The specified output format is intended not only to accommodate
20570 the historical C shell output, but also to permit an entirely vertical or entirely horizontal listing
20571 on systems for which this is appropriate.

20572 An early proposal invented the name SIGNULL as a *signal_name* for signal 0 (used by the System
20573 Interfaces volume of IEEE Std 1003.1-2001 to test for the existence of a process without sending it
20574 a signal). Since the *signal_name* 0 can be used in this case unambiguously, SIGNULL has been
20575 removed.

20576 An early proposal also required symbolic *signal_names* to be recognized with or without the **SIG**
20577 prefix. Historical versions of **kill** have not written the **SIG** prefix for the **-l** option and have not
20578 recognized the **SIG** prefix on *signal_names*. Since neither applications portability nor ease-of-use
20579 would be improved by requiring this extension, it is no longer required.

20580 To avoid an ambiguity of an initial negative number argument specifying either a signal number
20581 or a process group, IEEE Std 1003.1-2001 mandates that it is always considered the former by
20582 implementations that support the XSI option. It also requires that conforming applications
20583 always use the **--** options terminator argument when specifying a process group, unless an
20584 option is also specified.

20585 The **-s** option was added in response to international interest in providing some form of **kill** that
20586 meets the Utility Syntax Guidelines.

20587 The job control job ID notation is not required to work as expected when *kill* is operating in its
20588 own utility execution environment. In either of the following examples:

20589 nohup kill %1 &
20590 system("kill %1");
20591 the *kill* operates in a different environment and does not understand how the shell has managed
20592 its job numbers.

20593 **FUTURE DIRECTIONS**

20594 None.

20595 **SEE ALSO**

20596 Chapter 2 (on page 29), *ps*, *wait*, the System Interfaces volume of IEEE Std 1003.1-2001, *kill()*, the
20597 Base Definitions volume of IEEE Std 1003.1-2001, [<signal.h>](#)

20598 **CHANGE HISTORY**

20599 First released in Issue 2.

20600 **Issue 6**

20601 The obsolescent versions of the SYNOPSIS are turned into non-obsolescent features of the XSI
20602 option, corresponding to a similar change in the *trap* special built-in.

20603 NAME

20604 lex — generate programs for lexical tasks (**DEVELOPMENT**)

20605 SYNOPSIS

20606 CD lex [-t][-n|-v][file ...]

20607

20608 DESCRIPTION

20609 The *lex* utility shall generate C programs to be used in lexical processing of character input, and
20610 that can be used as an interface to *yacc*. The C programs shall be generated from *lex* source code
20611 and conform to the ISO C standard. Usually, the *lex* utility shall write the program it generates to
20612 the file **lex.yy.c**; the state of this file is unspecified if *lex* exits with a non-zero exit status. See the
20613 **EXTENDED DESCRIPTION** section for a complete description of the *lex* input language.

20614 OPTIONS

20615 The *lex* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section 12.2,
20616 Utility Syntax Guidelines.

20617 The following options shall be supported:

20618 **-n** Suppress the summary of statistics usually written with the **-v** option. If no table
20619 sizes are specified in the *lex* source code and the **-v** option is not specified, then **-n**
20620 is implied.

20621 **-t** Write the resulting program to standard output instead of **lex.yy.c**.

20622 **-v** Write a summary of *lex* statistics to the standard output. (See the discussion of *lex*
20623 table sizes in **Definitions in lex** (on page 537).) If the **-t** option is specified and **-n**
20624 is not specified, this report shall be written to standard error. If table sizes are
20625 specified in the *lex* source code, and if the **-n** option is not specified, the **-v** option
20626 may be enabled.

20627 OPERANDS

20628 The following operand shall be supported:

20629 **file** A pathname of an input file. If more than one such *file* is specified, all files shall be
20630 concatenated to produce a single *lex* program. If no *file* operands are specified, or if
20631 a *file* operand is '**-**', the standard input shall be used.

20632 STDIN

20633 The standard input shall be used if no *file* operands are specified, or if a *file* operand is '**-**'. See
20634 **INPUT FILES**.

20635 INPUT FILES

20636 The input files shall be text files containing *lex* source code, as described in the **EXTENDED**
20637 **DESCRIPTION** section.

20638 ENVIRONMENT VARIABLES

20639 The following environment variables shall affect the execution of *lex*:

20640 **LANG** Provide a default value for the internationalization variables that are unset or null.
20641 (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2,
20642 Internationalization Variables for the precedence of internationalization variables
20643 used to determine the values of locale categories.)

20644 **LC_ALL** If set to a non-empty string value, override the values of all the other
20645 internationalization variables.

20646 **LC_COLLATE**

20647 Determine the locale for the behavior of ranges, equivalence classes, and multi-

20648	character collating elements within regular expressions. If this variable is not set to the POSIX locale, the results are unspecified.
20649	
20650	<i>LC_CTYPE</i> Determine the locale for the interpretation of sequences of bytes of text data as characters (for example, single-byte as opposed to multi-byte characters in arguments and input files), and the behavior of character classes within regular expressions. If this variable is not set to the POSIX locale, the results are unspecified.
20651	
20652	
20653	
20654	
20655	<i>LC_MESSAGES</i>
20656	Determine the locale that should be used to affect the format and contents of diagnostic messages written to standard error.
20657	
20658 XSI	<i>NLSPATH</i> Determine the location of message catalogs for the processing of <i>LC_MESSAGES</i> .
20659	ASYNCHRONOUS EVENTS
20660	Default.
20661	STDOUT
20662	If the -t option is specified, the text file of C source code output of <i>lex</i> shall be written to standard output.
20663	
20664	If the -t option is not specified:
20665	• Implementation-defined informational, error, and warning messages concerning the contents of <i>lex</i> source code input shall be written to either the standard output or standard error.
20666	
20667	• If the -v option is specified and the -n option is not specified, <i>lex</i> statistics shall also be written to either the standard output or standard error, in an implementation-defined format.
20668	
20669	These statistics may also be generated if table sizes are specified with a '%' operator in the <i>Definitions</i> section, as long as the -n option is not specified.
20670	
20671	STDERR
20672	If the -t option is specified, implementation-defined informational, error, and warning messages concerning the contents of <i>lex</i> source code input shall be written to the standard error.
20673	
20674	If the -t option is not specified:
20675	1. Implementation-defined informational, error, and warning messages concerning the contents of <i>lex</i> source code input shall be written to either the standard output or standard error.
20676	
20677	
20678	2. If the -v option is specified and the -n option is not specified, <i>lex</i> statistics shall also be written to either the standard output or standard error, in an implementation-defined format.
20679	
20680	These statistics may also be generated if table sizes are specified with a '%' operator in the <i>Definitions</i> section, as long as the -n option is not specified.
20681	
20682	OUTPUT FILES
20683	A text file containing C source code shall be written to lex.yy.c , or to the standard output if the -t option is present.
20684	
20685	EXTENDED DESCRIPTION
20686	Each input file shall contain <i>lex</i> source code, which is a table of regular expressions with corresponding actions in the form of C program fragments.
20687	
20688	When lex.yy.c is compiled and linked with the <i>lex</i> library (using the -l l operand with <i>c99</i>), the resulting program shall read character input from the standard input and shall partition it into strings that match the given expressions.
20689	
20690	

- 20691 When an expression is matched, these actions shall occur:
- The input string that was matched shall be left in `yytext` as a null-terminated string; `yytext` shall either be an external character array or a pointer to a character string. As explained in **Definitions in lex**, the type can be explicitly selected using the `%array` or `%pointer` declarations, but the default is implementation-defined.
 - The external `int yyleng` shall be set to the length of the matching string.
 - The expression's corresponding program fragment, or action, shall be executed.
- 20698 During pattern matching, *lex* shall search the set of patterns for the single longest possible match. Among rules that match the same number of characters, the rule given first shall be chosen.
- 20701 The general format of *lex* source shall be:
- 20702 *Definitions*
20703 `%%`
20704 *Rules*
20705 `%%`
20706 *UserSubroutines*
- 20707 The first "%%" is required to mark the beginning of the rules (regular expressions and actions);
20708 the second "%%" is required only if user subroutines follow.
- 20709 Any line in the *Definitions* section beginning with a <blank> shall be assumed to be a C program
20710 fragment and shall be copied to the external definition area of the `lex.yy.c` file. Similarly,
20711 anything in the *Definitions* section included between delimiter lines containing only "%{" and
20712 "%}" shall also be copied unchanged to the external definition area of the `lex.yy.c` file.
- 20713 Any such input (beginning with a <blank> or within "%{" and "%}" delimiter lines) appearing
20714 at the beginning of the *Rules* section before any rules are specified shall be written to `lex.yy.c`
20715 after the declarations of variables for the `yylex()` function and before the first line of code in
20716 `yylex()`. Thus, user variables local to `yylex()` can be declared here, as well as application code to
20717 execute upon entry to `yylex()`.
- 20718 The action taken by *lex* when encountering any input beginning with a <blank> or within "%{"
20719 and "%}" delimiter lines appearing in the *Rules* section but coming after one or more rules is
20720 undefined. The presence of such input may result in an erroneous definition of the `yylex()`
20721 function.
- 20722 **Definitions in lex**
- 20723 *Definitions* appear before the first "%%" delimiter. Any line in this section not contained between
20724 "%{" and "%}" lines and not beginning with a <blank> shall be assumed to define a *lex*
20725 substitution string. The format of these lines shall be:
- 20726 `name substitute`
- 20727 If a *name* does not meet the requirements for identifiers in the ISO C standard, the result is
20728 undefined. The string *substitute* shall replace the string {*name*} when it is used in a rule. The *name*
20729 string shall be recognized in this context only when the braces are provided and when it does
20730 not appear within a bracket expression or within double-quotes.
- 20731 In the *Definitions* section, any line beginning with a '%' (percent sign) character and followed by
20732 an alphanumeric word beginning with either 's' or 'S' shall define a set of start conditions.
20733 Any line beginning with a '%' followed by a word beginning with either 'x' or 'X' shall define
20734 a set of exclusive start conditions. When the generated scanner is in a %s state, patterns with no

20735 state specified shall be also active; in a %x state, such patterns shall not be active. The rest of the
 20736 line, after the first word, shall be considered to be one or more <blank>-separated names of start
 20737 conditions. Start condition names shall be constructed in the same way as definition names. Start
 20738 conditions can be used to restrict the matching of regular expressions to one or more states as
 20739 described in **Regular Expressions in lex** (on page 539).

20740 Implementations shall accept either of the following two mutually-exclusive declarations in the
 20741 *Definitions* section:

20742 **%array** Declare the type of *yytext* to be a null-terminated character array.

20743 **%pointer** Declare the type of *yytext* to be a pointer to a null-terminated character string.

20744 The default type of *yytext* is implementation-defined. If an application refers to *yytext* outside of
 20745 the scanner source file (that is, via an **extern**), the application shall include the appropriate
 20746 **%array** or **%pointer** declaration in the scanner source file.

20747 Implementations shall accept declarations in the *Definitions* section for setting certain internal
 20748 table sizes. The declarations are shown in the following table.

20749 **Table 4-9** Table Size Declarations in *lex*

20750 Declaration	20751 Description	20752 Minimum Value
20751 %p n	20752 Number of positions	2 500
20752 %n n	20753 Number of states	500
20753 %a n	20754 Number of transitions	2 000
20754 %e n	20755 Number of parse tree nodes	1 000
20755 %k n	20756 Number of packed character classes	1 000
20756 %o n	Size of the output array	3 000

20757 In the table, *n* represents a positive decimal integer, preceded by one or more <blank>s. The
 20758 exact meaning of these table size numbers is implementation-defined. The implementation shall
 20759 document how these numbers affect the *lex* utility and how they are related to any output that
 20760 may be generated by the implementation should limitations be encountered during the
 20761 execution of *lex*. It shall be possible to determine from this output which of the table size values
 20762 needs to be modified to permit *lex* to successfully generate tables for the input language. The
 20763 values in the column Minimum Value represent the lowest values conforming implementations
 20764 shall provide.

20765 **Rules in lex**

20766 The rules in *lex* source files are a table in which the left column contains regular expressions and
 20767 the right column contains actions (C program fragments) to be executed when the expressions
 20768 are recognized.

20769 *ERE action*
 20770 *ERE action*
 20771 . . .

20772 The extended regular expression (ERE) portion of a row shall be separated from *action* by one or
 20773 more <blank>s. A regular expression containing <blank>s shall be recognized under one of the
 20774 following conditions:

- 20775 • The entire expression appears within double-quotes.
- 20776 • The <blank>s appear within double-quotes or square brackets.

- 20777 • Each <blank> is preceded by a backslash character.
- 20778 **User Subroutines in lex**
- 20779 Anything in the user subroutines section shall be copied to **lex.yy.c** following **yylex()**.
- 20780 **Regular Expressions in lex**
- 20781 The *lex* utility shall support the set of extended regular expressions (see the Base Definitions volume of IEEE Std 1003.1-2001, Section 9.4, Extended Regular Expressions), with the following additions and exceptions to the syntax:
- 20784 " . . ." Any string enclosed in double-quotes shall represent the characters within the double-quotes as themselves, except that backslash escapes (which appear in the following table) shall be recognized. Any backslash-escape sequence shall be terminated by the closing quote. For example, "\01" "1" represents a single string: the octal value 1 followed by the character '1'.
- 20789 <*state*>*r*, <*state1*,*state2*,...>*r* The regular expression *r* shall be matched only when the program is in one of the start conditions indicated by *state*, *state1*, and so on; see **Actions in lex** (on page 541). (As an exception to the typographical conventions of the rest of this volume of IEEE Std 1003.1-2001, in this case <*state*> does not represent a metavariable, but the literal angle-bracket characters surrounding a symbol.) The start condition shall be recognized as such only at the beginning of a regular expression.
- 20796 *r/x* The regular expression *r* shall be matched only if it is followed by an occurrence of regular expression *x* (*x* is the instance of trailing context, further defined below). The token returned in *yytext* shall only match *r*. If the trailing portion of *r* matches the beginning of *x*, the result is unspecified. The *r* expression cannot include further trailing context or the '\$' (match-end-of-line) operator; *x* cannot include the '^' (match-beginning-of-line) operator, nor trailing context, nor the '\$' operator. That is, only one occurrence of trailing context is allowed in a *lex* regular expression, and the '^' operator only can be used at the beginning of such an expression.
- 20805 {*name*} When *name* is one of the substitution symbols from the *Definitions* section, the string, including the enclosing braces, shall be replaced by the *substitute* value. The *substitute* value shall be treated in the extended regular expression as if it were enclosed in parentheses. No substitution shall occur if {*name*} occurs within a bracket expression or within double-quotes.
- 20810 Within an ERE, a backslash character shall be considered to begin an escape sequence as specified in the table in the Base Definitions volume of IEEE Std 1003.1-2001, Chapter 5, File Format Notation ('\\', '\a', '\b', '\f', '\n', '\r', '\t', '\v'). In addition, the escape sequences in the following table shall be recognized.
- 20814 A literal <newline> cannot occur within an ERE; the escape sequence '\n' can be used to represent a <newline>. A <newline> shall not be matched by a period operator.

20816

Table 4-10 Escape Sequences in *lex*

20817 20818 20819 20820 20821 20822 20823 20824 20825 20826	Escape Sequence	Description	Meaning
20827 20828 20829 20830 20831 20832 20833 20834	\digits	A backslash character followed by the longest sequence of one, two, or three octal-digit characters (01234567). If all of the digits are 0 (that is, representation of the NUL character), the behavior is undefined.	The character whose encoding is represented by the one, two, or three-digit octal integer. Multi-byte characters require multiple, concatenated escape sequences of this type, including the leading '\ ' for each byte.
20835 20836 20837 20838 20839 20840 20841 20842	\xdigits	A backslash character followed by the longest sequence of hexadecimal-digit characters (01234567abcdefABCDEF). If all of the digits are 0 (that is, representation of the NUL character), the behavior is undefined.	The character whose encoding is represented by the hexadecimal integer.
20843 20844 20845	\c	A backslash character followed by any character not described in this table or in the table in the Base Definitions volume of IEEE Std 1003.1-2001, Chapter 5, File Format Notation ('\\", '\a', '\b', '\f', '\n', '\r', '\t', '\v').	The character 'c', unchanged.

Note: If a '\x' sequence needs to be immediately followed by a hexadecimal digit character, a sequence such as "\x1" "1" can be used, which represents a character containing the value 1, followed by the character '1'.

The order of precedence given to extended regular expressions for *lex* differs from that specified in the Base Definitions volume of IEEE Std 1003.1-2001, Section 9.4, Extended Regular Expressions. The order of precedence for *lex* shall be as shown in the following table, from high to low.

Note: The escaped characters entry is not meant to imply that these are operators, but they are included in the table to show their relationships to the true operators. The start condition, trailing context, and anchoring notations have been omitted from the table because of the placement restrictions described in this section; they can only appear at the beginning or ending of an ERE.

2

20855

Table 4-11 ERE Precedence in *lex*

Extended Regular Expression	Precedence
<i>collation-related bracket symbols</i>	[= =] [: :] [. .]
<i>escaped characters</i>	\<special character>
<i>bracket expression</i>	[]
<i>quoting</i>	" . . . "
<i>grouping</i>	()
<i>definition</i>	{ name }
<i>single-character RE duplication</i>	* + ?
<i>concatenation</i>	
<i>interval expression</i>	{ m , n }
<i>alternation</i>	

20867 The ERE anchoring operators '^' and '\$' do not appear in the table. With *lex* regular
 20868 expressions, these operators are restricted in their use: the '^' operator can only be used at the
 20869 beginning of an entire regular expression, and the '\$' operator only at the end. The operators
 20870 apply to the entire regular expression. Thus, for example, the pattern "(^abc)|(def\$)" is
 20871 undefined; it can instead be written as two separate rules, one with the regular expression
 20872 "^abc" and one with "def\$", which share a common action via the special '|' action (see
 20873 below). If the pattern were written "^abc|def\$", it would match either "abc" or "def" on a
 20874 line by itself.

20875 Unlike the general ERE rules, embedded anchoring is not allowed by most historical *lex*
 20876 implementations. An example of embedded anchoring would be for patterns such as
 20877 "(^|)foo(|\$)" to match "foo" when it exists as a complete word. This functionality can
 20878 be obtained using existing *lex* features:

```
20879 ^foo/[ \n]      |
20880 " foo"/[ \n]    /* Found foo as a separate word. */
```

20881 Note also that '\$' is a form of trailing context (it is equivalent to "/\n") and as such cannot be
 20882 used with regular expressions containing another instance of the operator (see the preceding
 20883 discussion of trailing context).

20884 The additional regular expressions trailing-context operator '//' can be used as an ordinary
 20885 character if presented within double-quotes, "/" ; preceded by a backslash, "\/" ; or within a
 20886 bracket expression, "[/]". The start-condition '<' and '>' operators shall be special only in a
 20887 start condition at the beginning of a regular expression; elsewhere in the regular expression they
 20888 shall be treated as ordinary characters.

20889 Actions in lex

20890 The action to be taken when an ERE is matched can be a C program fragment or the special
 20891 actions described below; the program fragment can contain one or more C statements, and can
 20892 also include special actions. The empty C statement ';' shall be a valid action; any string in the
 20893 **lex.yy.c** input that matches the pattern portion of such a rule is effectively ignored or skipped.
 20894 However, the absence of an action shall not be valid, and the action *lex* takes in such a condition
 20895 is undefined.

20896 The specification for an action, including C statements and special actions, can extend across
 20897 several lines if enclosed in braces:

```
20898 ERE <one or more blanks> { program statement
20899                                program statement }
```

20900 The default action when a string in the input to a **lex.yy.c** program is not matched by any
 20901 expression shall be to copy the string to the output. Because the default behavior of a program
 20902 generated by *lex* is to read the input and copy it to the output, a minimal *lex* source program that
 20903 has just "%%" shall generate a C program that simply copies the input to the output unchanged.

20904 Four special actions shall be available:

20905 | ECHO; REJECT; BEGIN

20906 | The action ' | ' means that the action for the next rule is the action for this rule.
 20907 Unlike the other three actions, ' | ' cannot be enclosed in braces or be semicolon-
 20908 terminated; the application shall ensure that it is specified alone, with no other
 20909 actions.

20910 **ECHO;** Write the contents of the string *yytext* on the output.

20911 **REJECT;** Usually only a single expression is matched by a given string in the input. **REJECT**
 20912 means "continue to the next expression that matches the current input", and shall
 20913 cause whatever rule was the second choice after the current rule to be executed for
 20914 the same input. Thus, multiple rules can be matched and executed for one input
 20915 string or overlapping input strings. For example, given the regular expressions
 20916 "*xyz*" and "*xy*" and the input "*xyz*", usually only the regular expression "*xyz*"
 20917 would match. The next attempted match would start after **z**. If the last action in the
 20918 "*xyz*" rule is **REJECT**, both this rule and the "*xy*" rule would be executed. The
 20919 **REJECT** action may be implemented in such a fashion that flow of control does not
 20920 continue after it, as if it were equivalent to a **goto** to another part of *yylex()*. The
 20921 use of **REJECT** may result in somewhat larger and slower scanners.

20922 **BEGIN** The action:

20923 BEGIN *newstate*;

20924 switches the state (start condition) to *newstate*. If the string *newstate* has not been
 20925 declared previously as a start condition in the *Definitions* section, the results are
 20926 unspecified. The initial state is indicated by the digit '0' or the token **INITIAL**.

20927 The functions or macros described below are accessible to user code included in the *lex* input. It
 20928 is unspecified whether they appear in the C code output of *lex*, or are accessible only through the
 20929 **-l l** operand to *c99* (the *lex* library).

20930 **int yylex(void)**

20931 Performs lexical analysis on the input; this is the primary function generated by the *lex*
 20932 utility. The function shall return zero when the end of input is reached; otherwise, it shall
 20933 return non-zero values (tokens) determined by the actions that are selected.

20934 **int yymore(void)**

20935 When called, indicates that when the next input string is recognized, it is to be appended to
 20936 the current value of *yytext* rather than replacing it; the value in *yyleng* shall be adjusted
 20937 accordingly.

20938 **int yyless(int n)**

20939 Retains *n* initial characters in *yytext*, NUL-terminated, and treats the remaining characters
 20940 as if they had not been read; the value in *yyleng* shall be adjusted accordingly.

20941 **int input(void)**

20942 Returns the next character from the input, or zero on end-of-file. It shall obtain input from
 20943 the stream pointer *yyin*, although possibly via an intermediate buffer. Thus, once scanning
 20944 has begun, the effect of altering the value of *yyin* is undefined. The character read shall be
 20945 removed from the input stream of the scanner without any processing by the scanner.

20946 **int unput(int c)**
20947 Returns the character 'c' to the input; *yytext* and *yylen* are undefined until the next
20948 expression is matched. The result of using *unput()* for more characters than have been input
20949 is unspecified.

20950 The following functions shall appear only in the *lex* library accessible through the **-l l** operand;
20951 they can therefore be redefined by a conforming application:

20952 **int yywrap(void)**
20953 Called by *yylex()* at end-of-file; the default *yywrap()* shall always return 1. If the application
20954 requires *yylex()* to continue processing with another source of input, then the application
20955 can include a function *yywrap()*, which associates another file with the external variable
20956 **FILE * yyin** and shall return a value of zero.

20957 **int main(int argc, char *argv[])**
20958 Calls *yylex()* to perform lexical analysis, then exits. The user code can contain *main()* to
20959 perform application-specific operations, calling *yylex()* as applicable.

20960 Except for *input()*, *unput()*, and *main()*, all external and static names generated by *lex* shall begin
20961 with the prefix **yy** or **YY**.

20962 EXIT STATUS

20963 The following exit values shall be returned:

20964 0 Successful completion.
20965 >0 An error occurred.

20966 CONSEQUENCES OF ERRORS

20967 Default.

20968 APPLICATION USAGE

20969 Conforming applications are warned that in the *Rules* section, an ERE without an action is not
20970 acceptable, but need not be detected as erroneous by *lex*. This may result in compilation or
20971 runtime errors.

20972 The purpose of *input()* is to take characters off the input stream and discard them as far as the
20973 lexical analysis is concerned. A common use is to discard the body of a comment once the
20974 beginning of a comment is recognized.

20975 The *lex* utility is not fully internationalized in its treatment of regular expressions in the *lex*
20976 source code or generated lexical analyzer. It would seem desirable to have the lexical analyzer
20977 interpret the regular expressions given in the *lex* source according to the environment specified
20978 when the lexical analyzer is executed, but this is not possible with the current *lex* technology.
20979 Furthermore, the very nature of the lexical analyzers produced by *lex* must be closely tied to the
20980 lexical requirements of the input language being described, which is frequently locale-specific
20981 anyway. (For example, writing an analyzer that is used for French text is not automatically
20982 useful for processing other languages.)

20983 EXAMPLES

20984 The following is an example of a *lex* program that implements a rudimentary scanner for a
20985 Pascal-like syntax:

```
20986   %{  
20987   /* Need this for the call to atof() below. */  
20988   #include <math.h>  
20989   /* Need this for printf(), fopen(), and stdin below. */  
20990   #include <stdio.h>  
20991   %}
```

```

20992     DIGIT      [0-9]
20993     ID         [a-z][a-z0-9]*
20994     %%
20995     {DIGIT}+ {
20996         printf("An integer: %s (%d)\n", yytext,
20997             atoi(yytext));
20998     }
20999     {DIGIT}+.{DIGIT}*      {
21000         printf("A float: %s (%g)\n", yytext,
21001             atof(yytext));
21002     }
21003     if|then|begin|end|procedure|function      {
21004         printf("A keyword: %s\n", yytext);
21005     }
21006     {ID}      printf("An identifier: %s\n", yytext);
21007     "+" | "-" | "*" | "/"      printf("An operator: %s\n", yytext);
21008     " {[^}\n]*}" /* Eat up one-line comments. */
21009     [ \t\n]+      /* Eat up white space. */
21010     .  printf("Unrecognized character: %s\n", yytext);
21011     %%
21012     int main(int argc, char *argv[])
21013     {
21014         ++argv, --argc; /* Skip over program name. */
21015         if (argc > 0)
21016             yyin = fopen(argv[0], "r");
21017         else
21018             yyin = stdin;
21019         yylex();
21020     }

```

21021 RATIONALE

21022 Even though the `-c` option and references to the C language are retained in this description, *lex*
21023 may be generalized to other languages, as was done at one time for EFL, the Extended
21024 FORTRAN Language. Since the *lex* input specification is essentially language-independent,
21025 versions of this utility could be written to produce Ada, Modula-2, or Pascal code, and there are
21026 known historical implementations that do so.

21027 The current description of *lex* bypasses the issue of dealing with internationalized EREs in the *lex*
21028 source code or generated lexical analyzer. If it follows the model used by *awk* (the source code is
21029 assumed to be presented in the POSIX locale, but input and output are in the locale specified by
21030 the environment variables), then the tables in the lexical analyzer produced by *lex* would
21031 interpret EREs specified in the *lex* source in terms of the environment variables specified when
21032 *lex* was executed. The desired effect would be to have the lexical analyzer interpret the EREs
21033 given in the *lex* source according to the environment specified when the lexical analyzer is
21034 executed, but this is not possible with the current *lex* technology.

21035 The description of octal and hexadecimal-digit escape sequences agrees with the ISO C standard 2
21036 use of escape sequences. 2

21037	Previous versions of this standard allowed for implementations with bytes other than eight bits,	2
21038	but this has been modified in this version.	2
21039	There is no detailed output format specification. The observed behavior of <i>lex</i> under four	
21040	different historical implementations was that none of these implementations consistently	
21041	reported the line numbers for error and warning messages. Furthermore, there was a desire that	
21042	<i>lex</i> be allowed to output additional diagnostic messages. Leaving message formats unspecified	
21043	avoids these formatting questions and problems with internationalization.	
21044	Although the %x specifier for <i>exclusive</i> start conditions is not historical practice, it is believed to	
21045	be a minor change to historical implementations and greatly enhances the usability of <i>lex</i>	
21046	programs since it permits an application to obtain the expected functionality with fewer	
21047	statements.	
21048	The %array and %pointer declarations were added as a compromise between historical systems.	
21049	The System V-based <i>lex</i> copies the matched text to a yytext array. The <i>flex</i> program, supported in	
21050	BSD and GNU systems, uses a pointer. In the latter case, significant performance improvements	
21051	are available for some scanners. Most historical programs should require no change in porting	
21052	from one system to another because the string being referenced is null-terminated in both cases.	
21053	(The method used by <i>flex</i> in its case is to null-terminate the token in place by remembering the	
21054	character that used to come right after the token and replacing it before continuing on to the next	
21055	scan.) Multi-file programs with external references to yytext outside the scanner source file	
21056	should continue to operate on their historical systems, but would require one of the new	
21057	declarations to be considered strictly portable.	
21058	The description of EREs avoids unnecessary duplication of ERE details because their meanings	
21059	within a <i>lex</i> ERE are the same as that for the ERE in this volume of IEEE Std 1003.1-2001.	
21060	The reason for the undefined condition associated with text beginning with a <blank> or within	
21061	"%{ " and "% }" delimiter lines appearing in the <i>Rules</i> section is historical practice. Both the BSD	
21062	and System V <i>lex</i> copy the indented (or enclosed) input in the <i>Rules</i> section (except at the	
21063	beginning) to unreachable areas of the <i>yylex()</i> function (the code is written directly after a <i>break</i>	
21064	statement). In some cases, the System V <i>lex</i> generates an error message or a syntax error,	
21065	depending on the form of indented input.	
21066	The intention in breaking the list of functions into those that may appear in <i>lex.yy.c</i> versus those	
21067	that only appear in <i>libl.a</i> is that only those functions in <i>libl.a</i> can be reliably redefined by a	
21068	conforming application.	
21069	The descriptions of standard output and standard error are somewhat complicated because	
21070	historical <i>lex</i> implementations chose to issue diagnostic messages to standard output (unless -t	
21071	was given). IEEE Std 1003.1-2001 allows this behavior, but leaves an opening for the more	
21072	expected behavior of using standard error for diagnostics. Also, the System V behavior of	
21073	writing the statistics when any table sizes are given is allowed, while BSD-derived systems can	
21074	avoid it. The programmer can always precisely obtain the desired results by using either the -t	
21075	or -n options.	
21076	The OPERANDS section does not mention the use of - as a synonym for standard input; not all	
21077	historical implementations support such usage for any of the <i>file</i> operands.	
21078	A description of the <i>translation table</i> was deleted from early proposals because of its relatively	
21079	low usage in historical applications.	
21080	The change to the definition of the <i>input()</i> function that allows buffering of input presents the	
21081	opportunity for major performance gains in some applications.	
21082	The following examples clarify the differences between <i>lex</i> regular expressions and regular	
21083	expressions appearing elsewhere in this volume of IEEE Std 1003.1-2001. For regular expressions	

21084 of the form "*r/x*", the string matching *r* is always returned; confusion may arise when the
21085 beginning of *x* matches the trailing portion of *r*. For example, given the regular expression
21086 "*a*b/cc*" and the input "aaabcc", yytext would contain the string "aaab" on this match. But
21087 given the regular expression "*x*/xy*" and the input "xxxy", the token **xxx**, not **xx**, is returned
21088 by some implementations because **xxx** matches "*x**".

21089 In the rule "*ab*/bc*", the "*b**" at the end of *r* extends *r*'s match into the beginning of the
21090 trailing context, so the result is unspecified. If this rule were "*ab/bc*", however, the rule
21091 matches the text "ab" when it is followed by the text "bc". In this latter case, the matching of *r*
21092 cannot extend into the beginning of *x*, so the result is specified.

21093 **FUTURE DIRECTIONS**

21094 None.

21095 **SEE ALSO**

21096 *c99, ed, yacc*

21097 **CHANGE HISTORY**

21098 First released in Issue 2.

21099 **Issue 6**

21100 This utility is marked as part of the C-Language Development Utilities option.

21101 The obsolescent **-c** option is withdrawn in this issue.

21102 The normative text is reworded to avoid use of the term "must" for application requirements.

21103 IEEE Std 1003.1-2001/Cor 2-2004, item XCU/TC2/D6/14 is applied, removing text describing behavior on systems with bytes consisting of more than eight bits. 2
21104

21105 NAME

21106 link — call *link()* function

21107 SYNOPSIS

21108 XSI link *file1 file2*

21109

21110 DESCRIPTION

21111 The *link* utility shall perform the function call:

21112 *link(file1, file2);*

21113 A user may need appropriate privilege to invoke the *link* utility.

21114 OPTIONS

21115 None.

21116 OPERANDS

21117 The following operands shall be supported:

21118 *file1* The pathname of an existing file.

21119 *file2* The pathname of the new directory entry to be created.

21120 STDIN

21121 Not used.

21122 INPUT FILES

21123 Not used.

21124 ENVIRONMENT VARIABLES

21125 The following environment variables shall affect the execution of *link*:

21126 *LANG* Provide a default value for the internationalization variables that are unset or null.
(See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2,
Internationalization Variables for the precedence of internationalization variables
used to determine the values of locale categories.)

21130 *LC_ALL* If set to a non-empty string value, override the values of all the other
internationalization variables.

21132 *LC_CTYPE* Determine the locale for the interpretation of sequences of bytes of text data as
characters (for example, single-byte as opposed to multi-byte characters in
arguments).

21135 *LC_MESSAGES*

21136 Determine the locale that should be used to affect the format and contents of
diagnostic messages written to standard error.

21138 *NLSPATH* Determine the location of message catalogs for the processing of *LC_MESSAGES*.

21139 ASYNCHRONOUS EVENTS

21140 Default.

21141 STDOUT

21142 None.

21143 STDERR

21144 The standard error shall be used only for diagnostic messages.

21145 OUTPUT FILES

21146 None.

21147 EXTENDED DESCRIPTION

21148 None.

21149 EXIT STATUS

21150 The following exit values shall be returned:

21151 0 Successful completion.

21152 >0 An error occurred.

21153 CONSEQUENCES OF ERRORS

21154 Default.

21155 APPLICATION USAGE

21156 None.

21157 EXAMPLES

21158 None.

21159 RATIONALE

21160 None.

21161 FUTURE DIRECTIONS

21162 None.

21163 SEE ALSO

21164 *In*, *unlink*, the System Interfaces volume of IEEE Std 1003.1-2001, *link()*

21165 CHANGE HISTORY

21166 First released in Issue 5.

21167 **NAME**21168 *ln* — link files21169 **SYNOPSIS**21170 *ln* [-fs] *source_file target_file*21171 *ln* [-fs] *source_file ... target_dir*21172 **DESCRIPTION**

21173 In the first synopsis form, the *ln* utility shall create a new directory entry (link) at the destination path specified by the *target_file* operand. If the **-s** option is specified, a symbolic link shall be created for the file specified by the *source_file* operand. This first synopsis form shall be assumed when the final operand does not name an existing directory; if more than two operands are specified and the final is not an existing directory, an error shall result.

21178 In the second synopsis form, the *ln* utility shall create a new directory entry (link), or if the **-s** option is specified a symbolic link, for each file specified by a *source_file* operand, at a destination path in the existing directory named by *target_dir*.

21181 If the last operand specifies an existing file of a type not specified by the System Interfaces volume of IEEE Std 1003.1-2001, the behavior is implementation-defined.

21183 The corresponding destination path for each *source_file* shall be the concatenation of the target directory pathname, a slash character, and the last pathname component of the *source_file*. The second synopsis form shall be assumed when the final operand names an existing directory.

21186 For each *source_file*:

- 21187 1. If the destination path exists:

- 21188 a. If the **-f** option is not specified, *ln* shall write a diagnostic message to standard error, do nothing more with the current *source_file*, and go on to any remaining *source_files*.

- 21190 b. Actions shall be performed equivalent to the *unlink()* function defined in the System Interfaces volume of IEEE Std 1003.1-2001, called using *destination* as the *path* argument. If this fails for any reason, *ln* shall write a diagnostic message to standard error, do nothing more with the current *source_file*, and go on to any remaining *source_files*.

- 21195 2. If the **-s** option is specified, *ln* shall create a symbolic link named by the destination path and containing as its pathname *source_file*. The *ln* utility shall do nothing more with *source_file* and shall go on to any remaining files.

- 21199 3. If *source_file* is a symbolic link, actions shall be performed equivalent to the *link()* function using the object that *source_file* references as the *path1* argument and the destination path as the *path2* argument. The *ln* utility shall do nothing more with *source_file* and shall go on to any remaining files.

- 21203 4. Actions shall be performed equivalent to the *link()* function defined in the System Interfaces volume of IEEE Std 1003.1-2001 using *source_file* as the *path1* argument, and the destination path as the *path2* argument.

21205 **OPTIONS**

21206 The *ln* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section 12.2, Utility Syntax Guidelines.

21208 The following option shall be supported:

21209 **-f** Force existing destination pathnames to be removed to allow the link.

21210 **-s** Create symbolic links instead of hard links.

21211 OPERANDS

21212 The following operands shall be supported:

21213 *source_file* A pathname of a file to be linked. If the **-s** option is specified, no restrictions on the
21214 type of file or on its existence shall be made. If the **-s** option is not specified,
21215 whether a directory can be linked is implementation-defined.

21216 *target_file* The pathname of the new directory entry to be created.

21217 *target_dir* A pathname of an existing directory in which the new directory entries are created.

21218 STDIN

21219 Not used.

21220 INPUT FILES

21221 None.

21222 ENVIRONMENT VARIABLES

21223 The following environment variables shall affect the execution of *ln*:

21224 *LANG* Provide a default value for the internationalization variables that are unset or null.
21225 (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2,
21226 Internationalization Variables for the precedence of internationalization variables
21227 used to determine the values of locale categories.)

21228 *LC_ALL* If set to a non-empty string value, override the values of all the other
21229 internationalization variables.

21230 *LC_CTYPE* Determine the locale for the interpretation of sequences of bytes of text data as
21231 characters (for example, single-byte as opposed to multi-byte characters in
21232 arguments).

21233 *LC_MESSAGES*

21234 Determine the locale that should be used to affect the format and contents of
21235 diagnostic messages written to standard error.

21236 XSI *NLSPATH* Determine the location of message catalogs for the processing of *LC_MESSAGES*.

21237 ASYNCHRONOUS EVENTS

21238 Default.

21239 STDOUT

21240 Not used.

21241 STDERR

21242 The standard error shall be used only for diagnostic messages.

21243 OUTPUT FILES

21244 None.

21245 EXTENDED DESCRIPTION

21246 None.

21247 EXIT STATUS

21248 The following exit values shall be returned:

21249 0 All the specified files were linked successfully.

21250 >0 An error occurred.

21251 **CONSEQUENCES OF ERRORS**

21252 Default.

21253 **APPLICATION USAGE**

21254 None.

21255 **EXAMPLES**

21256 None.

21257 **RATIONALE**

21258 Some historic versions of *ln* (including the one specified by the SVID) unlink the destination file, if it exists, by default. If the mode does not permit writing, these versions prompt for confirmation before attempting the unlink. In these versions the **-f** option causes *ln* not to attempt to prompt for confirmation.

21262 This allows *ln* to succeed in creating links when the target file already exists, even if the file itself is not writable (although the directory must be). Early proposals specified this functionality.

21264 This volume of IEEE Std 1003.1-2001 does not allow the *ln* utility to unlink existing destination paths by default for the following reasons:

- 21266 • The *ln* utility has historically been used to provide locking for shell applications, a usage that is incompatible with *ln* unlinking the destination path by default. There was no corresponding technical advantage to adding this functionality.
- 21269 • This functionality gave *ln* the ability to destroy the link structure of files, which changes the historical behavior of *ln*.
- 21271 • This functionality is easily replicated with a combination of *rm* and *ln*.
- 21272 • It is not historical practice in many systems; BSD and BSD-derived systems do not support this behavior. Unfortunately, whichever behavior is selected can cause scripts written expecting the other behavior to fail.
- 21275 • It is preferable that *ln* perform in the same manner as the *link()* function, which does not permit the target to exist already.

21277 This volume of IEEE Std 1003.1-2001 retains the **-f** option to provide support for shell scripts depending on the SVID semantics. It seems likely that shell scripts would not be written to handle prompting by *ln* and would therefore have specified the **-f** option.

21280 The **-f** option is an undocumented feature of many historical versions of the *ln* utility, allowing linking to directories. These versions require modification.

21282 Early proposals of this volume of IEEE Std 1003.1-2001 also required a **-i** option, which behaved like the **-i** options in *cp* and *mv*, prompting for confirmation before unlinking existing files. This was not historical practice for the *ln* utility and has been omitted.

21285 **FUTURE DIRECTIONS**

21286 None.

21287 **SEE ALSO**21288 *chmod*, *find*, *pax*, *rm*, the System Interfaces volume of IEEE Std 1003.1-2001, *link()*, *unlink()*21289 **CHANGE HISTORY**

21290 First released in Issue 2.

21291 Issue 6

21292 The *ln* utility is updated to include symbolic link processing as defined in the IEEE P1003.2b
21293 draft standard.

21294 NAME

21295 **locale** — get locale-specific information

21296 SYNOPSIS

21297 **locale** [-a | -m]21298 **locale** [-ck] *name*...

21299 DESCRIPTION

21300 The *locale* utility shall write information about the current locale environment, or all public
 21301 locales, to the standard output. For the purposes of this section, a *public locale* is one provided by
 21302 the implementation that is accessible to the application.

21303 When *locale* is invoked without any arguments, it shall summarize the current locale
 21304 environment for each locale category as determined by the settings of the environment variables
 21305 defined in the Base Definitions volume of IEEE Std 1003.1-2001, Chapter 7, Locale.

21306 When invoked with operands, it shall write values that have been assigned to the keywords in
 21307 the locale categories, as follows:

- 21308 • Specifying a keyword name shall select the named keyword and the category containing that
 keyword.
- 21310 • Specifying a category name shall select the named category and all keywords in that
 category.

21312 OPTIONS

21313 The *locale* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section
 21314 12.2, Utility Syntax Guidelines.

21315 The following options shall be supported:

- 21316 **-a** Write information about all available public locales. The available locales shall
 include **POSIX**, representing the POSIX locale. The manner in which the
 implementation determines what other locales are available is implementation-
 defined.
- 21320 **-c** Write the names of selected locale categories; see the STDOUT section. The **-c**
 option increases readability when more than one category is selected (for example,
 via more than one keyword name or via a category name). It is valid both with
 and without the **-k** option.
- 21324 **-k** Write the names and values of selected keywords. The implementation may omit
 values for some keywords; see the OPERANDS section.
- 21326 **-m** Write names of available charmaps; see the Base Definitions volume of
 IEEE Std 1003.1-2001, Section 6.1, Portable Character Set.

21328 OPERANDS

21329 The following operand shall be supported:

- 21330 *name* The name of a locale category as defined in the Base Definitions volume of
 IEEE Std 1003.1-2001, Chapter 7, Locale, the name of a keyword in a locale
 category, or the reserved name **charmap**. The named category or keyword shall be
 selected for output. If a single *name* represents both a locale category name and a
 keyword name in the current locale, the results are unspecified. Otherwise, both
 category and keyword names can be specified as *name* operands, in any sequence.
 It is implementation-defined whether any keyword values are written for the
 categories **LC_CTYPE** and **LC_COLLATE**.

21338 **STDIN**

21339 Not used.

21340 **INPUT FILES**

21341 None.

21342 **ENVIRONMENT VARIABLES**21343 The following environment variables shall affect the execution of *locale*:21344 **LANG** Provide a default value for the internationalization variables that are unset or null.
21345 (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2,
21346 Internationalization Variables for the precedence of internationalization variables
21347 used to determine the values of locale categories.)21348 **LC_ALL** If set to a non-empty string value, override the values of all the other
21349 internationalization variables.21350 **LC_CTYPE** Determine the locale for the interpretation of sequences of bytes of text data as
21351 characters (for example, single-byte as opposed to multi-byte characters in
21352 arguments and input files).21353 **LC_MESSAGES**21354 Determine the locale that should be used to affect the format and contents of
21355 diagnostic messages written to standard error.21356 XSI **NLSPATH** Determine the location of message catalogs for the processing of *LC_MESSAGES*.21357 XSI The application shall ensure that the *LANG*, *LC_**, and *NLSPATH* environment variables specify
21358 the current locale environment to be written out; they shall be used if the **-a** option is not
21359 specified.21360 **ASYNCHRONOUS EVENTS**

21361 Default.

21362 **STDOUT**21363 If *locale* is invoked without any options or operands, the names and values of the *LANG* and
21364 *LC_** environment variables described in this volume of IEEE Std 1003.1-2001 shall be written to
21365 the standard output, one variable per line, with *LANG* first, and each line using the following
21366 format. Only those variables set in the environment and not overridden by *LC_ALL* shall be
21367 written using this format:

21368 "%s=%s\n", <variable_name>, <value>

21369 The names of those *LC_** variables associated with locale categories defined in this volume of
21370 IEEE Std 1003.1-2001 that are not set in the environment or are overridden by *LC_ALL* shall be
21371 written in the following format:

21372 "%s=\\"%s\\n", <variable_name>, <implied value>

1

21373 The <implied value> shall be the name of the locale that has been selected for that category by the
21374 implementation, based on the values in *LANG* and *LC_ALL*, as described in the Base Definitions
21375 volume of IEEE Std 1003.1-2001, Chapter 8, Environment Variables.21376 The <value> and <implied value> shown above shall be properly quoted for possible later reentry
21377 to the shell. The <value> shall not be quoted using double-quotes (so that it can be distinguished
21378 by the user from the <implied value> case, which always requires double-quotes).21379 The *LC_ALL* variable shall be written last, using the first format shown above. If it is not set, it
21380 shall be written as:

21381 "LC_ALL=\n"

21382 If any arguments are specified:

- 21383 1. If the **-a** option is specified, the names of all the public locales shall be written, each in the
21384 following format:

21385 "%s\n", <locale name>

- 21386 2. If the **-c** option is specified, the names of all selected categories shall be written, each in the
21387 following format:

21388 "%s\n", <category name>

21389 If keywords are also selected for writing (see following items), the category name output
21390 shall precede the keyword output for that category.

21391 If the **-c** option is not specified, the names of the categories shall not be written; only the
21392 keywords, as selected by the <name> operand, shall be written.

- 21393 3. If the **-k** option is specified, the names and values of selected keywords shall be written. If
21394 a value is non-numeric, it shall be written in the following format:

21395 "%s=%s\n", <keyword name>, <keyword value>

21396 If the keyword was **charmap**, the name of the charmap (if any) that was specified via the
21397 *localeddef -f* option when the locale was created shall be written, with the word **charmap** as
21398 <keyword name>.

21399 If a value is numeric, it shall be written in one of the following formats:

21400 "%s=%d\n", <keyword name>, <keyword value>

21401 "%s=%c%c\n", <keyword name>, <escape character>, <keyword value>

21402 "%s=%cx%x\n", <keyword name>, <escape character>, <keyword value>

21403 where the <escape character> is that identified by the **escape_char** keyword in the current
21404 locale; see the Base Definitions volume of IEEE Std 1003.1-2001, Section 7.3, Locale
21405 Definition.

21406 Compound keyword values (list entries) shall be separated in the output by semicolons.
21407 When included in keyword values, the semicolon, the double-quote, the backslash, and
21408 any control character shall be preceded (escaped) with the escape character.

- 21409 4. If the **-k** option is not specified, selected keyword values shall be written, each in the
21410 following format:

21411 "%s\n", <keyword value>

21412 If the keyword was **charmap**, the name of the charmap (if any) that was specified via the
21413 *localeddef -f* option when the locale was created shall be written.

- 21414 5. If the **-m** option is specified, then a list of all available charmaps shall be written, each in
21415 the format:

21416 "%s\n", <charmap>

21417 where <charmap> is in a format suitable for use as the option-argument to the *localeddef -f*
21418 option.

21419 STDRERR

21420 The standard error shall be used only for diagnostic messages.

21421 OUTPUT FILES

21422 None.

21423 EXTENDED DESCRIPTION

21424 None.

21425 EXIT STATUS

21426 The following exit values shall be returned:

21427 0 All the requested information was found and output successfully.

21428 >0 An error occurred.

21429 CONSEQUENCES OF ERRORS

21430 Default.

21431 APPLICATION USAGE

21432 If the *LANG* environment variable is not set or set to an empty value, or one of the *LC_** environment variables is set to an unrecognized value, the actual locales assumed (if any) are implementation-defined as described in the Base Definitions volume of IEEE Std 1003.1-2001, Chapter 8, Environment Variables.

21436 Implementations are not required to write out the actual values for keywords in the categories *LC_CTYPE* and *LC_COLLATE*; however, they must write out the categories (allowing an application to determine, for example, which character classes are available).

21439 EXAMPLES

21440 In the following examples, the assumption is that locale environment variables are set as follows:

21442 *LANG=locale_x*
21443 *LC_COLLATE=locale_y*

21444 The command *locale* would result in the following output:

21445 *LANG=locale_x*
21446 *LC_CTYPE="locale_x"*
21447 *LC_COLLATE=locale_y*
21448 *LC_TIME="locale_x"*
21449 *LC_NUMERIC="locale_x"*
21450 *LC_MONETARY="locale_x"*
21451 *LC_MESSAGES="locale_x"*
21452 *LC_ALL=*

21453 The order of presentation of the categories is not specified by this volume of
21454 IEEE Std 1003.1-2001.

21455 The command:

21456 *LC_ALL=POSIX locale -ck decimal_point*

21457 would produce:

21458 *LC_NUMERIC*
21459 *decimal_point=". "*

21460 The following command shows an application of *locale* to determine whether a user-supplied
21461 response is affirmative:

```
21462     if printf "%s\n" "$response" | grep -Eq "\$(locale yesexpr)"  
21463     then  
21464         affirmative processing goes here  
21465     else  
21466         non-affirmative processing goes here  
21467     fi
```

21468 RATIONALE

21469 The output for categories *LC_CTYPE* and *LC_COLLATE* has been made implementation-defined
21470 because there is a questionable value in having a shell script receive an entire array of characters.
21471 It is also difficult to return a logical collation description, short of returning a complete *localeddef*
21472 source.

21473 The **-m** option was included to allow applications to query for the existence of charmaps. The
21474 output is a list of the charmaps (implementation-supplied and user-supplied, if any) on the
21475 system.

21476 The **-c** option was included for readability when more than one category is selected (for
21477 example, via more than one keyword name or via a category name). It is valid both with and
21478 without the **-k** option.

21479 The **charmap** keyword, which returns the name of the charmap (if any) that was used when the
21480 current locale was created, was included to allow applications needing the information to
21481 retrieve it.

21482 FUTURE DIRECTIONS

21483 None.

21484 SEE ALSO

21485 *localeddef*, the Base Definitions volume of IEEE Std 1003.1-2001, Section 7.3, Locale Definition

21486 CHANGE HISTORY

21487 First released in Issue 4.

21488 Issue 5

21489 The FUTURE DIRECTIONS section is added.

21490 Issue 6

21491 The normative text is reworded to avoid use of the term “must” for application requirements.

21492 IEEE Std 1003.1-2001/Cor 1-2002, item XCU/TC1/D6/30 is applied, correcting an editorial error 1
21493 in the STDOUT section. 1

21494 NAME

21495 localedef — define locale environment

21496 SYNOPSIS

21497 `localedef [-c][-f charmap][-i sourcefile][-u code_set_name] name`

21498 DESCRIPTION

21499 The *localedef* utility shall convert source definitions for locale categories into a format usable by
21500 the functions and utilities whose operational behavior is determined by the setting of the locale
21501 environment variables defined in the Base Definitions volume of IEEE Std 1003.1-2001, Chapter
21502 7, Locale. It is implementation-defined whether users have the capability to create new locales,
21503 in addition to those supplied by the implementation. If the symbolic constant
21504 `XSI` `POSIX2_LOCALEDEF` is defined, the system supports the creation of new locales. On `XSI`-
21505 conformant systems, the symbolic constant `POSIX2_LOCALEDEF` shall be defined.

21506 The utility shall read source definitions for one or more locale categories belonging to the same
21507 locale from the file named in the `-i` option (if specified) or from standard input.

21508 The *name* operand identifies the target locale. The utility shall support the creation of *public*, or
21509 generally accessible locales, as well as *private*, or restricted-access locales. Implementations may
21510 restrict the capability to create or modify public locales to users with the appropriate privileges.

21511 Each category source definition shall be identified by the corresponding environment variable
21512 name and terminated by an `END category-name` statement. The following categories shall be
21513 supported. In addition, the input may contain source for implementation-defined categories.

21514 ***LC_CTYPE*** Defines character classification and case conversion.

21515 ***LC_COLLATE***

21516 Defines collation rules.

21517 ***LC_MONETARY***

21518 Defines the format and symbols used in formatting monetary information.

21519 ***LC_NUMERIC***

21520 Defines the decimal delimiter, grouping, and grouping symbol for non-monetary
21521 numeric editing.

21522 ***LC_TIME*** Defines the format and content of date and time information.

21523 ***LC_MESSAGES***

21524 Defines the format and values of affirmative and negative responses.

21525 OPTIONS

21526 The *localedef* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section
21527 12.2, Utility Syntax Guidelines.

21528 The following options shall be supported:

21529 ***-c*** Create permanent output even if warning messages have been issued.

21530 ***-f charmap*** Specify the pathname of a file containing a mapping of character symbols and
21531 collating element symbols to actual character encodings. The format of the
21532 *charmap* is described in the Base Definitions volume of IEEE Std 1003.1-2001,
21533 Section 6.4, Character Set Description File. The application shall ensure that this
21534 option is specified if symbolic names (other than collating symbols defined in a
21535 ***collating-symbol*** keyword) are used. If the `-f` option is not present, an
21536 implementation-defined character mapping shall be used.

- 21537 **-i *inputfile*** The pathname of a file containing the source definitions. If this option is not
 21538 present, source definitions shall be read from standard input. The format of the
 21539 *inputfile* is described in the Base Definitions volume of IEEE Std 1003.1-2001,
 21540 Section 7.3, Locale Definition.
- 21541 **-u *code_set_name***
 21542 Specify the name of a codeset used as the target mapping of character symbols and
 21543 collating element symbols whose encoding values are defined in terms of the
 21544 ISO/IEC 10646-1:2000 standard position constant values.

21545 OPERANDS

21546 The following operand shall be supported:

- 21547 ***name*** Identifies the locale; see the Base Definitions volume of IEEE Std 1003.1-2001,
 21548 Chapter 7, Locale for a description of the use of this name. If the name contains one
 21549 or more slash characters, *name* shall be interpreted as a pathname where the
 21550 created locale definitions shall be stored. If *name* does not contain any slash
 21551 characters, the interpretation of the name is implementation-defined and the locale
 21552 shall be public. The ability to create public locales in this way may be restricted to
 21553 users with appropriate privileges. (As a consequence of specifying one *name*,
 21554 although several categories can be processed in one execution, only categories
 21555 belonging to the same locale can be processed.)

21556 STDIN

21557 Unless the **-i** option is specified, the standard input shall be a text file containing one or more
 21558 locale category source definitions, as described in the Base Definitions volume of
 21559 IEEE Std 1003.1-2001, Section 7.3, Locale Definition. When lines are continued using the escape
 21560 character mechanism, there is no limit to the length of the accumulated continued line.

21561 INPUT FILES

21562 The character set mapping file specified as the **charmap** option-argument is described in the Base
 21563 Definitions volume of IEEE Std 1003.1-2001, Section 6.4, Character Set Description File. If a locale
 21564 category source definition contains a **copy** statement, as defined in the Base Definitions volume
 21565 of IEEE Std 1003.1-2001, Chapter 7, Locale, and the **copy** statement names a valid, existing locale,
 21566 then **localedef** shall behave as if the source definition had contained a valid category source
 21567 definition for the named locale.

21568 ENVIRONMENT VARIABLES

21569 The following environment variables shall affect the execution of **localedef**:

- 21570 **LANG** Provide a default value for the internationalization variables that are unset or null.
 21571 (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2,
 21572 Internationalization Variables for the precedence of internationalization variables
 21573 used to determine the values of locale categories.)
- 21574 **LC_ALL** If set to a non-empty string value, override the values of all the other
 21575 internationalization variables.
- 21576 **LC_COLLATE**
 21577 (This variable has no affect on **localedef**; the POSIX locale is used for this category.)
- 21578 **LC_CTYPE** Determine the locale for the interpretation of sequences of bytes of text data as
 21579 characters (for example, single-byte as opposed to multi-byte characters in
 21580 arguments and input files). This variable has no affect on the processing of **localedef**
 21581 input data; the POSIX locale is used for this purpose, regardless of the value of this
 21582 variable.

2
2

21583	LC_MESSAGES	Determine the locale that should be used to affect the format and contents of diagnostic messages written to standard error.
21584		
21585		
21586 XSI	NLSPATH	Determine the location of message catalogs for the processing of <i>LC_MESSAGES</i> .
21587	ASYNCHRONOUS EVENTS	
21588		Default.
21589	STDOUT	
21590		The utility shall report all categories successfully processed, in an unspecified format.
21591	STDERR	
21592		The standard error shall be used only for diagnostic messages.
21593	OUTPUT FILES	
21594		The format of the created output is unspecified. If the <i>name</i> operand does not contain a slash, the existence of an output file for the locale is unspecified.
21595		
21596	EXTENDED DESCRIPTION	
21597		When the -u option is used, the <i>code_set_name</i> option-argument shall be interpreted as an implementation-defined name of a codeset to which the ISO/IEC 10646-1:2000 standard position constant values shall be converted via an implementation-defined method. Both the ISO/IEC 10646-1:2000 standard position constant values and other formats (decimal, hexadecimal, or octal) shall be valid as encoding values within the <i>charmap</i> file. The codeset represented by the implementation-defined name can be any codeset that is supported by the implementation.
21598		
21599		
21600		
21601		
21602		
21603		
21604		When conflicts occur between the <i>charmap</i> specification of < <i>code_set_name</i> >, < <i>mb_cur_max</i> >, or < <i>mb_cur_min</i> > and the implementation-defined interpretation of these respective items for the codeset represented by the -u option-argument <i>code_set_name</i> , the result is unspecified.
21605		
21606		
21607		When conflicts occur between the <i>charmap</i> encoding values specified for symbolic names of characters of the portable character set and the implementation-defined assignment of character encoding values, the result is unspecified.
21608		
21609		
21610		If a non-printable character in the <i>charmap</i> has a width specified that is not -1 , the result will be undefined.
21611		2
21612	EXIT STATUS	
21613		The following exit values shall be returned:
21614	0	No errors occurred and the locales were successfully created.
21615	1	Warnings occurred and the locales were successfully created.
21616	2	The locale specification exceeded implementation limits or the coded character set or sets used were not supported by the implementation, and no locale was created.
21617		
21618	3	The capability to create new locales is not supported by the implementation.
21619	>3	Warnings or errors occurred and no output was created.
21620	CONSEQUENCES OF ERRORS	
21621		If an error is detected, no permanent output shall be created.
21622		If warnings occur, permanent output shall be created if the -c option was specified. The following conditions shall cause warning messages to be issued:
21623		
21624		<ul style="list-style-type: none"> • If a symbolic name not found in the <i>charmap</i> file is used for the descriptions of the <i>LC_CTYPE</i> or <i>LC_COLLATE</i> categories (for other categories, this shall be an error condition).
21625		

- 21626 • If the number of operands to the **order** keyword exceeds the {COLL_WEIGHTS_MAX} limit.
21627 • If optional keywords not supported by the implementation are present in the source.
21628 Other implementation-defined conditions may also cause warnings.

2

21629 APPLICATION USAGE

21630 The *charmap* definition is optional, and is contained outside the locale definition. This allows
21631 both completely self-defined source files, and generic sources (applicable to more than one
21632 codeset). To aid portability, all *charmap* definitions must use the same symbolic names for the
21633 portable character set. As explained in the Base Definitions volume of IEEE Std 1003.1-2001,
21634 Section 6.4, Character Set Description File, it is implementation-defined whether or not users or
21635 applications can provide additional character set description files. Therefore, the **-f** option might
21636 be operable only when an implementation-defined *charmap* is named.

21637 EXAMPLES

21638 None.

21639 RATIONALE

21640 The output produced by the *localedef* utility is implementation-defined. The *name* operand is
21641 used to identify the specific locale. (As a consequence, although several categories can be
21642 processed in one execution, only categories belonging to the same locale can be processed.)

21643 FUTURE DIRECTIONS

21644 None.

21645 SEE ALSO

21646 *locale*, the Base Definitions volume of IEEE Std 1003.1-2001, Section 7.3, Locale Definition

21647 CHANGE HISTORY

21648 First released in Issue 4.

21649 Issue 6

21650 The **-u** option is added, as specified in the IEEE P1003.2b draft standard.

21651 The normative text is reworded to avoid use of the term “must” for application requirements.

21652 IEEE Std 1003.1-2001/Cor 2-2004, item XCU/TC2/D6/15 is applied, rewording text in the
21653 OPERANDS section describing the ability to create public locales. 2

21654 IEEE Std 1003.1-2001/Cor 2-2004, item XCU/TC2/D6/16 is applied, making the text consistent
21655 with the descriptions of **WIDTH** and **WIDTH_DEFAULT** in the Base Definitions volume of
21656 IEEE Std 1003.1-2001. 2

21657 NAME

21658 **logger** — log messages

21659 SYNOPSIS

21660 **logger** *string* ...

21661 DESCRIPTION

21662 The *logger* utility saves a message, in an unspecified manner and format, containing the *string* operands provided by the user. The messages are expected to be evaluated later by personnel performing system administration tasks.

21665 It is implementation-defined whether messages written in locales other than the POSIX locale are effective.

21667 OPTIONS

21668 None.

21669 OPERANDS

21670 The following operand shall be supported:

21671 *string* One of the string arguments whose contents are concatenated together, in the order specified, separated by single <space>s.

21673 STDIN

21674 Not used.

21675 INPUT FILES

21676 None.

21677 ENVIRONMENT VARIABLES

21678 The following environment variables shall affect the execution of *logger*:

21679 **LANG** Provide a default value for the internationalization variables that are unset or null. (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2, Internationalization Variables for the precedence of internationalization variables used to determine the values of locale categories.)

21683 **LC_ALL** If set to a non-empty string value, override the values of all the other internationalization variables.

21685 **LC_CTYPE** Determine the locale for the interpretation of sequences of bytes of text data as characters (for example, single-byte as opposed to multi-byte characters in arguments).

21688 LC_MESSAGES

21689 Determine the locale that should be used to affect the format and contents of diagnostic messages written to standard error. (This means diagnostics from *logger* to the user or application, not diagnostic messages that the user is sending to the system administrator.)

21693 XSI **NLSPATH** Determine the location of message catalogs for the processing of *LC_MESSAGES*.

21694 ASYNCHRONOUS EVENTS

21695 Default.

21696 STDOUT

21697 Not used.

21698 STDERR

21699 The standard error shall be used only for diagnostic messages.

21700 OUTPUT FILES

21701 Unspecified.

21702 EXTENDED DESCRIPTION

21703 None.

21704 EXIT STATUS

21705 The following exit values shall be returned:

21706 0 Successful completion.

21707 >0 An error occurred.

21708 CONSEQUENCES OF ERRORS

21709 Default.

21710 APPLICATION USAGE

21711 This utility allows logging of information for later use by a system administrator or programmer
21712 in determining why non-interactive utilities have failed. The locations of the saved messages,
21713 their format, and retention period are all unspecified. There is no method for a conforming
21714 application to read messages, once written.

21715 EXAMPLES

21716 A batch application, running non-interactively, tries to read a configuration file and fails; it may
21717 attempt to notify the system administrator with:

21718 `logger myname: unable to read file foo. [timestamp]`

21719 RATIONALE

21720 The standard developers believed strongly that some method of alerting administrators to errors
21721 was necessary. The obvious example is a batch utility, running non-interactively, that is unable
21722 to read its configuration files or that is unable to create or write its results file. However, the
21723 standard developers did not wish to define the format or delivery mechanisms as they have
21724 historically been (and will probably continue to be) very system-specific, as well as involving
21725 functionality clearly outside the scope of this volume of IEEE Std 1003.1-2001.

21726 The text with *LC_MESSAGES* about diagnostic messages means diagnostics from *logger* to the
21727 user or application, not diagnostic messages that the user is sending to the system administrator.

21728 Multiple *string* arguments are allowed, similar to *echo*, for ease-of-use.

21729 Like the utilities *mailx* and *lp*, *logger* is admittedly difficult to test. This was not deemed sufficient
21730 justification to exclude these utilities from this volume of IEEE Std 1003.1-2001. It is also
21731 arguable that they are, in fact, testable, but that the tests themselves are not portable.

21732 FUTURE DIRECTIONS

21733 None.

21734 SEE ALSO

21735 *lp*, *mailx*, *write*

21736 CHANGE HISTORY

21737 First released in Issue 4.

21738 NAME

21739 *logname* — return the user's login name

21740 SYNOPSIS

21741 *logname*

21742 DESCRIPTION

21743 The *logname* utility shall write the user's login name to standard output. The login name shall be
21744 the string that would be returned by the *getlogin()* function defined in the System Interfaces
21745 volume of IEEE Std 1003.1-2001. Under the conditions where the *getlogin()* function would fail,
21746 the *logname* utility shall write a diagnostic message to standard error and exit with a non-zero
21747 exit status.

21748 OPTIONS

21749 None.

21750 OPERANDS

21751 None.

21752 STDIN

21753 Not used.

21754 INPUT FILES

21755 None.

21756 ENVIRONMENT VARIABLES

21757 The following environment variables shall affect the execution of *logname*:

21758 *LANG* Provide a default value for the internationalization variables that are unset or null.
21759 (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2,
21760 Internationalization Variables for the precedence of internationalization variables
21761 used to determine the values of locale categories.)

21762 *LC_ALL* If set to a non-empty string value, override the values of all the other
21763 internationalization variables.

21764 *LC_CTYPE* Determine the locale for the interpretation of sequences of bytes of text data as
21765 characters (for example, single-byte as opposed to multi-byte characters in
21766 arguments).

21767 *LC_MESSAGES*

21768 Determine the locale that should be used to affect the format and contents of
21769 diagnostic messages written to standard error.

21770 XSI *NLSPATH* Determine the location of message catalogs for the processing of *LC_MESSAGES*.

21771 ASYNCHRONOUS EVENTS

21772 Default.

21773 STDOUT

21774 The *logname* utility output shall be a single line consisting of the user's login name:

21775 "%s\n", <*login name*>

21776 STDERR

21777 The standard error shall be used only for diagnostic messages.

21778 OUTPUT FILES

21779 None.

21780 EXTENDED DESCRIPTION

21781 None.

21782 EXIT STATUS

21783 The following exit values shall be returned:

21784 0 Successful completion.

21785 >0 An error occurred.

21786 CONSEQUENCES OF ERRORS

21787 Default.

21788 APPLICATION USAGE

21789 The *logname* utility explicitly ignores the *LOGNAME* environment variable because environment changes could produce erroneous results.

21791 EXAMPLES

21792 None.

21793 RATIONALE

21794 The *passwd* file is not listed as required because the implementation may have other means of mapping login names.

21796 FUTURE DIRECTIONS

21797 None.

21798 SEE ALSO

21799 *id*, *who*, the System Interfaces volume of IEEE Std 1003.1-2001, *getlogin()*

21800 CHANGE HISTORY

21801 First released in Issue 2.

21802 NAME

21803 lp — send files to a printer

21804 SYNOPSIS

21805 lp [-c][-d *dest*][-n *copies*][-msw][-o *option*]... [-t *title*][*file*...]

21806 DESCRIPTION

21807 The *lp* utility shall copy the input files to an output destination in an unspecified manner. The
21808 default output destination should be to a hardcopy device, such as a printer or microfilm
21809 recorder, that produces non-volatile, human-readable documents. If such a device is not
21810 available to the application, or if the system provides no such device, the *lp* utility shall exit with
21811 a non-zero exit status.

21812 The actual writing to the output device may occur some time after the *lp* utility successfully
21813 exits. During the portion of the writing that corresponds to each input file, the implementation
21814 shall guarantee exclusive access to the device.

21815 The *lp* utility shall associate a unique *request ID* with each request.

21816 Normally, a banner page is produced to separate and identify each print job. This page may be
21817 suppressed by implementation-defined conditions, such as an operator command or one of the
21818 **-o option** values.

21819 OPTIONS

21820 The *lp* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section 12.2,
21821 Utility Syntax Guidelines.

21822 The following options shall be supported:

21823 **-c** Exit only after further access to any of the input files is no longer required. The
21824 application can then safely delete or modify the files without affecting the output
21825 operation. Normally, files are not copied, but are linked whenever possible. If the
21826 **-c** option is not given, then the user should be careful not to remove any of the
21827 files before the request has been printed in its entirety. It should also be noted that
21828 in the absence of the **-c** option, any changes made to the named files after the
21829 request is made but before it is printed may be reflected in the printed output. On
21830 some implementations, **-c** may be on by default.

21831 **-d dest** Specify a string that names the destination (*dest*). If *dest* is a printer, the request
21832 shall be printed only on that specific printer. If *dest* is a class of printers, the request
21833 shall be printed on the first available printer that is a member of the class. Under
21834 certain conditions (printer unavailability, file space limitation, and so on), requests
21835 for specific destinations need not be accepted. Destination names vary between
21836 systems.

21837 If **-d** is not specified, and neither the *LPDEST* nor *PRINTER* environment variable
21838 is set, an unspecified destination is used. The **-d dest** option shall take precedence
21839 over *LPDEST*, which in turn shall take precedence over *PRINTER*. Results are
21840 undefined when *dest* contains a value that is not a valid destination name.

21841 **-m** Send mail (see *mailx*) after the files have been printed. By default, no mail is sent
21842 upon normal completion of the print request.

21843 **-n copies** Write *copies* number of copies of the files, where *copies* is a positive decimal integer.
21844 The methods for producing multiple copies and for arranging the multiple copies
21845 when multiple *file* operands are used are unspecified, except that each file shall be
21846 output as an integral whole, not interleaved with portions of other files.

21847	-o <i>option</i>	Specify printer-dependent or class-dependent <i>options</i> . Several such <i>options</i> may be collected by specifying the -o option more than once.
21848		
21849	-s	Suppress messages from <i>lp</i> .
21850	-t <i>title</i>	Write <i>title</i> on the banner page of the output.
21851	-w	Write a message on the user's terminal after the files have been printed. If the user is not logged in, then mail shall be sent instead.
21852		

21853 OPERANDS

21854 The following operand shall be supported:

21855	<i>file</i>	A pathname of a file to be output. If no <i>file</i> operands are specified, or if a <i>file</i> operand is ' - ', the standard input shall be used. If a <i>file</i> operand is used, but the -c option is not specified, the process performing the writing to the output device may have user and group permissions that differ from that of the process invoking <i>lp</i> .
21856		
21857		
21858		
21859		

21860 STDIN

21861 The standard input shall be used only if no *file* operands are specified, or if a *file* operand is '**-**'.
21862 See the INPUT FILES section.

21863 INPUT FILES

21864 The input files shall be text files.

21865 ENVIRONMENT VARIABLES

21866 The following environment variables shall affect the execution of *lp*:

21867	LANG	Provide a default value for the internationalization variables that are unset or null. (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2, Internationalization Variables for the precedence of internationalization variables used to determine the values of locale categories.)
21868		
21869		
21870		
21871	LC_ALL	If set to a non-empty string value, override the values of all the other internationalization variables.
21872		
21873	LC_CTYPE	Determine the locale for the interpretation of sequences of bytes of text data as characters (for example, single-byte as opposed to multi-byte characters in arguments and input files).
21874		
21875		
21876	LC_MESSAGES	Determine the locale that should be used to affect the format and contents of diagnostic messages written to standard error and informative messages written to standard output.
21877		
21878		
21879		
21880	LC_TIME	Determine the format and contents of date and time strings displayed in the <i>lp</i> banner page, if any.
21881		
21882	LPDEST	Determine the destination. If the <i>LPDEST</i> environment variable is not set, the <i>PRINTER</i> environment variable shall be used. The -d <i>dest</i> option takes precedence over <i>LPDEST</i> . Results are undefined when -d is not specified and <i>LPDEST</i> contains a value that is not a valid destination name.
21883		
21884		
21885		
21886 XSI	NLSPATH	Determine the location of message catalogs for the processing of <i>LC_MESSAGES</i> .
21887	PRINTER	Determine the output device or destination. If the <i>LPDEST</i> and <i>PRINTER</i> environment variables are not set, an unspecified output device is used. The -d <i>dest</i> option and the <i>LPDEST</i> environment variable shall take precedence over <i>PRINTER</i> . Results are undefined when -d is not specified, <i>LPDEST</i> is unset, and
21888		
21889		
21890		

- 21891 `PRINTER` contains a value that is not a valid device or destination name.
- 21892 **TZ** Determine the timezone used to calculate date and time strings displayed in the *lp* banner page, if any. If *TZ* is unset or null, an unspecified default timezone shall be used.
- 21895 **ASYNCHRONOUS EVENTS**
- 21896 Default.
- 21897 **STDOUT**
- 21898 The *lp* utility shall write a *request ID* to the standard output, unless *-s* is specified. The format of the message is unspecified. The request ID can be used on systems supporting the historical *cancel* and *lpstat* utilities.
- 21901 **STDERR**
- 21902 The standard error shall be used only for diagnostic messages.
- 21903 **OUTPUT FILES**
- 21904 None.
- 21905 **EXTENDED DESCRIPTION**
- 21906 None.
- 21907 **EXIT STATUS**
- 21908 The following exit values shall be returned:
- 21909 0 All input files were processed successfully.
- 21910 >0 No output device was available, or an error occurred.
- 21911 **CONSEQUENCES OF ERRORS**
- 21912 Default.
- 21913 **APPLICATION USAGE**
- 21914 The *pr* and *fold* utilities can be used to achieve reasonable formatting for the implementation's default page size.
- 21916 A conforming application can use one of the *file* operands only with the *-c* option or if the file is publicly readable and guaranteed to be available at the time of printing. This is because IEEE Std 1003.1-2001 gives the implementation the freedom to queue up the request for printing at some later time by a different process that might not be able to access the file.
- 21920 **EXAMPLES**
- 21921 1. To print file *file*:
- 21922 *lp -c file*
- 21923 2. To print multiple files with headers:
- 21924 *pr file1 file2 | lp*
- 21925 **RATIONALE**
- 21926 The *lp* utility was designed to be a basic version of a utility that is already available in many historical implementations. The standard developers considered that it should be implementable simply as:
- 21929 *cat "\$@" > /dev/lp*
- 21930 after appropriate processing of options, if that is how the implementation chose to do it and if exclusive access could be granted (so that two users did not write to the device simultaneously).
- 21931 Although in the future the standard developers may add other options to this utility, it should

21933 always be able to execute with no options or operands and send the standard input to an
21934 unspecified output device.

21935 This volume of IEEE Std 1003.1-2001 makes no representations concerning the format of the
21936 printed output, except that it must be “human-readable” and “non-volatile”. Thus, writing by
21937 default to a disk or tape drive or a display terminal would not qualify. (Such destinations are not
21938 prohibited when **-d dest**, **LPDEST**, or **PRINTER** are used, however.)

21939 This volume of IEEE Std 1003.1-2001 is worded such that a “print job” consisting of multiple
21940 input files, possibly in multiple copies, is guaranteed to print so that any one file is not
21941 intermixed with another, but there is no statement that all the files or copies have to print out
21942 together.

21943 The **-c** option may imply a spooling operation, but this is not required. The utility can be
21944 implemented to wait until the printer is ready and then wait until it is finished. Because of that,
21945 there is no attempt to define a queuing mechanism (priorities, classes of output, and so on).

21946 On some historical systems, the request ID reported on the **STDOUT** can be used to later cancel
21947 or find the status of a request using utilities not defined in this volume of IEEE Std 1003.1-2001.

21948 Although the historical System V *lp* and BSD *lpr* utilities have provided similar functionality,
21949 they used different names for the environment variable specifying the destination printer. Since
21950 the name of the utility here is *lp*, **LPDEST** (used by the System V *lp* utility) was given precedence
21951 over **PRINTER** (used by the BSD *lpr* utility). Since environments of users frequently contain one
21952 or the other environment variable, the *lp* utility is required to recognize both. If this was not
21953 done, many applications would send output to unexpected output devices when users moved
21954 from system to system.

21955 Some have commented that *lp* has far too little functionality to make it worthwhile. Requests
21956 have proposed additional options or operands or both that added functionality. The requests
21957 included:

- 21958 • Wording *requiring* the output to be “hardcopy”
- 21959 • A requirement for multiple printers
- 21960 • Options for supporting various page-description languages

21961 Given that a compliant system is not required to even have a printer, placing further restrictions
21962 upon the behavior of the printer is not useful. Since hardcopy format is so application-
21963 dependent, it is difficult, if not impossible, to select a reasonable subset of functionality that
21964 should be required on all compliant systems.

21965 The term *unspecified* is used in this section in lieu of *implementation-defined* as most known
21966 implementations would not be able to make definitive statements in their conformance
21967 documents; the existence and usage of printers is very dependent on how the system
21968 administrator configures each individual system.

21969 Since the default destination, device type, queuing mechanisms, and acceptable forms of input
21970 are all *unspecified*, usage guidelines for what a conforming application can do are as follows:

- 21971 • Use the command in a pipeline, or with **-c**, so that there are no permission problems and the
21972 files can be safely deleted or modified.
- 21973 • Limit output to text files of reasonable line lengths and printable characters and include no
21974 device-specific formatting information, such as a page description language. The meaning of
21975 “reasonable” in this context can only be answered as a quality-of-implementation issue, but
21976 it should be apparent from historical usage patterns in the industry and the locale. The *pr* and
21977 *fold* utilities can be used to achieve reasonable formatting for the default page size of the

21978 implementation.

21979 Alternatively, the application can arrange its installation in such a way that it requires the
21980 system administrator or operator to provide the appropriate information on *lp* options and
21981 environment variable values.

21982 At a minimum, having this utility in this volume of IEEE Std 1003.1-2001 tells the industry that
21983 conforming applications require a means to print output and provides at least a command name
21984 and *LPDEST* routing mechanism that can be used for discussions between vendors, application
21985 writers, and users. The use of “should” in the DESCRIPTION of *lp* clearly shows the intent of
21986 the standard developers, even if they cannot mandate that all systems (such as laptops) have
21987 printers.

21988 This volume of IEEE Std 1003.1-2001 does not specify what the ownership of the process
21989 performing the writing to the output device may be. If **-c** is not used, it is unspecified whether
21990 the process performing the writing to the output device has permission to read *file* if there are
21991 any restrictions in place on who may read *file* until after it is printed. Also, if **-c** is not used, the
21992 results of deleting *file* before it is printed are unspecified.

21993 FUTURE DIRECTIONS

21994 None.

21995 SEE ALSO

21996 *mailx*

21997 CHANGE HISTORY

21998 First released in Issue 2.

21999 Issue 6

22000 The following new requirements on POSIX implementations derive from alignment with the
22001 Single UNIX Specification:

- 22002 • In the DESCRIPTION, the requirement to associate a unique request ID, and the normal
22003 generation of a banner page is added.
- 22004 • In the OPTIONS section:
 - 22005 — The **-d** *dest* description is expanded, but references to *lpstat* are removed.
 - 22006 — The **-m**, **-o**, **-s**, **-t**, and **-w** options are added.
- 22007 • In the ENVIRONMENT VARIABLES section, *LC_TIME* may now affect the execution.
- 22008 • The STDOUT section is added.

22009 The normative text is reworded to avoid use of the term “must” for application requirements.

22010 The *TZ* entry is added to the ENVIRONMENT VARIABLES section.

22011 NAME

22012 ls — list directory contents

22013 SYNOPSIS

22014 XSI ls [-CFRacdilqrtu1][-H | -L][-fgmnopsx][file...]

22015 DESCRIPTION

22016 For each operand that names a file of a type other than directory or symbolic link to a directory,
 22017 *ls* shall write the name of the file as well as any requested, associated information. For each
 22018 operand that names a file of type directory, *ls* shall write the names of files contained within the
 22019 directory as well as any requested, associated information. If one of the **-d**, **-F**, or **-l** options are
 22020 specified, and one of the **-H** or **-L** options are not specified, for each operand that names a file of
 22021 type symbolic link to a directory, *ls* shall write the name of the file as well as any requested,
 22022 associated information. If none of the **-d**, **-F**, or **-l** options are specified, or the **-H** or **-L** options
 22023 are specified, for each operand that names a file of type symbolic link to a directory, *ls* shall write
 22024 the names of files contained within the directory as well as any requested, associated
 22025 information.

22026 If no operands are specified, *ls* shall write the contents of the current directory. If more than one
 22027 operand is specified, *ls* shall write non-directory operands first; it shall sort directory and non-
 22028 directory operands separately according to the collating sequence in the current locale.

22029 The *ls* utility shall detect infinite loops; that is, entering a previously visited directory that is an
 22030 ancestor of the last file encountered. When it detects an infinite loop, *ls* shall write a diagnostic
 22031 message to standard error and shall either recover its position in the hierarchy or terminate.

22032 OPTIONS

22033 The *ls* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section 12.2,
 22034 Utility Syntax Guidelines.

22035 The following options shall be supported:

- 22036 **-C** Write multi-text-column output with entries sorted down the columns, according
 22037 to the collating sequence. The number of text columns and the column separator
 22038 characters are unspecified, but should be adapted to the nature of the output
 22039 device.
- 22040 **-F** Do not follow symbolic links named as operands unless the **-H** or **-L** options are
 22041 specified. Write a slash ('/') immediately after each pathname that is a directory,
 22042 an asterisk ('*') after each that is executable, a vertical bar ('|') after each that is
 22043 a FIFO, and an at sign ('@') after each that is a symbolic link. For other file types,
 22044 other symbols may be written.
- 22045 **-H** If a symbolic link referencing a file of type directory is specified on the command
 22046 line, *ls* shall evaluate the file information and file type to be those of the file
 22047 referenced by the link, and not the link itself; however, *ls* shall write the name of
 22048 the link itself and not the file referenced by the link.
- 22049 **-L** Evaluate the file information and file type for all symbolic links (whether named
 22050 on the command line or encountered in a file hierarchy) to be those of the file
 22051 referenced by the link, and not the link itself; however, *ls* shall write the name of
 22052 the link itself and not the file referenced by the link. When **-L** is used with **-l**, write
 22053 the contents of symbolic links in the long format (see the STDOUT section).
- 22054 **-R** Recursively list subdirectories encountered.
- 22055 **-a** Write out all directory entries, including those whose names begin with a period
 22056 ('.'). Entries beginning with a period shall not be written out unless explicitly

22057		referenced, the -a option is supplied, or an implementation-defined condition shall cause them to be written.
22059	-c	Use time of last modification of the file status information (see <sys/stat.h> in the System Interfaces volume of IEEE Std 1003.1-2001) instead of last modification of the file itself for sorting (-t) or writing (-l).
22062	-d	Do not follow symbolic links named as operands unless the -H or -L options are specified. Do not treat directories differently than other types of files. The use of -d with -R produces unspecified results.
22065 XSI	-f	Force each argument to be interpreted as a directory and list the name found in each slot. This option shall turn off -l , -t , -s , and -r , and shall turn on -a ; the order is the order in which entries appear in the directory.
22068 XSI	-g	The same as -l , except that the owner shall not be written.
22069	-i	For each file, write the file's file serial number (see stat() in the System Interfaces volume of IEEE Std 1003.1-2001).
22071	-l	(The letter ell.) Do not follow symbolic links named as operands unless the -H or -L options are specified. Write out in long format (see the STDOUT section). When -l (ell) is specified, -1 (one) shall be assumed.
22074 XSI	-m	Stream output format; list files across the page, separated by commas.
22075 XSI	-n	The same as -l , except that the owner's UID and GID numbers shall be written, rather than the associated character strings.
22077 XSI	-o	The same as -l , except that the group shall not be written.
22078 XSI	-p	Write a slash (' / ') after each filename if that file is a directory.
22079	-q	Force each instance of non-printable filename characters and <tab> s to be written as the question-mark (' ? ') character. Implementations may provide this option by default if the output is to a terminal device.
22082	-r	Reverse the order of the sort to get reverse collating sequence or oldest first.
22083 XSI	-s	Indicate the total number of file system blocks consumed by each file displayed. The block size is implementation-defined.
22085	-t	Sort with the primary key being time modified (most recently modified first) and the secondary key being filename in the collating sequence.
22087	-u	Use time of last access (see <sys/stat.h>) instead of last modification of the file for sorting (-t) or writing (-l).
22089 XSI	-x	The same as -C , except that the multi-text-column output is produced with entries sorted across, rather than down, the columns.
22091	-1	(The numeric digit one.) Force output to be one entry per line.
22092		Specifying more than one of the options in the following mutually-exclusive pairs shall not be considered an error: -C and -l (ell), -m and -l (ell), -x and -l (ell), -C and -1 (one), -H and -L , -c and -u . The last option specified in each pair shall determine the output format.

22095 OPERANDS

22096 The following operand shall be supported:

22097 *file* A pathname of a file to be written. If the file specified is not found, a diagnostic message shall be output on standard error.

22099 **STDIN**

22100 Not used.

22101 **INPUT FILES**

22102 None.

22103 **ENVIRONMENT VARIABLES**22104 The following environment variables shall affect the execution of *ls*:

22105 **COLUMNS** Determine the user's preferred column position width for writing multiple text-column output. If this variable contains a string representing a decimal integer, the *ls* utility shall calculate how many pathname text columns to write (see **-C**) based on the width provided. If **COLUMNS** is not set or invalid, an implementation-defined number of column positions shall be assumed, based on the implementation's knowledge of the output device. The column width chosen to write the names of files in any given directory shall be constant. Filenames shall not be truncated to fit into the multiple text-column output.

22113 **LANG** Provide a default value for the internationalization variables that are unset or null. (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2, Internationalization Variables for the precedence of internationalization variables used to determine the values of locale categories.)

22117 **LC_ALL** If set to a non-empty string value, override the values of all the other internationalization variables.

22119 **LC_COLLATE** Determine the locale for character collation information in determining the pathname collation sequence.

22122 **LC_CTYPE** Determine the locale for the interpretation of sequences of bytes of text data as characters (for example, single-byte as opposed to multi-byte characters in arguments) and which characters are defined as printable (character class **print**).

22125 **LC_MESSAGES** Determine the locale that should be used to affect the format and contents of diagnostic messages written to standard error.

22128 **LC_TIME** Determine the format and contents for date and time strings written by *ls*.

22129 **XSI** **NLSPATH** Determine the location of message catalogs for the processing of **LC_MESSAGES**.

22130 **TZ** Determine the timezone for date and time strings written by *ls*. If **TZ** is unset or null, an unspecified default timezone shall be used.

22132 **ASYNCHRONOUS EVENTS**

22133 Default.

22134 **STDOUT**

22135 The default format shall be to list one entry per line to standard output; the exceptions are to terminals or when one of the **-C**, **-m**, or **-x** options is specified. If the output is to a terminal, the format is implementation-defined.

22138 **XSI** When **-m** is specified, the format used shall be:

22139 "%s, %s, ... \n", <filename1>, <filename2>

22140 where the largest number of filenames shall be written without exceeding the length of the line.

22141 If the **-i** option is specified, the file's file serial number (see **<sys/stat.h>**) shall be written in the following format before any other output for the corresponding entry:

22143 %u ", <file serial number>

22144 If the **-l** option is specified without **-L**, the following information shall be written:

22145 "%s %u %s %s %u %s %s\n", <file mode>, <number of links>,
 22146 <owner name>, <group name>, <number of bytes in the file>,
 22147 <date and time>, <pathname>

22148 If the file is a symbolic link, this information shall be about the link itself and the <pathname>
 22149 field shall be of the form:

22150 "%s -> %s", <pathname of link>, <contents of link>

22151 If both **-l** and **-L** are specified, the following information shall be written:

22152 "%s %u %s %s %u %s %s\n", <file mode>, <number of links>,
 22153 <owner name>, <group name>, <number of bytes in the file>,
 22154 <date and time>, <pathname of link>

22155 where all fields except <pathname of link> shall be for the file resolved from the symbolic link.

22156 XSI The **-g**, **-n**, and **-o** options use the same format as **-l**, but with omitted items and their
 22157 associated <blank>s. See the OPTIONS section.

22158 XSI In both the preceding **-l** forms, if <owner name> or <group name> cannot be determined, or if **-n**
 22159 is given, they shall be replaced with their associated numeric values using the format %u.

22160 The <date and time> field shall contain the appropriate date and timestamp of when the file was
 22161 last modified. In the POSIX locale, the field shall be the equivalent of the output of the following
 22162 *date* command:

22163 date "+%b %e %H:%M"

22164 if the file has been modified in the last six months, or:

22165 date "+%b %e %Y"

22166 (where two <space>s are used between %e and %Y) if the file has not been modified in the last six
 22167 months or if the modification date is in the future, except that, in both cases, the final <newline>
 22168 produced by *date* shall not be included and the output shall be as if the *date* command were
 22169 executed at the time of the last modification date of the file rather than the current time. When
 22170 the *LC_TIME* locale category is not set to the POSIX locale, a different format and order of
 22171 presentation of this field may be used.

22172 If the file is a character special or block special file, the size of the file may be replaced with
 22173 implementation-defined information associated with the device in question.

22174 If the pathname was specified as a *file* operand, it shall be written as specified.

22175 XSI The file mode written under the **-l**, **-g**, **-n**, and **-o** options shall consist of the following format:

22176 "%c%s%s%s%c", <entry type>, <owner permissions>,
 22177 <group permissions>, <other permissions>,
 22178 <optional alternate access method flag>

22179 The <optional alternate access method flag> shall be a single <space> if there is no alternate or
 22180 additional access control method associated with the file; otherwise, a printable character shall
 22181 be used.

22182 The <entry type> character shall describe the type of file, as follows:

22183 d Directory.

22184 b Block special file.
 22185 c Character special file.
 22186 l (ell) Symbolic link.
 22187 p FIFO.
 22188 – Regular file.

22189 Implementations may add other characters to this list to represent other implementation-defined
 22190 file types.

22191 The next three fields shall be three characters each:

22192 <*owner permissions*> Permissions for the file owner class (see the Base Definitions volume of
 22193 IEEE Std 1003.1-2001, Section 4.4, File Access Permissions).

22195 <*group permissions*> Permissions for the file group class.

22197 <*other permissions*> Permissions for the file other class.

22199 Each field shall have three character positions:

- 22200 1. If 'r', the file is readable; if '–', the file is not readable.
- 22201 2. If 'w', the file is writable; if '–', the file is not writable.
- 22202 3. The first of the following that applies:

22203 S If in <*owner permissions*>, the file is not executable and set-user-ID mode is set. If in
 22204 <*group permissions*>, the file is not executable and set-group-ID mode is set.
 22205 s If in <*owner permissions*>, the file is executable and set-user-ID mode is set. If in
 22206 <*group permissions*>, the file is executable and set-group-ID mode is set.

22207 XSI T If in <*other permissions*> and the file is a directory, search permission is not granted to
 22208 others, and the restricted deletion flag is set.

22209 XSI t If in <*other permissions*> and the file is a directory, search permission is granted to
 22210 others, and the restricted deletion flag is set.

22211 x The file is executable or the directory is searchable.

22212 – None of the attributes of 'S', 's', 'T', 't', or 'x' applies.

22213 Implementations may add other characters to this list for the third character position. Such
 22214 additions shall, however, be written in lowercase if the file is executable or searchable, and
 22215 in uppercase if it is not.

22216 XSI If any of the **-l**, **-g**, **-n**, **-o**, or **-s** options is specified, each list of files within the directory shall be
 22217 preceded by a status line indicating the number of file system blocks occupied by files in the
 22218 directory in 512-byte units, rounded up to the next integral number of units, if necessary. In the
 22219 POSIX locale, the format shall be:

22220 "total %u\n", <number of units in the directory>

22221 If more than one directory, or a combination of non-directory files and directories are written,
 22222 either as a result of specifying multiple operands, or the **-R** option, each list of files within a
 22223 directory shall be preceded by:

22224 "\\n%*s*:\\n", <directory name>

22225 If this string is the first thing to be written, the first <newline> shall not be written. This output
22226 shall precede the number of units in the directory.

22227 XSI If the **-s** option is given, each file shall be written with the number of blocks used by the file.
22228 Along with **-C**, **-1**, **-m**, or **-x**, the number and a <space> shall precede the filename; with **-g**, **-l**,
22229 **-n**, or **-o**, they shall precede each line describing a file.

22230 STDERR

22231 The standard error shall be used only for diagnostic messages.

22232 OUTPUT FILES

22233 None.

22234 EXTENDED DESCRIPTION

22235 None.

22236 EXIT STATUS

22237 The following exit values shall be returned:

22238 0 Successful completion.

22239 >0 An error occurred.

22240 CONSEQUENCES OF ERRORS

22241 Default.

22242 APPLICATION USAGE

22243 Many implementations use the equal sign ('=') to denote sockets bound to the file system for
22244 the **-F** option. Similarly, many historical implementations use the 's' character to denote
22245 sockets as the entry type characters for the **-I** option.

22246 It is difficult for an application to use every part of the file modes field of *ls -l* in a portable
22247 manner. Certain file types and executable bits are not guaranteed to be exactly as shown, as
22248 implementations may have extensions. Applications can use this field to pass directly to a user
22249 printout or prompt, but actions based on its contents should generally be deferred, instead, to
22250 the *test* utility.

22251 The output of *ls* (with the **-l** and related options) contains information that logically could be
22252 used by utilities such as *chmod* and *touch* to restore files to a known state. However, this
22253 information is presented in a format that cannot be used directly by those utilities or be easily
22254 translated into a format that can be used. A character has been added to the end of the
22255 permissions string so that applications at least have an indication that they may be working in
22256 an area they do not understand instead of assuming that they can translate the permissions
22257 string into something that can be used. Future issues or related documents may define one or
22258 more specific characters to be used based on different standard additional or alternative access
22259 control mechanisms.

22260 As with many of the utilities that deal with filenames, the output of *ls* for multiple files or in one
22261 of the long listing formats must be used carefully on systems where filenames can contain
22262 embedded white space. Systems and system administrators should institute policies and user
22263 training to limit the use of such filenames.

22264 The number of disk blocks occupied by the file that it reports varies depending on underlying
22265 file system type, block size units reported, and the method of calculating the number of blocks.
22266 On some file system types, the number is the actual number of blocks occupied by the file
22267 (counting indirect blocks and ignoring holes in the file); on others it is calculated based on the
22268 file size (usually making an allowance for indirect blocks, but ignoring holes).

22269 EXAMPLES

22270 An example of a small directory tree being fully listed with *ls -laRF a* in the POSIX locale:

```
22271      total 11
22272      drwxr-xr-x    3 hlj      prog        64 Jul  4 12:07 .
22273      drwxrwxrwx    4 hlj      prog        3264 Jul  4 12:09 ..
22274      drwxr-xr-x    2 hlj      prog        48 Jul  4 12:07 b/
22275      -rwxr--r--    1 hlj      prog        572 Jul  4 12:07 foo*
22276
22277      a/b:
22278      total 4
22279      drwxr-xr-x    2 hlj      prog        48 Jul  4 12:07 .
22280      drwxr-xr-x    3 hlj      prog        64 Jul  4 12:07 ..
22281      -rw-r--r--    1 hlj      prog        700 Jul  4 12:07 bar
```

22281 RATIONALE

22282 Some historical implementations of the *ls* utility show all entries in a directory except dot and
 22283 dot-dot when a superuser invokes *ls* without specifying the **-a** option. When “normal” users
 22284 invoke *ls* without specifying **-a**, they should not see information about any files with names
 22285 beginning with a period unless they were named as *file* operands.

22286 Implementations are expected to traverse arbitrary depths when processing the **-R** option. The
 22287 only limitation on depth should be based on running out of physical storage for keeping track of
 22288 untraversed directories.

22289 The **-1** (one) option was historically found in BSD and BSD-derived implementations only. It is
 22290 required in this volume of IEEE Std 1003.1-2001 so that conforming applications might ensure
 22291 that output is one entry per line, even if the output is to a terminal.

22292 Generally, this volume of IEEE Std 1003.1-2001 is silent about what happens when options are
 22293 given multiple times. In the cases of **-C**, **-l**, and **-1**, however, it does specify the results of these
 22294 overlapping options. Since *ls* is one of the most aliased commands, it is important that the
 22295 implementation perform intuitively. For example, if the alias were:

```
22296 alias ls="ls -C"
```

22297 and the user typed *ls -1*, single-text-column output should result, not an error.

22298 The BSD *ls* provides a **-A** option (like **-a**, but dot and dot-dot are not written out). The small
 22299 difference from **-a** did not seem important enough to require both.

22300 Implementations may make **-q** the default for terminals to prevent trojan horse attacks on
 22301 terminals with special escape sequences. This is not required because:

- 22302 • Some control characters may be useful on some terminals; for example, a system might write
 22303 them as "\001" or "^A".
- 22304 • Special behavior for terminals is not relevant to applications portability.

22305 An early proposal specified that the optional alternate access method flag had to be '+' if there
 22306 was an alternate access method used on the file or <space> if there was not. This was changed to
 22307 be <space> if there is not and a single printable character if there is. This was done for three
 22308 reasons:

- 22309 1. There are historical implementations using characters other than '+'.
- 22310 2. There are implementations that vary this character used in that position to distinguish
 22311 between various alternate access methods in use.

22312 3. The standard developers did not want to preclude future specifications that might need a
22313 way to specify more than one alternate access method.

22314 Nonetheless, implementations providing a single alternate access method are encouraged to use
22315 '+'.

22316 In an early proposal, the units used to specify the number of blocks occupied by files in a
22317 directory in an *ls -l* listing were implementation-defined. This was because BSD systems have
22318 historically used 1 024-byte units and System V systems have historically used 512-byte units. It
22319 was pointed out by BSD developers that their system has used 512-byte units in some places and
22320 1 024-byte units in other places. (System V has consistently used 512.) Therefore, this volume of
22321 IEEE Std 1003.1-2001 usually specifies 512. Future releases of BSD are expected to consistently
22322 provide 512 bytes as a default with a way of specifying 1 024-byte units where appropriate.

22323 The *<date and time>* field in the *-l* format is specified only for the POSIX locale. As noted, the
22324 format can be different in other locales. No mechanism for defining this is present in this volume
22325 of IEEE Std 1003.1-2001, as the appropriate vehicle is a messaging system; that is, the format
22326 should be specified as a "message".

22327 FUTURE DIRECTIONS

22328 The *-s* uses implementation-defined units and cannot be used portably; it may be withdrawn in
22329 a future version.

22330 SEE ALSO

22331 *chmod*, *find*, the System Interfaces volume of IEEE Std 1003.1-2001, *stat()*, the Base Definitions
22332 volume of IEEE Std 1003.1-2001, *<sys/stat.h>*

22333 CHANGE HISTORY

22334 First released in Issue 2.

22335 Issue 5

22336 A second FUTURE DIRECTION is added.

22337 Issue 6

22338 The following new requirements on POSIX implementations derive from alignment with the
22339 Single UNIX Specification:

- In the *-F* option, other symbols are allowed for other file types.

22341 Treatment of symbolic links is added, as defined in the IEEE P1003.2b draft standard.

22342 The Open Group Base Resolution bwg2001-010 is applied, adding the *T* and *t* fields as an XSI
22343 extension.

22344 NAME

22345 m4 — macro processor (**DEVELOPMENT**)

22346 SYNOPSIS

22347 XSI m4 [-s][-D name[=val]]...[-U name]... file...

22348

22349 DESCRIPTION

22350 The *m4* utility is a macro processor that shall read one or more text files, process them according
22351 to their included macro statements, and write the results to standard output.

22352 OPTIONS

22353 The *m4* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section 12.2,
22354 Utility Syntax Guidelines, except that the order of the **-D** and **-U** options shall be significant.

22355 The following options shall be supported:

22356 **-s** Enable line synchronization output for the *c99* preprocessor phase (that is, #line
22357 directives).22358 **-D name[=val]**
22359 Define *name* to *val* or to null if =*val* is omitted.22360 **-U name** Undefine *name*.

22361 OPERANDS

22362 The following operand shall be supported:

22363 *file* A pathname of a text file to be processed. If no *file* is given, or if it is ‘-’, the
22364 standard input shall be read.

22365 STDIN

22366 The standard input shall be a text file that is used if no *file* operand is given, or if it is ‘-’.

22367 INPUT FILES

22368 The input file named by the *file* operand shall be a text file.

22369 ENVIRONMENT VARIABLES

22370 The following environment variables shall affect the execution of *m4*:22371 **LANG** Provide a default value for the internationalization variables that are unset or null.
22372 (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2,
22373 Internationalization Variables for the precedence of internationalization variables
22374 used to determine the values of locale categories.)22375 **LC_ALL** If set to a non-empty string value, override the values of all the other
22376 internationalization variables.22377 **LC_CTYPE** Determine the locale for the interpretation of sequences of bytes of text data as
22378 characters (for example, single-byte as opposed to multi-byte characters in
22379 arguments and input files).22380 **LC_MESSAGES**22381 Determine the locale that should be used to affect the format and contents of
22382 diagnostic messages written to standard error.22383 **NLSPATH** Determine the location of message catalogs for the processing of *LC_MESSAGES*.

22384 ASYNCHRONOUS EVENTS

22385 Default.

22386 STDOUT

22387 The standard output shall be the same as the input files, after being processed for macro
22388 expansion.

22389 STDERR

22390 The standard error shall be used to display strings with the **errprint** macro, macro tracing
22391 enabled by the **traceon** macro, the defined text for macros written by the **dumpdef** macro, or for
22392 diagnostic messages.

22393 OUTPUT FILES

22394 None.

22395 EXTENDED DESCRIPTION

22396 The *m4* utility shall compare each token from the input against the set of built-in and user-
22397 defined macros. If the token matches the name of a macro, then the token shall be replaced by
22398 the macro's defining text, if any, and rescanned for matching macro names. Once no portion of
22399 the token matches the name of a macro, it shall be written to standard output. Macros may have
22400 arguments, in which case the arguments shall be substituted into the defining text before it is
22401 rescanned.

22402 Macro calls have the form:

22403 *name*(*arg1*, *arg2*, ..., *argn*)22404 Macro names shall consist of letters, digits, and underscores, where the first character is not a
22405 digit. Tokens not of this form shall not be treated as macros.22406 The application shall ensure that the left parenthesis immediately follows the name of the
22407 macro. If a token matching the name of a macro is not followed by a left parenthesis, it is
22408 handled as a use of that macro without arguments.22409 If a macro name is followed by a left parenthesis, its arguments are the comma-separated tokens
22410 between the left parenthesis and the matching right parenthesis. Unquoted <blank>s and
22411 <newline>s preceding each argument shall be ignored. All other characters, including trailing
22412 <blank>s and <newline>s, are retained. Commas enclosed between left and right parenthesis
22413 characters do not delimit arguments.22414 Arguments are positionally defined and referenced. The string "\$1" in the defining text shall be
22415 replaced by the first argument. Systems shall support at least nine arguments; only the first nine
22416 can be referenced, using the strings "\$1" to "\$9", inclusive. The string "\$0" is replaced with
22417 the name of the macro. The string "\$#" is replaced by the number of arguments as a string. The
22418 string "\$*" is replaced by a list of all of the arguments, separated by commas. The string "\$@"
22419 is replaced by a list of all of the arguments separated by commas, and each argument is quoted
22420 using the current left and right quoting strings.22421 If fewer arguments are supplied than are in the macro definition, the omitted arguments are
22422 taken to be null. It is not an error if more arguments are supplied than are in the macro
22423 definition.22424 No special meaning is given to any characters enclosed between matching left and right quoting
22425 strings, but the quoting strings are themselves discarded. By default, the left quoting string
22426 consists of a grave accent ('') and the right quoting string consists of an acute accent ('');
22427 see also the **changequote** macro.22428 Comments are written but not scanned for matching macro names; by default, the begin-
22429 comment string consists of the number sign character and the end-comment string consists of a

22430	<newline>. See also the changeom and dnl macros.
22431	The <i>m4</i> utility shall make available the following built-in macros. They can be redefined, but once this is done the original meaning is lost. Their values shall be null unless otherwise stated.
22432	In the descriptions below, the term <i>defining text</i> refers to the value of the macro: the second argument to the define macro, among other things. Except for the first argument to the eval macro, all numeric arguments to built-in macros shall be interpreted as decimal values. The string values produced as the defining text of the decr , divnum , incr , index , len , and sysval built-in macros shall be in the form of a decimal-constant as defined in the C language.
22433	
22434	
22435	
22436	
22437	
22438	changeom The changeom macro shall set the begin-comment and end-comment strings.
22439	With no arguments, the comment mechanism shall be disabled. With a single argument, that argument shall become the begin-comment string and the <newline> shall become the end-comment string. With two arguments, the first argument shall become the begin-comment string and the second argument shall become the end-comment string. Systems shall support comment strings of at least five characters.
22440	
22441	
22442	
22443	
22444	
22445	changequote The changequote macro shall set the begin-quote and end-quote strings. With no arguments, the quote strings shall be set to the default values (that is, ``'). With a single argument, that argument shall become the begin-quote string and the <newline> shall become the end-quote string. With two arguments, the first argument shall become the begin-quote string and the second argument shall become the end-quote string. Systems shall support quote strings of at least five characters.
22446	
22447	
22448	
22449	
22450	
22451	
22452	decr The defining text of the decr macro shall be its first argument decremented by 1. It shall be an error to specify an argument containing any non-numeric characters.
22453	
22454	define The second argument shall become the defining text of the macro whose name is the first argument.
22455	
22456	defn The defining text of the defn macro shall be the quoted definition (using the current quoting strings) of its arguments.
22457	
22458	divert The <i>m4</i> utility maintains nine temporary buffers, numbered 1 to 9, inclusive. When the last of the input has been processed, any output that has been placed in these buffers shall be written to standard output in buffer-numerical order. The divert macro shall divert future output to the buffer specified by its argument. Specifying no argument or an argument of 0 shall resume the normal output process. Output diverted to a stream other than 0 to 9 shall be discarded. It shall be an error to specify an argument containing any non-numeric characters.
22459	
22460	
22461	
22462	
22463	
22464	
22465	divnum The defining text of the divnum macro shall be the number of the current output stream as a string.
22466	
22467	dnl The dnl macro shall cause <i>m4</i> to discard all input characters up to and including the next <newline>.
22468	
22469	dumpdef The dumpdef macro shall write the defined text to standard error for each of the macros specified as arguments, or, if no arguments are specified, for all macros.
22470	
22471	errprint The errprint macro shall write its arguments to standard error.
22472	eval The eval macro shall evaluate its first argument as an arithmetic expression, using 32-bit signed integer arithmetic. All of the C-language operators shall be supported, except for:
22473	
22474	

```

22475      [ ]
22476      ->
22477      ++
22478      --
22479      ( type )
22480      unary *
22481      sizeof
22482      ,
22483      .
22484      ?: 
22485      unary &

```

and all assignment operators. It shall be an error to specify any of these operators. Precedence and associativity shall be as in the ISO C standard. Systems shall support octal and hexadecimal numbers as in the ISO C standard. The second argument, if specified, shall set the radix for the result; the default is 10. The third argument, if specified, sets the minimum number of digits in the result. It shall be an error to specify the second or third argument containing any non-numeric characters.

22493 ifdef If the first argument to the **ifdef** macro is defined, the defining text shall be the second argument. Otherwise, the defining text shall be the third argument, if specified, or the null string, if not.

22496 ifelse The **ifelse** macro takes three or more arguments. If the first two arguments compare as equal strings (after macro expansion of both arguments), the defining text shall be the third argument. If the first two arguments do not compare as equal strings and there are three arguments, the defining text shall be null. If the first two arguments do not compare as equal strings and there are four or five arguments, the defining text shall be the fourth argument. If the first two arguments do not compare as equal strings and there are six or more arguments, the first three arguments shall be discarded and processing shall restart with the remaining arguments.

22505 include The defining text for the **include** macro shall be the contents of the file named by the first argument. It shall be an error if the file cannot be read.

22507 incr The defining text of the **incr** macro shall be its first argument incremented by 1. It shall be an error to specify an argument containing any non-numeric characters.

22509 index The defining text of the **index** macro shall be the first character position (as a string) in the first argument where a string matching the second argument begins (zero origin), or -1 if the second argument does not occur.

22512 len The defining text of the **len** macro shall be the length (as a string) of the first argument.

22514 m4exit Exit from the **m4** utility. If the first argument is specified, it is the exit code. The default is zero. It shall be an error to specify an argument containing any non-numeric characters.

22517 m4wrap The first argument shall be processed when EOF is reached. If the **m4wrap** macro is used multiple times, the arguments specified shall be processed in the order in which the **m4wrap** macros were processed.

22520 maketemp The defining text shall be the first argument, with any trailing 'x' characters replaced with the current process ID as a string.

22522	popdef	The popdef macro shall delete the current definition of its arguments, replacing that definition with the previous one. If there is no previous definition, the macro is undefined.
22523		
22524		
22525	pushdef	The pushdef macro shall be equivalent to the define macro with the exception that it shall preserve any current definition for future retrieval using the popdef macro.
22526		
22527	shift	The defining text for the shift macro shall be all of its arguments except for the first one.
22528		
22529	sinclude	The sinclude macro shall be equivalent to the include macro, except that it shall not be an error if the file is inaccessible.
22530		
22531	substr	The defining text for the substr macro shall be the substring of the first argument beginning at the zero-offset character position specified by the second argument. The third argument, if specified, shall be the number of characters to select; if not specified, the characters from the starting point to the end of the first argument shall become the defining text. It shall not be an error to specify a starting point beyond the end of the first argument and the defining text shall be null. It shall be an error to specify an argument containing any non-numeric characters.
22532		
22533		
22534		
22535		
22536		
22537		
22538	syscmd	The syscmd macro shall interpret its first argument as a shell command line. The defining text shall be the string result of that command. No output redirection shall be performed by the <i>m4</i> utility. The exit status value from the command can be retrieved using the sysval macro.
22539		
22540		
22541		
22542	sysval	The defining text of the sysval macro shall be the exit value of the utility last invoked by the syscmd macro (as a string).
22543		
22544	traceon	The traceon macro shall enable tracing for the macros specified as arguments, or, if no arguments are specified, for all macros. The trace output shall be written to standard error in an unspecified format.
22545		
22546		
22547	traceoff	The traceoff macro shall disable tracing for the macros specified as arguments, or, if no arguments are specified, for all macros.
22548		
22549	translit	The defining text of the translit macro shall be the first argument with every character that occurs in the second argument replaced with the corresponding character from the third argument.
22550		
22551		
22552	undefine	The undefine macro shall delete all definitions (including those preserved using the pushdef macro) of the macros named by its arguments.
22553		
22554	undivert	The undivert macro shall cause immediate output of any text in temporary buffers named as arguments, or all temporary buffers if no arguments are specified. Buffers can be undiverted into other temporary buffers. Undiverting shall discard the contents of the temporary buffer. It shall be an error to specify an argument containing any non-numeric characters.
22555		
22556		
22557		
22558		
22559	EXIT STATUS	
22560		The following exit values shall be returned:
22561	0	Successful completion.
22562	>0	An error occurred
22563		If the m4exit macro is used, the exit value can be specified by the input file.

22564 CONSEQUENCES OF ERRORS

22565 Default.

22566 APPLICATION USAGE

22567 The **defn** macro is useful for renaming macros, especially built-ins.

22568 EXAMPLES

22569 If the file **m4src** contains the lines:

```
22570     The value of 'VER' is "VER".  
22571     ifdef('VER', ''VER'' is defined to be VER., VER is not defined.)  
22572     ifelse(VER, 1, ''VER'' is 'VER'.)  
22573     ifelse(VER, 2, ''VER'' is 'VER'', ''VER'' is not 2.)  
22574     end
```

22575 then the command

```
22576 m4 m4src
```

22577 or the command:

```
22578 m4 -U VER m4src
```

22579 produces the output:

```
22580     The value of VER is "VER".  
22581     VER is not defined.
```

```
22582     VER is not 2.  
22583     end
```

22584 The command:

```
22585 m4 -D VER m4src
```

22586 produces the output:

```
22587     The value of VER is "".  
22588     VER is defined to be .
```

```
22589     VER is not 2.  
22590     end
```

22591 The command:

```
22592 m4 -D VER=1 m4src
```

22593 produces the output:

```
22594     The value of VER is "1".  
22595     VER is defined to be 1.  
22596     VER is 1.  
22597     VER is not 2.  
22598     end
```

22599 The command:

```
22600 m4 -D VER=2 m4src
```

22601 produces the output:

```
22602     The value of VER is "2".  
22603     VER is defined to be 2.
```

22604	VER is 2.	1
22605	end	1
22606	RATIONALE	
22607	None.	
22608	FUTURE DIRECTIONS	
22609	None.	
22610	SEE ALSO	
22611	<i>c99</i>	
22612	CHANGE HISTORY	
22613	First released in Issue 2.	
22614	Issue 5	
22615	The phrase “the defined text for macros written by the dumpdef macro” is added to the	
22616	description of STDERR, and the description of dumpdef is updated to indicate that output is	
22617	written to standard error. The description of eval is updated to indicate that the list of excluded	
22618	C operators excludes unary ‘&’ and ‘.’. In the description of ifdef , the phrase “and it is not	
22619	defined to be zero” is deleted.	
22620	Issue 6	
22621	In the EXTENDED DESCRIPTION, the eval text is updated to include a ‘&’ character in the	
22622	excepted list.	
22623	The EXTENDED DESCRIPTION of divert is updated to clarify that there are only nine diversion	
22624	buffers.	
22625	The normative text is reworded to avoid use of the term “must” for application requirements.	
22626	The Open Group Base Resolution bwg2000-006 is applied.	
22627	IEEE Std 1003.1-2001/Cor 1-2002, item XCU/TC1/D6/31 is applied, replacing the EXAMPLES	1
22628	section.	1

22629 **NAME**

22630 mailx — process messages

22631 **SYNOPSIS**

22632 **Send Mode**

22633 mailx [-s *subject*] *address...*

22634 **Receive Mode**

22635 mailx -e

22636 mailx [-HiNn][-F][-u *user*]

22637 mailx -f[-HiNn][-F][*file*]

22638 **DESCRIPTION**

22639 The *mailx* utility provides a message sending and receiving facility. It has two major modes, selected by the options used: Send Mode and Receive Mode.

22641 On systems that do not support the User Portability Utilities option, an application using *mailx* shall have the ability to send messages in an unspecified manner (Send Mode). Unless the first character of one or more lines is tilde ('~'), all characters in the input message shall appear in the delivered message, but additional characters may be inserted in the message before it is retrieved.

22646 On systems supporting the User Portability Utilities option, mail-receiving capabilities and other interactive features, Receive Mode, described below, also shall be enabled.

22648 **Send Mode**

22649 Send Mode can be used by applications or users to send messages from the text in standard input.

22651 **Receive Mode**

22652 Receive Mode is more oriented towards interactive users. Mail can be read and sent in this interactive mode.

22654 When reading mail, *mailx* provides commands to facilitate saving, deleting, and responding to messages. When sending mail, *mailx* allows editing, reviewing, and other modification of the message as it is entered.

22657 Incoming mail shall be stored in one or more unspecified locations for each user, collectively called the system *mailbox* for that user. When *mailx* is invoked in Receive Mode, the system mailbox shall be the default place to find new mail. As messages are read, they shall be marked to be moved to a secondary file for storage, unless specific action is taken. This secondary file is called the **mbox** and is normally located in the directory referred to by the *HOME* environment variable (see *MBOX* in the ENVIRONMENT VARIABLES section for a description of this file). Messages shall remain in this file until explicitly removed. When the -f option is used to read mail messages from secondary files, messages shall be retained in those files unless specifically removed. All three of these locations—system mailbox, **mbox**, and secondary file—are referred to in this section as simply “mailboxes”, unless more specific identification is required.

22667 **OPTIONS**

- 22668 The *mailx* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section
 22669 12.2, Utility Syntax Guidelines.
- 22670 The following options shall be supported. (Only the **-s** *subject* option shall be required on all
 22671 systems. The other options are required only on systems supporting the User Portability Utilities
 22672 option.)
- 22673 **-e** Test for the presence of mail in the system mailbox. The *mailx* utility shall write
 22674 nothing and exit with a successful return code if there is mail to read.
- 22675 **-f** Read messages from the file named by the *file* operand instead of the system
 22676 mailbox. (See also **folder**.) If no *file* operand is specified, read messages from **mbox**
 22677 instead of the system mailbox.
- 22678 **-F** Record the message in a file named after the first recipient. The name is the login-
 22679 name portion of the address found first on the **To:** line in the mail header.
 22680 Overrides the **record** variable, if set (see **Internal Variables in mailx** (on page
 22681 593).)
- 22682 **-H** Write a header summary only.
- 22683 **-i** Ignore interrupts. (See also **ignore**.)
- 22684 **-n** Do not initialize from the system default start-up file. See the EXTENDED
 22685 DESCRIPTION section.
- 22686 **-N** Do not write an initial header summary.
- 22687 **-s** *subject* Set the **Subject** header field to *subject*. All characters in the *subject* string shall
 22688 appear in the delivered message. The results are unspecified if *subject* is longer
 22689 than {LINE_MAX} – 10 bytes or contains a <newline>.
- 22690 **-u** *user* Read the system mailbox of the login name *user*. This shall only be successful if
 22691 the invoking user has the appropriate privileges to read the system mailbox of that
 22692 user.

22693 **OPERANDS**

22694 The following operands shall be supported:

- 22695 **address** Addressee of message. When **-n** is specified and no user start-up files are accessed
 22696 (see the EXTENDED DESCRIPTION section), the user or application shall ensure
 22697 this is an address to pass to the mail delivery system. Any system or user start-up
 22698 files may enable aliases (see **alias** under **Commands in mailx** (on page 596)) that
 22699 may modify the form of **address** before it is passed to the mail delivery system.
- 22700 **file** A pathname of a file to be read instead of the system mailbox when **-f** is specified.
 22701 The meaning of the *file* option-argument shall be affected by the contents of the
 22702 **folder** internal variable; see **Internal Variables in mailx** (on page 593).

22703 **STDIN**

22704 When *mailx* is invoked in Send Mode (the first synopsis line), standard input shall be the
 22705 message to be delivered to the specified addresses. When in Receive Mode, user commands shall
 22706 be accepted from *stdin*. If the User Portability Utilities option is not supported, standard input
 22707 lines beginning with a tilde ('~') character produce unspecified results.

22708 If the User Portability Utilities option is supported, then in both Send and Receive Modes,
 22709 standard input lines beginning with the escape character (usually tilde ('~')) shall affect
 22710 processing as described in **Command Escapes in mailx** (on page 604).

22711 INPUT FILES

22712 When *mailx* is used as described by this volume of IEEE Std 1003.1-2001, the *file* option-
22713 argument (see the **-f** option) and the **mbox** shall be text files containing mail messages,
22714 formatted as described in the OUTPUT FILES section. The nature of the system mailbox is
22715 unspecified; it need not be a file.

22716 ENVIRONMENT VARIABLES

22717 The following environment variables shall affect the execution of *mailx*:

22718	DEAD	Determine the pathname of the file in which to save partial messages in case of 22719 interrupts or delivery errors. The default shall be dead.letter in the directory 22720 named by the HOME variable. The behavior of <i>mailx</i> in saving partial messages is 22721 unspecified if the User Portability Utilities option is not supported and DEAD is 22722 not defined with the value /dev/null .
22723	EDITOR	Determine the name of a utility to invoke when the edit (see Commands in mailx 22724 (on page 596)) or ~e (see Command Escapes in mailx (on page 604)) command is 22725 XSI used. The default editor is unspecified. On XSI-conformant systems it is ed . The 22726 effects of this variable are unspecified if the User Portability Utilities option is not 22727 supported.
22728	HOME	Determine the pathname of the user's home directory.
22729	LANG	Provide a default value for the internationalization variables that are unset or null. 22730 (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2, 22731 Internationalization Variables for the precedence of internationalization variables 22732 used to determine the values of locale categories.)
22733	LC_ALL	If set to a non-empty string value, override the values of all the other 22734 internationalization variables.
22735	LC_CTYPE	Determine the locale for the interpretation of sequences of bytes of text data as 22736 characters (for example, single-byte as opposed to multi-byte characters in 22737 arguments and input files) and the handling of case-insensitive address and 22738 header-field comparisons.
22739	LC_TIME	Determine the format and contents of the date and time strings written by <i>mailx</i> .
22740	LC_MESSAGES	Determine the locale that should be used to affect the format and contents of 22741 diagnostic messages written to standard error and informative messages written to 22742 standard output.
22744	LISTER	Determine a string representing the command for writing the contents of the 22745 folder directory to standard output when the folders command is given (see 22746 folders in Commands in mailx (on page 596)). Any string acceptable as a 22747 command_string operand to the sh -c command shall be valid. If this variable is null 22748 or not set, the output command shall be ls . The effects of this variable are 22749 unspecified if the User Portability Utilities option is not supported.
22750	MAILRC	Determine the pathname of the start-up file. The default shall be .mailrc in the 22751 directory referred to by the HOME environment variable. The behavior of <i>mailx</i> is 22752 unspecified if the User Portability Utilities option is not supported and MAILRC is 22753 not defined with the value /dev/null .
22754	MBOX	Determine a pathname of the file to save messages from the system mailbox that 22755 have been read. The exit command shall override this function, as shall saving the 22756 message explicitly in another file. The default shall be mbox in the directory

22757	named by the <i>HOME</i> variable. The effects of this variable are unspecified if the User Portability Utilities option is not supported.
22758	
22759 XSI	NLSPATH Determine the location of message catalogs for the processing of <i>LC_MESSAGES</i> .
22760	
22761	PAGER Determine a string representing an output filtering or pagination command for writing the output to the terminal. Any string acceptable as a <i>command_string</i> operand to the <i>sh -c</i> command shall be valid. When standard output is a terminal device, the message output shall be piped through the command if the <i>mailx</i> internal variable <i>crt</i> is set to a value less than the number of lines in the message; see Internal Variables in mailx (on page 593). If the <i>PAGER</i> variable is null or not set, the paginator shall be either <i>more</i> or another paginator utility documented in the system documentation. The effects of this variable are unspecified if the User Portability Utilities option is not supported.
22762	
22763	
22764	
22765	
22766	
22767	
22768	
22769	SHELL Determine the name of a preferred command interpreter. The default shall be <i>sh</i> . The effects of this variable are unspecified if the User Portability Utilities option is not supported.
22770	
22771	
22772	TERM If the internal variable screen is not specified, determine the name of the terminal type to indicate in an unspecified manner the number of lines in a screenful of headers. If <i>TERM</i> is not set or is set to null, an unspecified default terminal type shall be used and the value of a screenful is unspecified. The effects of this variable are unspecified if the User Portability Utilities option is not supported.
22773	
22774	
22775	
22776	
22777	TZ This variable may determine the timezone used to calculate date and time strings written by <i>mailx</i> . If <i>TZ</i> is unset or null, an unspecified default timezone shall be used.
22778	
22779	
22780	VISUAL Determine a pathname of a utility to invoke when the visual command (see Commands in mailx (on page 596)) or ~v command-escape (see Command Escapes in mailx (on page 604)) is used. If this variable is null or not set, the full-screen editor shall be <i>vi</i> . The effects of this variable are unspecified if the User Portability Utilities option is not supported.
22781	
22782	
22783	
22784	

22785 ASYNCHRONOUS EVENTS

22786 When *mailx* is in Send Mode and standard input is not a terminal, it shall take the standard
 22787 action for all signals.

22788 In Receive Mode, or in Send Mode when standard input is a terminal, if a SIGINT signal is
 22789 received:

- 22790 1. If in command mode, the current command, if there is one, shall be aborted, and a
 22791 command-mode prompt shall be written.
- 22792 2. If in input mode:
 - 22793 a. If **ignore** is set, *mailx* shall write "@\n", discard the current input line, and continue
 22794 processing, bypassing the message-abort mechanism described in item 2b.
 - 22795 b. If the interrupt was received while sending mail, either when in Receive Mode or in
 22796 Send Mode, a message shall be written, and another subsequent interrupt, with no
 22797 other intervening characters typed, shall be required to abort the mail message. If in
 22798 Receive Mode and another interrupt is received, a command-mode prompt shall be
 22799 written. If in Send Mode and another interrupt is received, *mailx* shall terminate with
 22800 a non-zero status.

22801 In both cases listed in item b, if the message is not empty:

- 22802 i. If **save** is enabled and the file named by *DEAD* can be created, the message
22803 shall be written to the file named by *DEAD*. If the file exists, the message shall
22804 be written to replace the contents of the file.
22805 ii. If **save** is not enabled, or the file named by *DEAD* cannot be created, the
22806 message shall not be saved.

22807 The *mailx* utility shall take the standard action for all other signals.

22808 STDOUT

22809 In command and input modes, all output, including prompts and messages, shall be written to
22810 standard output.

22811 STDERR

22812 The standard error shall be used only for diagnostic messages.

22813 OUTPUT FILES

22814 Various *mailx* commands and command escapes can create or add to files, including the **mbox**,
22815 the dead-letter file, and secondary mailboxes. When *mailx* is used as described in this volume of
22816 IEEE Std 1003.1-2001, these files shall be text files, formatted as follows:

```
22817 line beginning with From<space>
22818 [one or more header-lines; see Commands in mailx (on page 596)]
22819 empty line
22820 [zero or more body lines
22821 empty line]
22822 [line beginning with From<space>...]
```

22823 where each message begins with the **From <space>** line shown, preceded by the beginning of
22824 the file or an empty line. (The **From <space>** line is considered to be part of the message header,
22825 but not one of the header-lines referred to in **Commands in mailx** (on page 596); thus, it shall not
22826 be affected by the **discard**, **ignore**, or **retain** commands.) The formats of the remainder of the
22827 **From <space>** line and any additional header lines are unspecified, except that none shall be
22828 empty. The format of a message body line is also unspecified, except that no line following an
22829 empty line shall start with **From <space>**; *mailx* shall modify any such user-entered message
22830 body lines (following an empty line and beginning with **From <space>**) by adding one or more
22831 characters to precede the '**F**'; it may add these characters to **From <space>** lines that are not
22832 preceded by an empty line.

22833 When a message from the system mailbox or entered by the user is not a text file, it is
22834 implementation-defined how such a message is stored in files written by *mailx*.

22835 EXTENDED DESCRIPTION

22836 The entire EXTENDED DESCRIPTION section shall apply only to implementations supporting
22837 the User Portability Utilities option.

22838 The *mailx* utility cannot guarantee support for all character encodings in all circumstances. For
22839 example, inter-system mail may be restricted to 7-bit data by the underlying network, 8-bit data
22840 need not be portable to non-internationalized systems, and so on. Under these circumstances, it
22841 is recommended that only characters defined in the ISO/IEC 646: 1991 standard International
22842 Reference Version (equivalent to ASCII) 7-bit range of characters be used.

22843 When *mailx* is invoked using one of the Receive Mode synopsis forms, it shall write a page of
22844 header-summary lines (if **-N** was not specified and there are messages, see below), followed by
22845 a prompt indicating that *mailx* can accept regular commands (see **Commands in mailx** (on page
22846 596)); this is termed *command mode*. The page of header-summary lines shall contain the first
22847 new message if there are new messages, or the first unread message if there are unread
22848 messages, or the first message. When *mailx* is invoked using the Send Mode synopsis and

standard input is a terminal, if no subject is specified on the command line and the **asksub** variable is set, a prompt for the subject shall be written. At this point, **mailx** shall be in input mode. This input mode shall also be entered when using one of the Receive Mode synopsis forms and a reply or new message is composed using the **reply**, **Reply**, **followup**, **Followup**, or **mail** commands and standard input is a terminal. When the message is typed and the end of the message is encountered, the message shall be passed to the mail delivery software. Commands can be entered by beginning a line with the escape character (by default, tilde ('~')) followed by a single command letter and optional arguments. See **Commands in mailx** (on page 596) for a summary of these commands. It is unspecified what effect these commands will have if standard input is not a terminal when a message is entered using either the Send Mode synopsis, or the Read Mode commands **reply**, **Reply**, **followup**, **Followup**, or **mail**.

Note: For notational convenience, this section uses the default escape character, tilde, in all references and examples.

At any time, the behavior of **mailx** shall be governed by a set of environmental and internal variables. These are flags and valued parameters that can be set and cleared via the **mailx set** and **unset** commands.

Regular commands are of the form:

[*command*] [*msglist*] [*argument* ...]

If no *command* is specified in command mode, **next** shall be assumed. In input mode, commands shall be recognized by the escape character, and lines not treated as commands shall be taken as input for the message.

In command mode, each message shall be assigned a sequential number, starting with 1.

All messages have a state that shall affect how they are displayed in the header summary and how they are retained or deleted upon termination of **mailx**. There is at any time the notion of a *current* message, which shall be marked by a '>' at the beginning of a line in the header summary. When **mailx** is invoked using one of the Receive Mode synopsis forms, the current message shall be the first new message, if there is a new message, or the first unread message if there is an unread message, or the first message if there are any messages, or unspecified if there are no messages in the mailbox. Each command that takes an optional list of messages (*msglist*) or an optional single message (*message*) on which to operate shall leave the current message set to the highest-numbered message of the messages specified, unless the command deletes messages, in which case the current message shall be set to the first undeleted message (that is, a message not in the deleted state) after the highest-numbered message deleted by the command, if one exists, or the first undeleted message before the highest-numbered message deleted by the command, if one exists, or to an unspecified value if there are no remaining undeleted messages. All messages shall be in one of the following states:

- | | |
|---------------------|--|
| 22885 new | The message is present in the system mailbox and has not been viewed by the user or moved to any other state. Messages in state <i>new</i> when mailx quits shall be retained in the system mailbox. |
| 22888 unread | The message has been present in the system mailbox for more than one invocation of mailx and has not been viewed by the user or moved to any other state. Messages in state <i>unread</i> when mailx quits shall be retained in the system mailbox. |
| 22891 read | The message has been processed by one of the following commands: ~f , ~m , ~F , ~M , copy , mbox , next , pipe , print , Print , top , type , Type , undelete . The delete , dp , and dt commands may also cause the next message to be marked as <i>read</i> , depending on the value of the autoprint variable. Messages that are in the system mailbox and in state <i>read</i> when mailx quits shall be saved in the mbox , unless the internal variable hold was set. Messages that are in the mbox or in a secondary mailbox and in state |

22897		<i>read</i> when <i>mailx</i> quits shall be retained in their current location.
22898	<i>deleted</i>	The message has been processed by one of the following commands: delete , dp , dt . Messages in state <i>deleted</i> when <i>mailx</i> quits shall be deleted. Deleted messages shall be ignored until <i>mailx</i> quits or changes mailboxes or they are specified to the undelete command; for example, the message specification <i>/string</i> shall only search the subject lines of messages that have not yet been deleted, unless the command operating on the list of messages is undelete . No deleted message or deleted message header shall be displayed by any <i>mailx</i> command other than undelete .
22906	<i>preserved</i>	The message has been processed by a preserve command. When <i>mailx</i> quits, the message shall be retained in its current location.
22908	<i>saved</i>	The message has been processed by one of the following commands: save or write . If the current mailbox is the system mailbox, and the internal variable keepsave is set, messages in the state <i>saved</i> shall be saved to the file designated by the <i>MBOX</i> variable (see the ENVIRONMENT VARIABLES section). If the current mailbox is the system mailbox, messages in the state <i>saved</i> shall be deleted from the current mailbox, when the quit or file command is used to exit the current mailbox.
22915		The header-summary line for each message shall indicate the state of the message.
22916		Many commands take an optional list of messages (<i>msglist</i>) on which to operate, which defaults to the current message. A <i>msglist</i> is a list of message specifications separated by <blank>s, which can include:
22919	n	Message number <i>n</i> .
22920	+	The next undeleted message, or the next deleted message for the undelete command.
22921	-	The next previous undeleted message, or the next previous deleted message for the undelete command.
22923	.	The current message.
22924	^	The first undeleted message, or the first deleted message for the undelete command.
22925	\$	The last message.
22926	*	All messages.
22927	<i>n-m</i>	An inclusive range of message numbers.
22928	<i>address</i>	All messages from <i>address</i> ; any address as shown in a header summary shall be matchable in this form.
22930	<i>/string</i>	All messages with <i>string</i> in the subject line (case ignored).
22931	:c	All messages of type <i>c</i> , where <i>c</i> shall be one of:
22932	d	Deleted messages.
22933	n	New messages.
22934	o	Old messages (any not in state <i>read</i> or <i>new</i>).
22935	r	Read messages.
22936	u	Unread messages.

22937 Other commands take an optional message (*message*) on which to operate, which defaults to the
22938 current message. All of the forms allowed for *msglist* are also allowed for *message*, but if more
22939 than one message is specified, only the first shall be operated on.

22940 Other arguments are usually arbitrary strings whose usage depends on the command involved.

22941 Start-Up in mailx

22942 At start-up time, *mailx* shall take the following steps in sequence:

- 22943 1. Establish all variables at their stated default values.
- 22944 2. Process command line options, overriding corresponding default values.
- 22945 3. Import any of the *DEAD*, *EDITOR*, *MBOX*, *LISTER*, *PAGER*, *SHELL*, or *VISUAL* variables
22946 that are present in the environment, overriding the corresponding default values.
- 22947 4. Read *mailx* commands from an unspecified system start-up file, unless the **-n** option is
22948 given, to initialize any internal *mailx* variables and aliases.
- 22949 5. Process the start-up file of *mailx* commands named in the user *MAILRC* variable.

22950 Most regular *mailx* commands are valid inside start-up files, the most common use being to set
22951 up initial display options and alias lists. The following commands shall be invalid in the start-up
22952 file: **!**, **edit**, **hold**, **mail**, **preserve**, **reply**, **Reply**, **shell**, **visual**, **Copy**, **followup**, and **Followup**.
22953 Any errors in the start-up file shall either cause *mailx* to terminate with a diagnostic message and
22954 a non-zero status or to continue after writing a diagnostic message, ignoring the remainder of
22955 the lines in the start-up file.

22956 A blank line in a start-up file shall be ignored.

22957 Internal Variables in mailx

22958 The following variables are internal *mailx* variables. Each internal variable can be set via the
22959 *mailx set* command at any time. The **unset** and **set no name** commands can be used to erase
22960 variables.

22961 In the following list, variables shown as:

22962 *variable*

22963 represent Boolean values. Variables shown as:

22964 *variable=value*

22965 shall be assigned string or numeric values. For string values, the rules in **Commands in mailx**
22966 (on page 596) concerning filenames and quoting shall also apply.

22967 The defaults specified here may be changed by the unspecified system start-up file unless the
22968 user specifies the **-n** option. 2 2

22969 **allnet** All network names whose login name components match shall be treated as
22970 identical. This shall cause the *msglist* message specifications to behave similarly.
22971 The default shall be **noallnet**. See also the **alternates** command and the **metoo**
22972 variable.

22973 **append** Append messages to the end of the **mbox** file upon termination instead of placing
22974 them at the beginning. The default shall be **noappend**. This variable shall not
22975 affect the **save** command when saving to **mbox**.

22976 **ask**, **asksub** Prompt for a subject line on outgoing mail if one is not specified on the command
22977 line with the **-s** option. The **ask** and **asksub** forms are synonyms; the system shall

22978		refer to asksub and noasksub in its messages, but shall accept ask and noask as user input to mean asksub and noasksub . It shall not be possible to set both ask and noasksub , or noask and asksub . The default shall be asksub , but no prompting shall be done if standard input is not a terminal.
22982	askbcc	Prompt for the blind copy list. The default shall be noaskbcc .
22983	askcc	Prompt for the copy list. The default shall be noaskcc .
22984	autoprint	Enable automatic writing of messages after delete and undelete commands. The default shall be noautoprint .
22986	bang	Enable the special-case treatment of exclamation marks (' ! ') in escape command lines; see the escape command and Command Escapes in mailx (on page 604). The default shall be nobang , disabling the expansion of ' ! ' in the <i>command</i> argument to the ~! command and the ~<!command escape.
22990	cmd=command	Set the default command to be invoked by the pipe command. The default shall be nocmd .
22993	crt=number	Pipe messages having more than <i>number</i> lines through the command specified by the value of the <i>PAGER</i> variable. The default shall be nocrt . If it is set to null, the value used is implementation-defined.
22996 XSI	debug	Enable verbose diagnostics for debugging. Messages are not delivered. The default shall be nodebug .
22998	dot	When dot is set, a period on a line by itself during message input from a terminal shall also signify end-of-file (in addition to normal end-of-file). The default shall be nodot . If ignoreeof is set (see below), a setting of nodot shall be ignored and the period is the only method to terminate input mode.
23002	escape=c	Set the command escape character to be the character 'c'. By default, the command escape character shall be tilde. If escape is unset, tilde shall be used; if it is set to null, command escaping shall be disabled.
23005	flipr	Reverse the meanings of the R and r commands. The default shall be noflipr .
23006	folder=directory	The default directory for saving mail files. User-specified filenames beginning with a plus sign ('+') shall be expanded by preceding the filename with this directory name to obtain the real pathname. If <i>directory</i> does not start with a slash (' / '), the contents of <i>HOME</i> shall be prefixed to it. The default shall be nofolder . If folder is unset or set to null, user-specified filenames beginning with '+' shall refer to files in the current directory that begin with the literal '+' character. See also outfolder below. The folder value need not affect the processing of the files named in <i>MBOX</i> and <i>DEAD</i> .
23015	header	Enable writing of the header summary when entering <i>mailx</i> in Receive Mode. The default shall be header .
23017	hold	Preserve all messages that are read in the system mailbox instead of putting them in the mbox save file. The default shall be nohold .
23019	ignore	Ignore interrupts while entering messages. The default shall be noignore .
23020	ignoreeof	Ignore normal end-of-file during message input. Input can be terminated only by entering a period (' . ') on a line by itself or by the ~. command escape. The default shall be noignoreeof . See also dot above.

23023	indentprefix=string	A string that shall be added as a prefix to each line that is inserted into the message by the ~m command escape. This variable shall default to one <tab>.
23024		
23025		
23026	keep	When a system mailbox, secondary mailbox, or mbox is empty, truncate it to zero length instead of removing it. The default shall be nokeep .
23027		
23028	keepsave	Keep the messages that have been saved from the system mailbox into other files in the file designated by the variable MBOX , instead of deleting them. The default shall be nokeepsave .
23029		
23030		
23031	metoo	Suppress the deletion of the login name of the user from the recipient list when replying to a message or sending to a group. The default shall be nometoo .
23032		
23033 XSI	onehop	When responding to a message that was originally sent to several recipients, the other recipient addresses are normally forced to be relative to the originating author's machine for the response. This flag disables alteration of the recipients' addresses, improving efficiency in a network where all machines can send directly to all other machines (that is, one hop away). The default shall be noonehop .
23034		
23035		
23036		
23037		
23038	outfolder	Cause the files used to record outgoing messages to be located in the directory specified by the folder variable unless the pathname is absolute. The default shall be nooutfolder . See the record variable.
23039		
23040		
23041	page	Insert a <form-feed> after each message sent through the pipe created by the pipe command. The default shall be nopage .
23042		
23043	prompt=string	Set the command-mode prompt to <i>string</i> . If <i>string</i> is null or if noprompt is set, no prompting shall occur. The default shall be to prompt with the string "? ".
23044		
23045		
23046	quiet	Refrain from writing the opening message and version when entering mailx . The default shall be noquiet .
23047		
23048	record=file	Record all outgoing mail in the file with the pathname <i>file</i> . The default shall be norecord . See also outfolder above.
23049		
23050	save	Enable saving of messages in the dead-letter file on interrupt or delivery error. See the variable DEAD for the location of the dead-letter file. The default shall be save .
23051		
23052	screen=number	Set the number of lines in a screenful of headers for the headers and z commands. If screen is not specified, a value based on the terminal type identified by the TERM environment variable, the window size, the baud rate, or some combination of these shall be used.
23053		
23054		
23055		
23056		
23057	sendwait	Wait for the background mailer to finish before returning. The default shall be nosendwait .
23058		
23059	showto	When the sender of the message was the user who is invoking mailx , write the information from the To: line instead of the From: line in the header summary. The default shall be noshowto .
23060		
23061		
23062	sign=string	Set the variable inserted into the text of a message when the ~a command escape is given. The default shall be nosign . The character sequences ' \t ' and ' \n ' shall be recognized in the variable as <tab>s and <newline>s, respectively. (See also ~i in Command Escapes in mailx (on page 604).)
23063		
23064		
23065		
23066	Sign=string	Set the variable inserted into the text of a message when the ~A command escape is given. The default shall be noSign . The character sequences ' \t ' and ' \n ' shall
23067		

23068 be recognized in the variable as <tab>s and <newline>s, respectively.

23069 **toplines=number**

23070 Set the number of lines of the message to write with the **top** command. The default

23071 shall be 5.

23072 **Commands in mailx**

23073 The following *mailx* commands shall be provided. In the following list, header refers to lines

23074 from the message header, as shown in the OUTPUT FILES section. Header-line refers to lines

23075 within the header that begin with one or more non-white-space characters, immediately

23076 followed by a colon and white space and continuing until the next line beginning with a non-

23077 white-space character or an empty line. Header-field refers to the portion of a header line prior

23078 to the first colon in that line.

23079 For each of the commands listed below, the command can be entered as the abbreviation (those

23080 characters in the Synopsis command word preceding the '['), the full command (all characters

23081 shown for the command word, omitting the '[' and ']'), or any truncation of the full

23082 command down to the abbreviation. For example, the **exit** command (shown as **ex[it]** in the

23083 Synopsis) can be entered as **ex**, **exi**, or **exit**.

23084 The arguments to commands can be quoted, using the following methods:

- 23085 • An argument can be enclosed between paired double-quotes (" ") or single-quotes (' '); any
23086 white space, shell word expansion, or backslash characters within the quotes shall be treated
23087 literally as part of the argument. A double-quote shall be treated literally within single-
23088 quotes and *vice versa*. These special properties of the quote marks shall occur only when they
23089 are paired at the beginning and end of the argument.
- 23090 • A backslash outside of the enclosing quotes shall be discarded and the following character
23091 treated literally as part of the argument.
- 23092 • An unquoted backslash at the end of a command line shall be discarded and the next line
23093 shall continue the command.

23094 Filenames, where expected, shall be subjected to the following transformations, in sequence: 1

- 23095 • If the filename begins with an unquoted plus sign, and the **folder** variable is defined (see the 1
23096 **folder** variable), the plus sign shall be replaced by the value of the **folder** variable followed 1
23097 by a slash. If the **folder** variable is unset or is set to null, the filename shall be unchanged. 1
- 23098 • Shell word expansions shall be applied to the filename (see Section 2.6 (on page 36)). If more 1
23099 than a single pathname results from this expansion and the command is expecting one file, 1
23100 the effects are unspecified. 1

23101 **Declare Aliases**

23102 *Synopsis:* **a[lias] [alias [address...]]**

23103 **g[roup] [alias [address...]]**

23104 Add the given addresses to the alias specified by *alias*. The names shall be substituted when

23105 *alias* is used as a recipient address specified by the user in an outgoing message (that is, other

23106 recipients addressed indirectly through the **reply** command shall not be substituted in this

23107 manner). Mail address alias substitution shall apply only when the alias string is used as a full

23108 address; for example, when **hlj** is an alias, **hlj@posix.com** does not trigger the alias substitution. If

23109 no arguments are given, write a listing of the current aliases to standard output. If only an *alias*

23110 argument is given, write a listing of the specified alias to standard output. These listings need

23111 not reflect the same order of addresses that were entered.

23112 **Declare Alternatives**

23113 *Synopsis:* `alt[ernates] name...`

23114 (See also the **metoo** command.) Declare a list of alternative names for the user's login. When
23115 responding to a message, these names shall be removed from the list of recipients for the
23116 response. The comparison of names shall be in a case-insensitive manner. With no arguments,
23117 **alternates** shall write the current list of alternative names.

23118 **Change Current Directory**

23119 *Synopsis:* `cd [directory]`
23120 `ch[dir] [directory]`

23121 Change directory. If *directory* is not specified, the contents of *HOME* shall be used.

23122 **Copy Messages**

23123 *Synopsis:* `c[opy] [file]`
23124 `c[opy] [msglist] file`
23125 `C[opy] [msglist]`

23126 Copy messages to the file named by the pathname *file* without marking the messages as saved.
23127 Otherwise, it shall be equivalent to the **save** command.

23128 In the capitalized form, save the specified messages in a file whose name is derived from the
23129 author of the message to be saved, without marking the messages as saved. Otherwise, it shall
23130 be equivalent to the **Save** command.

23131 **Delete Messages**

23132 *Synopsis:* `d[elete] [msglist]`

23133 Mark messages for deletion from the mailbox. The deletions shall not occur until *mailx* quits (see
23134 the **quit** command) or changes mailboxes (see the **folder** command). If **autoprint** is set and there
23135 are messages remaining after the **delete** command, the current message shall be written as
23136 described for the **print** command (see the **print** command); otherwise, the *mailx* prompt shall be
23137 written.

23138 **Discard Header Fields**

23139 *Synopsis:* `di[scard] [header-field...]`
23140 `ig[nore] [header-field...]`

23141 Suppress the specified header fields when writing messages. Specified *header-fields* shall be
23142 added to the list of suppressed header fields. Examples of header fields to ignore are **status** and
23143 **cc**. The fields shall be included when the message is saved. The **Print** and **Type** commands shall
23144 override this command. The comparison of header fields shall be in a case-insensitive manner. If
23145 no arguments are specified, write a list of the currently suppressed header fields to standard
23146 output; the listing need not reflect the same order of header fields that were entered.

23147 If both **retain** and **discard** commands are given, **discard** commands shall be ignored.

23148 **Delete Messages and Display**

23149 *Synopsis:* dp [msglist]
23150 dt [msglist]

23151 Delete the specified messages as described for the **delete** command, except that the **autoprint**
23152 variable shall have no effect, and the current message shall be written only if it was set to a
23153 message after the last message deleted by the command. Otherwise, an informational message
23154 to the effect that there are no further messages in the mailbox shall be written, followed by the
23155 **mailx** prompt.

23156 **Echo a String**

23157 *Synopsis:* ec[ho] string ...

23158 Echo the given strings, equivalent to the shell **echo** utility.

23159 **Edit Messages**

23160 *Synopsis:* e[dit] [msglist]

23161 Edit the given messages. The messages shall be placed in a temporary file and the utility named
23162 by the **EDITOR** variable is invoked to edit each file in sequence. The default **EDITOR** is
23163 unspecified.

23164 The **edit** command does not modify the contents of those messages in the mailbox.

23165 **Exit**

23166 *Synopsis:* ex[it]
23167 x[it]

23168 Exit from **mailx** without changing the mailbox. No messages shall be saved in the **mbox** (see also
23169 **quit**).

23170 **Change Folder**

23171 *Synopsis:* fi[le] [file]
23172 fold[er] [file]

23173 Quit (see the **quit** command) from the current file of messages and read in the file named by the
23174 pathname *file*. If no argument is given, the name and status of the current mailbox shall be
23175 written.

23176 Several unquoted special characters shall be recognized when used as *file* names, with the
23177 following substitutions:

23178 % The system mailbox for the invoking user.

23179 %user The system mailbox for *user*.

23180 # The previous file.

23181 & The current **mbox**.

23182 +file The named file in the **folder** directory. (See the **folder** variable.)

23183 The default file shall be the current mailbox.

23184 **Display List of Folders**

23185 *Synopsis:* folders

23186 Write the names of the files in the directory set by the **folder** variable. The command specified by
23187 the *LISTER* environment variable shall be used (see the ENVIRONMENT VARIABLES section).

23188 **Follow Up Specified Messages**

23189 *Synopsis:* fo[llowup] [message]
23190 F[ollowup] [msglist]

23191 In the lowercase form, respond to a message, recording the response in a file whose name is
23192 derived from the author of the message. See also the **save** and **copy** commands and **outfolder**.

23193 In the capitalized form, respond to the first message in the *msglist*, sending the message to the
23194 author of each message in the *msglist*. The subject line shall be taken from the first message and
23195 the response shall be recorded in a file whose name is derived from the author of the first
23196 message. See also the **Save** and **Copy** commands and **outfolder**.

23197 Both forms shall override the **record** variable, if set.

23198 **Display Header Summary for Specified Messages**

23199 *Synopsis:* f[rom] [msglist]

23200 Write the header summary for the specified messages.

23201 **Display Header Summary**

23202 *Synopsis:* h[eaders] [message]

23203 Write the page of headers that includes the message specified. If the *message* argument is not
23204 specified, the current message shall not change. However, if the *message* argument is specified,
23205 the current message shall become the message that appears at the top of the page of headers that
23206 includes the message specified. The **screen** variable sets the number of headers per page. See
23207 also the **z** command.

23208 **Help**

23209 *Synopsis:* hel[p]
23210 ?

23211 Write a summary of commands.

23212 **Hold Messages**

23213 *Synopsis:* ho[ld] [msglist]
23214 pre[serve] [msglist]

23215 Mark the messages in *msglist* to be retained in the mailbox when *mailx* terminates. This shall
23216 override any commands that might previously have marked the messages to be deleted. During
23217 the current invocation of *mailx*, only the **delete**, **dp**, or **dt** commands shall remove the *preserve*
23218 marking of a message.

23219 **Execute Commands Conditionally**

23220 *Synopsis:* *i[f] s|r
 mail-commands
 el[se]
 mail-commands
 en[dif]*

23225 Execute commands conditionally, where **if** *s* executes the following *mail-commands*, up to an
23226 **else** or **endif**, if the program is in Send Mode, and **if** *r* shall cause the *mail-commands* to be
23227 executed only in Receive Mode.

23228 **List Available Commands**

23229 *Synopsis:* *l[ist]*

23230 Write a list of all commands available. No explanation shall be given.

23231 **Mail a Message**

23232 *Synopsis:* *m[ail] address...*

23233 Mail a message to the specified addresses or aliases.

23234 **Direct Messages to mbox**

23235 *Synopsis:* *mb[ox] [msglist]*

23236 Arrange for the given messages to end up in the **mbox** save file when *mailx* terminates normally.
23237 See *MBOX*. See also the **exit** and **quit** commands.

23238 **Process Next Specified Message**

23239 *Synopsis:* *n[ext] [message]*

23240 If the current message has not been written (for example, by the **print** command) since *mailx*
23241 started or since any other message was the current message, behave as if the **print** command
23242 was entered. Otherwise, if there is an undeleted message after the current message, make it the
23243 current message and behave as if the **print** command was entered. Otherwise, an informational
23244 message to the effect that there are no further messages in the mailbox shall be written, followed
23245 by the *mailx* prompt.

23246 **Pipe Message**

23247 *Synopsis:* *pi[pe] [[msglist] command]
 | [[msglist] command]*

23249 Pipe the messages through the given *command* by invoking the command interpreter specified
23250 by *SHELL* with two arguments: **-c** and *command*. (See also *sh -c*.) The application shall ensure
23251 that the command is given as a single argument. Quoting, described previously, can be used to
23252 accomplish this. If no arguments are given, the current message shall be piped through the
23253 command specified by the value of the **cmd** variable. If the **page** variable is set, a <form-feed>
23254 shall be inserted after each message.

Display Message with Headers

23255 *Synopsis:* P[rint] [msglist]
23256 T[ype] [msglist]

23258 Write the specified messages, including all header lines, to standard output. Override
23259 suppression of lines by the **discard**, **ignore**, and **retain** commands. If **crt** is set, the messages
23260 longer than the number of lines specified by the **crt** variable shall be paged through the
23261 command specified by the *PAGER* environment variable.

Display Message

23262 *Synopsis:* p[rint] [msglist]
23263 t[ype] [msglist]

23264 Write the specified messages to standard output. If **crt** is set, the messages longer than the
23265 number of lines specified by the **crt** variable shall be paged through the command specified by
23266 the *PAGER* environment variable.

Quit

23267 *Synopsis:* q[uit]
23268 end-of-file

23269 Terminate **mailx**, storing messages that were read in **mbox** (if the current mailbox is the system
23270 mailbox and unless **hold** is set), deleting messages that have been explicitly saved (unless
23271 **keepsave** is set), discarding messages that have been deleted, and saving all remaining messages
23272 in the mailbox.

Reply to a Message List

23273 *Synopsis:* R[eply] [msglist]
23274 R[espond] [msglist]

23275 Mail a reply message to the sender of each message in the *msglist*. The subject line shall be
23276 formed by concatenating **Re:<space>** (unless it already begins with that string) and the subject
23277 from the first message. If **record** is set to a filename, the response shall be saved at the end of that
23278 file.

23279 See also the **flipr** variable.

Reply to a Message

23280 *Synopsis:* r[eply] [message]
23281 r[espond] [message]

23282 Mail a reply message to all recipients included in the header of the message. The subject line
23283 shall be formed by concatenating **Re:<space>** (unless it already begins with that string) and the
23284 subject from the message. If **record** is set to a filename, the response shall be saved at the end of
23285 that file.

23286 See also the **flipr** variable.

23291 **Retain Header Fields**

23292 *Synopsis:* ret[ain] [*header-field...*]

23293 Retain the specified header fields when writing messages. This command shall override all
23294 **discard** and **ignore** commands. The comparison of header fields shall be in a case-insensitive
23295 manner. If no arguments are specified, write a list of the currently retained header fields to
23296 standard output; the listing need not reflect the same order of header fields that were entered.

23297 **Save Messages**

23298 *Synopsis:* s[ave] [*file*]
23299 s[ave] [*msglist*] *file*
23300 S[ave] [*msglist*]

23301 Save the specified messages in the file named by the pathname *file*, or the **mbox** if the *file*
23302 argument is omitted. The file shall be created if it does not exist; otherwise, the messages shall be
23303 appended to the file. The message shall be put in the state *saved*, and shall behave as specified in
23304 the description of the *saved* state when the current mailbox is exited by the **quit** or **file**
23305 command.

23306 In the capitalized form, save the specified messages in a file whose name is derived from the
23307 author of the first message. The name of the file shall be taken to be the author's name with all
23308 network addressing stripped off. See also the **Copy**, **followup**, and **Followup** commands and
23309 **outfolder** variable.

23310 **Set Variables**

23311 *Synopsis:* se[t] [*name*[=[*string*]] ...] [*name=number* ...] [*noname* ...]

23312 Define one or more variables called *name*. The variable can be given a null, string, or numeric
23313 value. Quoting and backslash escapes can occur anywhere in *string*, as described previously, as
23314 if the *string* portion of the argument were the entire argument. The forms *name* and *name=* shall
23315 be equivalent to *name=""* for variables that take string values. The **set** command without
23316 arguments shall write a list of all defined variables and their values. The **no** *name* form shall be
23317 equivalent to **unset** *name*.

23318 **Invoke a Shell**

23319 *Synopsis:* sh[ell]

23320 Invoke an interactive command interpreter (see also **SHELL**).

23321 **Display Message Size**

23322 *Synopsis:* si[ze] [*msglist*]

23323 Write the size in bytes of each of the specified messages.

23324 **Read mailx Commands From a File**

23325 *Synopsis:* so[urce] *file*

23326 Read and execute commands from the file named by the pathname *file* and return to command
23327 mode.

23328 **Display Beginning of Messages**23329 *Synopsis:* `to[p] [msglist]`23330 Write the top few lines of each of the specified messages. If the **toplines** variable is set, it is taken
23331 as the number of lines to write. The default shall be 5.23332 **Touch Messages**23333 *Synopsis:* `tou[ch] [msglist]`23334 Touch the specified messages. If any message in *msglist* is not specifically deleted nor saved in a
23335 file, it shall be placed in the **mbox** upon normal termination. See **exit** and **quit**.23336 **Delete Aliases**23337 *Synopsis:* `una[lias] [alias]...`

23338 Delete the specified alias names. If a specified alias does not exist, the results are unspecified.

23339 **Undelete Messages**23340 *Synopsis:* `u[ndelete] [msglist]`23341 Change the state of the specified messages from deleted to read. If **autoprint** is set, the last
23342 message of those restored shall be written. If *msglist* is not specified, the message shall be
23343 selected as follows:

- 23344 • If there are any deleted messages that follow the current message, the first of these shall be
23345 chosen.
- 23346 • Otherwise, the last deleted message that also precedes the current message shall be chosen.

23347 **Unset Variables**23348 *Synopsis:* `uns[et] name...`

23349 Cause the specified variables to be erased.

23350 **Edit Message with Full-Screen Editor**23351 *Synopsis:* `v[isual] [msglist]`23352 Edit the given messages with a screen editor. Each message shall be placed in a temporary file,
23353 and the utility named by the **VISUAL** variable shall be invoked to edit each file in sequence. The
23354 default editor shall be **vi**.23355 The **visual** command does not modify the contents of those messages in the mailbox.23356 **Write Messages to a File**23357 *Synopsis:* `w[rite] [msglist] file`23358 Write the given messages to the file specified by the pathname *file*, minus the message header.
23359 Otherwise, it shall be equivalent to the **save** command.

23360 **Scroll Header Display**

23361 *Synopsis:* `z [+ | -]`

23362 Scroll the header display forward (if '+' is specified or if no option is specified) or backward (if
23363 '-' is specified) one screenful. The number of headers written shall be set by the **screen**
23364 variable.

23365 **Invoke Shell Command**

23366 *Synopsis:* `! command`

23367 Invoke the command interpreter specified by *SHELL* with two arguments: `-c` and *command*.
23368 (See also *sh -c*.) If the **bang** variable is set, each unescaped occurrence of '!' in *command* shall
23369 be replaced with the command executed by the previous ! command or ?! command escape.

23370 **Null Command**

23371 *Synopsis:* `# comment`

23372 This null command (comment) shall be ignored by *mailx*.

23373 **Display Current Message Number**

23374 *Synopsis:* `=`

23375 Write the current message number.

23376 **Command Escapes in mailx**

23377 The following commands can be entered only from input mode, by beginning a line with the
23378 escape character (by default, tilde ('~')). See the **escape** variable description for changing this
23379 special character. The format for the commands shall be:

23380 `<escape-character><command-char><separator>[<arguments>]`

23381 where the <separator> can be zero or more <blank>s.

23382 In the following descriptions, the application shall ensure that the argument *command* (but not
23383 *mailx-command*) is a shell command string. Any string acceptable to the command interpreter
23384 specified by the *SHELL* variable when it is invoked as *SHELL -c command_string* shall be valid.
23385 The command can be presented as multiple arguments (that is, quoting is not required).

23386 Command escapes that are listed with *msglist* or *mailx-command* arguments are invalid in Send
23387 Mode and produce unspecified results.

23388 `~! command` Invoke the command interpreter specified by *SHELL* with two arguments: `-c` and
23389 *command*; and then return to input mode. If the **bang** variable is set, each
23390 unescaped occurrence of '!' in *command* shall be replaced with the command
23391 executed by the previous ! command or ?! command escape.

23392 `~.` Simulate end-of-file (terminate message input).

23393 `~: mailx-command, ~_ mailx-command`
23394 Perform the command-level request.

23395 `~?` Write a summary of command escapes.

23396 `~A` This shall be equivalent to `~i Sign`.

23397 `~a` This shall be equivalent to `~i sign`.

23398	~b <i>name...</i>	Add the <i>names</i> to the blind carbon copy (Bcc) list.
23399	~c <i>name...</i>	Add the <i>names</i> to the carbon copy (Cc) list.
23400	~d	Read in the dead-letter file. See <i>DEAD</i> for a description of this file.
23401	~e	Invoke the editor, as specified by the <i>EDITOR</i> environment variable, on the partial message.
23402		
23403	~f [<i>msglist</i>]	Forward the specified messages. The specified messages shall be inserted into the current message without alteration. This command escape also shall insert message headers into the message with field selection affected by the discard , ignore , and retain commands.
23404		
23405		
23406		
23407	~F [<i>msglist</i>]	This shall be the equivalent of the ~f command escape, except that all headers shall be included in the message, regardless of previous discard , ignore , and retain commands.
23408		
23409		
23410	~h	If standard input is a terminal, prompt for a Subject line and the To , Cc , and Bcc lists. Other implementation-defined headers may also be presented for editing. If the field is written with an initial value, it can be edited as if it had just been typed.
23411		
23412		
23413	~i <i>string</i>	Insert the value of the named variable, followed by a <newline>, into the text of the message. If the string is unset or null, the message shall not be changed.
23414		
23415	~m [<i>msglist</i>]	Insert the specified messages into the message, prefixing non-empty lines with the string in the indentprefix variable. This command escape also shall insert message headers into the message, with field selection affected by the discard , ignore , and retain commands.
23416		
23417		
23418		
23419	~M [<i>msglist</i>]	This shall be the equivalent of the ~m command escape, except that all headers shall be included in the message, regardless of previous discard , ignore , and retain commands.
23420		
23421		
23422	~p	Write the message being entered. If the message is longer than crt lines (see Internal Variables in mailx (on page 593)), the output shall be paginated as described for the PAGER variable.
23423		
23424		
23425	~q	Quit (see the quit command) from input mode by simulating an interrupt. If the body of the message is not empty, the partial message shall be saved in the dead-letter file. See <i>DEAD</i> for a description of this file.
23426		
23427		
23428	~r <i>file</i> , ~< <i>file</i> , ~r !command , ~< !command	Read in the file specified by the pathname <i>file</i> . If the argument begins with an exclamation mark (' ! '), the rest of the string shall be taken as an arbitrary system command; the command interpreter specified by <i>SHELL</i> shall be invoked with two arguments: -c and <i>command</i> . The standard output of <i>command</i> shall be inserted into the message.
23429		
23430		
23431		
23432		
23433		
23434	~s <i>string</i>	Set the subject line to <i>string</i> .
23435	~t <i>name...</i>	Add the given <i>names</i> to the To list.
23436	~v	Invoke the full-screen editor, as specified by the <i>VISUAL</i> environment variable, on the partial message.
23437		
23438	~w <i>file</i>	Write the partial message, without the header, onto the file named by the pathname <i>file</i> . The file shall be created or the message shall be appended to it if the file exists.
23439		
23440		

23441 ~x Exit as with `~q`, except the message shall not be saved in the dead-letter file.
23442 ~ | command Pipe the body of the message through the given *command* by invoking the
23443 command interpreter specified by SHELL with two arguments: `-c` and *command*.
23444 If the *command* returns a successful exit status, the standard output of the
23445 command shall replace the message. Otherwise, the message shall remain
23446 unchanged. If the *command* fails, an error message giving the exit status shall be
23447 written.

23448 EXIT STATUS

23449 When the `-e` option is specified, the following exit values are returned:

23450 0 Mail was found.

23451 >0 Mail was not found or an error occurred.

23452 Otherwise, the following exit values are returned:

23453 0 Successful completion; note that this status implies that all messages were *sent*, but it gives
23454 no assurances that any of them were actually *delivered*.

23455 >0 An error occurred.

23456 CONSEQUENCES OF ERRORS

23457 When in input mode (Receive Mode) or Send Mode:

- If an error is encountered processing a command escape (see **Command Escapes in mailx** (on page 604)), a diagnostic message shall be written to standard error, and the message being composed may be modified, but this condition shall not prevent the message from being sent.
- Other errors shall prevent the sending of the message.

23463 When in command mode:

- Default.

23465 APPLICATION USAGE

23466 Delivery of messages to remote systems requires the existence of communication paths to such
23467 systems. These need not exist.

23468 Input lines are limited to {LINE_MAX} bytes, but mailers between systems may impose more
23469 severe line-length restrictions. This volume of IEEE Std 1003.1-2001 does not place any
23470 restrictions on the length of messages handled by *mailx*, and for delivery of local messages the
23471 only limitations should be the normal problems of available disk space for the target mail file.
23472 When sending messages to external machines, applications are advised to limit messages to less
23473 than 100 000 bytes because some mail gateways impose message-length restrictions.

23474 The format of the system mailbox is intentionally unspecified. Not all systems implement
23475 system mailboxes as flat files, particularly with the advent of multimedia mail messages. Some
23476 system mailboxes may be multiple files, others records in a database. The internal format of the
23477 messages themselves is specified with the historical format from Version 7, but only after the
23478 messages have been saved in some file other than the system mailbox. This was done so that
23479 many historical applications expecting text-file mailboxes are not broken.

23480 Some new formats for messages can be expected in the future, probably including binary data,
23481 bit maps, and various multimedia objects. As described here, *mailx* is not prohibited from
23482 handling such messages, but it must store them as text files in secondary mailboxes (unless
23483 some extension, such as a variable or command line option, is used to change the stored format).
23484 Its method of doing so is implementation-defined and might include translating the data into

23485 text file-compatible or readable form or omitting certain portions of the message from the stored
23486 output.

23487 The **discard** and **ignore** commands are not inverses of the **retain** command. The **retain**
23488 command discards all header-fields except those explicitly retained. The **discard** command
23489 keeps all header-fields except those explicitly discarded. If headers exist on the retained header
23490 list, **discard** and **ignore** commands are ignored.

23491 **EXAMPLES**

23492 None.

23493 **RATIONALE**

23494 The standard developers felt strongly that a method for applications to send messages to
23495 specific users was necessary. The obvious example is a batch utility, running non-interactively,
23496 that wishes to communicate errors or results to a user. However, the actual format, delivery
23497 mechanism, and method of reading the message are clearly beyond the scope of this volume of
23498 IEEE Std 1003.1-2001.

23499 The intent of this command is to provide a simple, portable interface for sending messages non-
23500 interactively. It merely defines a “front-end” to the historical mail system. It is suggested that
23501 implementations explicitly denote the sender and recipient in the body of the delivered message.
23502 Further specification of formats for either the message envelope or the message itself were
23503 deliberately not made, as the industry is in the midst of changing from the current standards to a
23504 more internationalized standard and it is probably incorrect, at this time, to require either one.

23505 Implementations are encouraged to conform to the various delivery mechanisms described in
23506 the CCITT X.400 standards or to the equivalent Internet standards, described in Internet Request
23507 for Comment (RFC) documents RFC 819, RFC 822, RFC 920, RFC 921, and RFC 1123.

23508 Many historical systems modified each body line that started with **From** by prefixing the ‘F’
23509 with ‘>’. It is unnecessary, but allowed, to do that when the string does not follow a blank line
23510 because it cannot be confused with the next header.

23511 The **edit** and **visual** commands merely edit the specified messages in a temporary file. They do
23512 not modify the contents of those messages in the mailbox; such a capability could be added as an
23513 extension, such as by using different command names.

23514 The restriction on a subject line being {LINE_MAX}-10 bytes is based on the historical format
23515 that consumes 10 bytes for **Subject:** and the trailing <newline>. Many historical mailers that a
23516 message may encounter on other systems are not able to handle lines that long, however.

23517 Like the utilities *logger* and *lp*, *mailx* admittedly is difficult to test. This was not deemed sufficient
23518 justification to exclude this utility from this volume of IEEE Std 1003.1-2001. It is also arguable
23519 that it is, in fact, testable, but that the tests themselves are not portable.

23520 When *mailx* is being used by an application that wishes to receive the results as if none of the
23521 User Portability Utilities option features were supported, the **DEAD** environment variable must
23522 be set to **/dev/null**. Otherwise, it may be subject to the file creations described in *mailx*
23523 ASYNCHRONOUS EVENTS. Similarly, if the **MAILRC** environment variable is not set to
23524 **/dev/null**, historical versions of *mailx* and *Mail* read initialization commands from a file before
23525 processing begins. Since the initialization that a user specifies could alter the contents of
23526 messages an application is trying to send, such applications must set **MAILRC** to **/dev/null**.

23527 The description of **LC_TIME** uses “may affect” because many historical implementations do not
23528 or cannot manipulate the date and time strings in the incoming mail headers. Some headers
23529 found in incoming mail do not have enough information to determine the timezone in which the
23530 mail originated, and, therefore, *mailx* cannot convert the date and time strings into the internal
23531 form that then is parsed by routines like *strftime()* that can take **LC_TIME** settings into account.

23532 Changing all these times to a user-specified format is allowed, but not required.

23533 The paginator selected when *PAGER* is null or unset is partially unspecified to allow the System
23534 V historical practice of using *pg* as the default. Bypassing the pagination function, such as by
23535 declaring that *cat* is the paginator, would not meet with the intended meaning of this
23536 description. However, any “portable user” would have to set *PAGER* explicitly to get his or her
23537 preferred paginator on all systems. The paginator choice was made partially unspecified, unlike
23538 the *VISUAL* editor choice (mandated to be *vi*) because most historical pagers follow a common
23539 theme of user input, whereas editors differ dramatically.

23540 Options to specify addresses as **cc** (carbon copy) or **bcc** (blind carbon copy) were considered to
23541 be format details and were omitted.

23542 A zero exit status implies that all messages were *sent*, but it gives no assurances that any of them
23543 were actually *delivered*. The reliability of the delivery mechanism is unspecified and is an
23544 appropriate marketing distinction between systems.

23545 In order to conform to the Utility Syntax Guidelines, a solution was required to the optional *file*
23546 option-argument to **-f**. By making *file* an operand, the guidelines are satisfied and users remain
23547 portable. However, it does force implementations to support usage such as:

23548 mailx -fin mymail.box

23549 The **no name** method of unsetting variables is not present in all historical systems, but it is in
23550 System V and provides a logical set of commands corresponding to the format of the display of
23551 options from the *mailx set* command without arguments.

23552 The **ask** and **asksub** variables are the names selected by BSD and System V, respectively, for the
23553 same feature. They are synonyms in this volume of IEEE Std 1003.1-2001.

23554 The *mailx echo* command was not documented in the BSD version and has been omitted here
23555 because it is not obviously useful for interactive users.

23556 The default prompt on the System V *mailx* is a question mark, on BSD *Mail* an ampersand. Since
23557 this volume of IEEE Std 1003.1-2001 chose the *mailx* name, it kept the System V default,
23558 assuming that BSD users would not have difficulty with this minor incompatibility (that they
23559 can override).

23560 The meanings of **r** and **R** are reversed between System V *mailx* and SunOS *Mail*. Once again,
23561 since this volume of IEEE Std 1003.1-2001 chose the *mailx* name, it kept the System V default, but
23562 allows the SunOS user to achieve the desired results using **flipr**, an internal variable in System V
23563 *mailx*, although it has not been documented in the SVID.

23564 The **indentprefix** variable, the **retain** and **unalias** commands, and the **~F** and **~M** command
23565 escapes were adopted from 4.3 BSD *Mail*.

23566 The **version** command was not included because no sufficiently general specification of the
23567 version information could be devised that would still be useful to a portable user. This
23568 command name should be used by suppliers who wish to provide version information about the
23569 *mailx* command.

23570 The “implementation-specific (unspecified) system start-up file” historically has been named
23571 */etc/mailx.rc*, but this specific name and location are not required.

23572 The intent of the wording for the **next** command is that if any command has already displayed
23573 the current message it should display a following message, but, otherwise, it should display the
23574 current message. Consider the command sequence:

23575 next 3
23576 delete 3

23577	next
23578	where the autoprint option was not set. The normative text specifies that the second next command should display a message following the third message, because even though the current message has not been displayed since it was set by the delete command, it has been displayed since the current message was anything other than message number 3. This does not always match historical practice in some implementations, where the command file address followed by next (or the default command) would skip the message for which the user had searched.
23585	FUTURE DIRECTIONS
23586	None.
23587	SEE ALSO
23588	Chapter 2 (on page 29), <i>ed</i> , <i>ls</i> , <i>more</i> , <i>vi</i>
23589	CHANGE HISTORY
23590	First released in Issue 2.
23591	Issue 5
23592	The description of the <i>EDITOR</i> environment variable is changed to indicate that <i>ed</i> is the default editor if this variable is not set. In previous issues, this default was not stated explicitly at this point but was implied further down in the text.
23595	The FUTURE DIRECTIONS section is added.
23596	Issue 6
23597	The following new requirements on POSIX implementations derive from alignment with the Single UNIX Specification:
23599	• The -F option is added.
23600	• The allnet , debug , and sendwait internal variables are added.
23601	• The C , ec , fo , and S <i>mailx</i> commands are added.
23602	In the DESCRIPTION and ENVIRONMENT VARIABLES sections, text stating “ <i>HOME</i> directory” is replaced by “directory referred to by the <i>HOME</i> environment variable”.
23604	The <i>mailx</i> utility is aligned with the IEEE P1003.2b draft standard, which includes various clarifications to resolve IEEE PASC Interpretations submitted for the ISO POSIX-2:1993 standard. In particular, the changes here address IEEE PASC Interpretations 1003.2 #10, #11, #103, #106, #108, #114, #115, #122, and #129.
23608	The normative text is reworded to avoid use of the term “must” for application requirements.
23609	The TZ entry is added to the ENVIRONMENT VARIABLES section.
23610	IEEE Std 1003.1-2001/Cor 1-2002, item XCU/TC1/D6/32 is applied, applying a change to the EXTENDED DESCRIPTION, raised by IEEE PASC Interpretation 1003.2 #122, which was overlooked in the first version of IEEE Std 1003.1-2001. 1
23613	IEEE Std 1003.1-2001/Cor 2-2004, item XCU/TC2/D6/17 is applied, updating the EXTENDED DESCRIPTION (Internal Variables in <i>mailx</i>). The system start-up file is changed from “implementation-defined” to “unspecified” for consistency with other text in the EXTENDED DESCRIPTION. 2
23614	2
23615	2
23616	2

23617 NAME

23618 make — maintain, update, and regenerate groups of programs (DEVELOPMENT)

23619 SYNOPSIS

23620 SD make [-einpqrst][-f *makefile*]...[-k | -S][macro=value]...
23621 [target_name...]
23622

23623 DESCRIPTION

23624 The *make* utility shall update files that are derived from other files. A typical case is one where
23625 object files are derived from the corresponding source files. The *make* utility examines time
23626 relationships and shall update those derived files (called targets) that have modified times
23627 earlier than the modified times of the files (called prerequisites) from which they are derived. A
23628 description file (makefile) contains a description of the relationships between files, and the
23629 commands that need to be executed to update the targets to reflect changes in their
23630 prerequisites. Each specification, or rule, shall consist of a target, optional prerequisites, and
23631 optional commands to be executed when a prerequisite is newer than the target. There are two
23632 types of rule:

- 23633 1. *Inference rules*, which have one target name with at least one period ('.') and no slash
23634 ('/')

- 23635 2. *Target rules*, which can have more than one target name

23636 In addition, *make* shall have a collection of built-in macros and inference rules that infer
23637 prerequisite relationships to simplify maintenance of programs.

23638 To receive exactly the behavior described in this section, the user shall ensure that a portable
23639 makefile shall:

- 23640 • Include the special target .POSIX
- 23641 • Omit any special target reserved for implementations (a leading period followed by
23642 uppercase letters) that has not been specified by this section

23643 The behavior of *make* is unspecified if either or both of these conditions are not met.

23644 OPTIONS

23645 The *make* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section
23646 12.2, Utility Syntax Guidelines.

23647 The following options shall be supported:

- 23648 -e Cause environment variables, including those with null values, to override macro
23649 assignments within makefiles.
- 23650 -f *makefile* Specify a different makefile. The argument *makefile* is a pathname of a description
23651 file, which is also referred to as the *makefile*. A pathname of '-' shall denote the
23652 standard input. There can be multiple instances of this option, and they shall be
23653 processed in the order specified. The effect of specifying the same option-
23654 argument more than once is unspecified.
- 23655 -i Ignore error codes returned by invoked commands. This mode is the same as if the
23656 special target .IGNORE were specified without prerequisites.
- 23657 -k Continue to update other targets that do not depend on the current target if a non-
23658 ignored error occurs while executing the commands to bring a target up-to-date.
- 23659 -n Write commands that would be executed on standard output, but do not execute
23660 them. However, lines with a plus sign ('+') prefix shall be executed. In this mode,

23661		lines with an at sign ('@') character prefix shall be written to standard output.
23662	-p	Write to standard output the complete set of macro definitions and target descriptions. The output format is unspecified.
23663		
23664	-q	Return a zero exit value if the target file is up-to-date; otherwise, return an exit value of 1. Targets shall not be updated if this option is specified. However, a makefile command line (associated with the targets) with a plus sign ('+') prefix shall be executed.
23665		
23666		
23667		
23668	-r	Clear the suffix list and do not use the built-in rules.
23669	-S	Terminate <i>make</i> if an error occurs while executing the commands to bring a target up-to-date. This shall be the default and the opposite of -k .
23670		
23671	-s	Do not write makefile command lines or touch messages (see -t) to standard output before executing. This mode shall be the same as if the special target .SILENT were specified without prerequisites.
23672		
23673		
23674	-t	Update the modification time of each target as though a <i>touch target</i> had been executed. Targets that have prerequisites but no commands (see Target Rules (on page 614)), or that are already up-to-date, shall not be touched in this manner. Write messages to standard output for each target file indicating the name of the file and that it was touched. Normally, the <i>makefile</i> command lines associated with each target are not executed. However, a command line with a plus sign ('+') prefix shall be executed.
23675		
23676		
23677		
23678		
23679		
23680		
23681		Any options specified in the <i>MAKEFLAGS</i> environment variable shall be evaluated before any options specified on the <i>make</i> utility command line. If the -k and -S options are both specified on the <i>make</i> utility command line or by the <i>MAKEFLAGS</i> environment variable, the last option specified shall take precedence. If the -f or -p options appear in the <i>MAKEFLAGS</i> environment variable, the result is undefined.
23682		
23683		
23684		
23685		

23686 OPERANDS

23687 The following operands shall be supported:

23688 *target_name* Target names, as defined in the EXTENDED DESCRIPTION section. If no target is specified, while *make* is processing the makefiles, the first target that *make* encounters that is not a special target or an inference rule shall be used.

23689 *macro=value* Macro definitions, as defined in **Macros** (on page 616).

23690 If the *target_name* and *macro=value* operands are intermixed on the *make* utility command line, the results are unspecified.

23694 STDIN

23695 The standard input shall be used only if the *makefile* option-argument is '-'. See the INPUT FILES section.

23697 INPUT FILES

23698 The input file, otherwise known as the makefile, is a text file containing rules, macro definitions, and comments. See the EXTENDED DESCRIPTION section.

23700 ENVIRONMENT VARIABLES

23701 The following environment variables shall affect the execution of *make*:

23702 *LANG* Provide a default value for the internationalization variables that are unset or null.
23703 (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2,
23704 Internationalization Variables for the precedence of internationalization variables
23705 used to determine the values of locale categories.)

23706	<i>LC_ALL</i>	If set to a non-empty string value, override the values of all the other internationalization variables.
23708	<i>LC_CTYPE</i>	Determine the locale for the interpretation of sequences of bytes of text data as characters (for example, single-byte as opposed to multi-byte characters in arguments and input files).
23711	<i>LC_MESSAGES</i>	Determine the locale that should be used to affect the format and contents of diagnostic messages written to standard error.
23714	<i>MAKEFLAGS</i>	This variable shall be interpreted as a character string representing a series of option characters to be used as the default options. The implementation shall accept both of the following formats (but need not accept them when intermixed): <ul style="list-style-type: none">• The characters are option letters without the leading hyphens or <blank> separation used on a <i>make</i> utility command line.• The characters are formatted in a manner similar to a portion of the <i>make</i> utility command line: options are preceded by hyphens and <blank>-separated as described in the Base Definitions volume of IEEE Std 1003.1-2001, Section 12.2, Utility Syntax Guidelines. The <i>macro=value</i> macro definition operands can also be included. The difference between the contents of <i>MAKEFLAGS</i> and the <i>make</i> utility command line is that the contents of the variable shall not be subjected to the word expansions (see Section 2.6 (on page 36)) associated with parsing the command line values.
23728 XSI	<i>NLSPATH</i>	Determine the location of message catalogs for the processing of <i>LC_MESSAGES</i> .
23729 XSI	<i>PROJECTDIR</i>	Provide a directory to be used to search for SCCS files not found in the current directory. In all of the following cases, the search for SCCS files is made in the directory SCCS in the identified directory. If the value of <i>PROJECTDIR</i> begins with a slash, it shall be considered an absolute pathname; otherwise, the value of <i>PROJECTDIR</i> is treated as a user name and that user's initial working directory shall be examined for a subdirectory src or source . If such a directory is found, it shall be used. Otherwise, the value is used as a relative pathname. If <i>PROJECTDIR</i> is not set or has a null value, the search for SCCS files shall be made in the directory SCCS in the current directory. The setting of <i>PROJECTDIR</i> affects all files listed in the remainder of this utility description for files with a component named SCCS .
23741	The value of the <i>SHELL</i> environment variable shall not be used as a macro and shall not be modified by defining the SHELL macro in a makefile or on the command line. All other environment variables, including those with null values, shall be used as macros, as defined in Macros (on page 616).	
23745	ASYNCHRONOUS EVENTS	If not already ignored, <i>make</i> shall trap SIGHUP, SIGTERM, SIGINT, and SIGQUIT and remove the current target unless the target is a directory or the target is a prerequisite of the special target .PRECIOUS or unless one of the -n , -p , or -q options was specified. Any targets removed in this manner shall be reported in diagnostic messages of unspecified format, written to standard error. After this cleanup process, if any, <i>make</i> shall take the standard action for all other signals.

23752 **STDOUT**

23753 The *make* utility shall write all commands to be executed to standard output unless the **-s** option
23754 was specified, the command is prefixed with an at sign, or the special target **.SILENT** has either
23755 the current target as a prerequisite or has no prerequisites. If *make* is invoked without any work
23756 needing to be done, it shall write a message to standard output indicating that no action was
23757 taken. If the **-t** option is present and a file is touched, *make* shall write to standard output a
23758 message of unspecified format indicating that the file was touched, including the filename of the
23759 file.

23760 **STDERR**

23761 The standard error shall be used only for diagnostic messages.

23762 **OUTPUT FILES**

23763 Files can be created when the **-t** option is present. Additional files can also be created by the
23764 utilities invoked by *make*.

23765 **EXTENDED DESCRIPTION**

23766 The *make* utility attempts to perform the actions required to ensure that the specified targets are
23767 up-to-date. A target is considered out-of-date if it is older than any of its prerequisites or if it
23768 does not exist. The *make* utility shall treat all prerequisites as targets themselves and recursively
23769 ensure that they are up-to-date, processing them in the order in which they appear in the rule.
23770 The *make* utility shall use the modification times of files to determine whether the corresponding
23771 targets are out-of-date.

23772 After *make* has ensured that all of the prerequisites of a target are up-to-date and if the target is
23773 out-of-date, the commands associated with the target entry shall be executed. If there are no
23774 commands listed for the target, the target shall be treated as up-to-date.

23775 **Makefile Syntax**

23776 A makefile can contain rules, macro definitions (see **Macros** (on page 616)), and comments.
23777 There are two kinds of rules: *inference rules* and *target rules*. The *make* utility shall contain a set of
23778 built-in inference rules. If the **-r** option is present, the built-in rules shall not be used and the
23779 suffix list shall be cleared. Additional rules of both types can be specified in a makefile. If a rule
23780 is defined more than once, the value of the rule shall be that of the last one specified. Macros can
23781 also be defined more than once, and the value of the macro is specified in **Macros** (on page 616).
23782 Comments start with a number sign ('#') and continue until an unescaped <newline> is
23783 reached.

23784 By default, the following files shall be tried in sequence: **./makefile** and **./Makefile**. If neither
23785 XSI **./makefile** or **./Makefile** are found, other implementation-defined files may also be tried. On
23786 XSI-conformant systems, the additional files **./s.makefile**, **SCCS/s.makefile**, **./s.Makefile**, and
23787 **SCCS/s.Makefile** shall also be tried.

23788 The **-f** option shall direct *make* to ignore any of these default files and use the specified argument
23789 as a makefile instead. If the '-' argument is specified, standard input shall be used.

23790 The term *makefile* is used to refer to any rules provided by the user, whether in **./makefile** or its
23791 variants, or specified by the **-f** option.

23792 The rules in makefiles shall consist of the following types of lines: target rules, including special
23793 targets (see **Target Rules** (on page 614)), inference rules (see **Inference Rules** (on page 617)),
23794 macro definitions (see **Macros** (on page 616)), empty lines, and comments.

23795 When an escaped <newline> (one preceded by a backslash) is found anywhere in the makefile
23796 except in a command line, it shall be replaced, along with any leading white space on the
23797 following line, with a single <space>. When an escaped <newline> is found in a command line

23798 in a makefile, the command line shall contain the backslash, the <newline>, and the next line,
23799 except that the first character of the next line shall not be included if it is a <tab>.

23800 Makefile Execution

23801 Makefile command lines shall be processed one at a time by writing the makefile command line
23802 to the standard output (unless one of the conditions listed under '@' suppresses the writing)
23803 and executing the command(s) in the line. A <tab> may precede the command to standard
23804 output. Command execution shall be as if the makefile command line were the argument to the
23805 *system()* function. The environment for the command being executed shall contain all of the
23806 variables in the environment of *make*.

23807 By default, when *make* receives a non-zero status from the execution of a command, it shall
23808 terminate with an error message to standard error.

23809 Makefile command lines can have one or more of the following prefixes: a hyphen ('-'), an at
23810 sign ('@'), or a plus sign ('+'). These shall modify the way in which *make* processes the
23811 command. When a command is written to standard output, the prefix shall not be included in
23812 the output.

- 23813 – If the command prefix contains a hyphen, or the **-i** option is present, or the special target
23814 **.IGNORE** has either the current target as a prerequisite or has no prerequisites, any error
23815 found while executing the command shall be ignored.
- 23816 @ If the command prefix contains an at sign and the *make* utility command line **-n** option is
23817 not specified, or the **-s** option is present, or the special target **.SILENT** has either the current
23818 target as a prerequisite or has no prerequisites, the command shall not be written to
23819 standard output before it is executed.
- 23820 + If the command prefix contains a plus sign, this indicates a makefile command line that
23821 shall be executed even if **-n**, **-q**, or **-t** is specified.

23822 Target Rules

23823 Target rules are formatted as follows:

```
23824 target [target...]: [prerequisite...][;command]  
23825 [<tab>command  
23826 <tab>command  
23827 ...]  
23828 line that does not begin with <tab>
```

23829 Target entries are specified by a <blank>-separated, non-null list of targets, then a colon, then a
23830 <blank>-separated, possibly empty list of prerequisites. Text following a semicolon, if any, and
23831 all following lines that begin with a <tab>, are makefile command lines to be executed to update
23832 the target. The first non-empty line that does not begin with a <tab> or '#' shall begin a new
23833 entry. An empty or blank line, or a line beginning with '#', may begin a new entry.

23834 Applications shall select target names from the set of characters consisting solely of periods,
23835 underscores, digits, and alphabets from the portable character set (see the Base Definitions
23836 volume of IEEE Std 1003.1-2001, Section 6.1, Portable Character Set). Implementations may allow
23837 other characters in target names as extensions. The interpretation of targets containing the
23838 characters '%' and '"' is implementation-defined.

23839 A target that has prerequisites, but does not have any commands, can be used to add to the
23840 prerequisite list for that target. Only one target rule for any given target can contain commands.

23841	Lines that begin with one of the following are called <i>special targets</i> and control the operation of
23842	make:
23843	.DEFAULT If the makefile uses this special target, the application shall ensure that it is
23844	specified with commands, but without prerequisites. The commands shall be used
23845	by <i>make</i> if there are no other rules available to build a target.
23846	.IGNORE Prerequisites of this special target are targets themselves; this shall cause errors
23847	from commands associated with them to be ignored in the same manner as
23848	specified by the -i option. Subsequent occurrences of .IGNORE shall add to the
23849	list of targets ignoring command errors. If no prerequisites are specified, <i>make</i> shall
23850	behave as if the -i option had been specified and errors from all commands
23851	associated with all targets shall be ignored.
23852	.POSIX The application shall ensure that this special target is specified without
23853	prerequisites or commands. If it appears as the first non-comment line in the
23854	makefile, <i>make</i> shall process the makefile as specified by this section; otherwise, the
23855	behavior of <i>make</i> is unspecified.
23856	.PRECIOUS Prerequisites of this special target shall not be removed if <i>make</i> receives one of the
23857	asynchronous events explicitly described in the ASYNCHRONOUS EVENTS
23858	section. Subsequent occurrences of .PRECIOUS shall add to the list of precious
23859	files. If no prerequisites are specified, all targets in the makefile shall be treated as
23860	if specified with .PRECIOUS .
23861 XSI	.SCCS_GET The application shall ensure that this special target is specified without
23862	prerequisites. If this special target is included in a makefile, the commands
23863	specified with this target shall replace the default commands associated with this
23864	special target (see Default Rules (on page 620)). The commands specified with
23865	this target are used to get all SCCS files that are not found in the current directory.
23866	When source files are named in a dependency list, <i>make</i> shall treat them just like
23867	any other target. Because the source file is presumed to be present in the directory,
23868	there is no need to add an entry for it to the makefile. When a target has no
23869	dependencies, but is present in the directory, <i>make</i> shall assume that that file is up-
23870	to-date. If, however, an SCCS file named SCCS/s.source_file is found for a target
23871	<i>source_file</i> , <i>make</i> compares the timestamp of the target file with that of the
23872	SCCS/s.source_file to ensure the target is up-to-date. If the target is missing, or if
23873	the SCCS file is newer, <i>make</i> shall automatically issue the commands specified for
23874	the .SCCS_GET special target to retrieve the most recent version. However, if the
23875	target is writable by anyone, <i>make</i> shall not retrieve a new version.
23876	.SILENT Prerequisites of this special target are targets themselves; this shall cause
23877	commands associated with them not to be written to the standard output before
23878	they are executed. Subsequent occurrences of .SILENT shall add to the list of
23879	targets with silent commands. If no prerequisites are specified, <i>make</i> shall behave
23880	as if the -s option had been specified and no commands or touch messages
23881	associated with any target shall be written to standard output.
23882	.SUFFIXES Prerequisites of .SUFFIXES shall be appended to the list of known suffixes and are
23883	used in conjunction with the inference rules (see Inference Rules (on page 617)). If
23884	.SUFFIXES does not have any prerequisites, the list of known suffixes shall be
23885	cleared.
23886	The special targets .IGNORE , .POSIX , .PRECIOUS , .SILENT , and .SUFFIXES shall be specified
23887	without commands.

23888 Targets with names consisting of a leading period followed by the uppercase letters "POSIX"
23889 and then any other characters are reserved for future standardization. Targets with names
23890 consisting of a leading period followed by one or more uppercase letters are reserved for
23891 implementation extensions.

23892 **Macros**

23893 Macro definitions are in the form:

23894 *string1* = [*string2*]

23895 The macro named *string1* is defined as having the value of *string2*, where *string2* is defined as all
23896 characters, if any, after the equal sign, up to a comment character ('#') or an unescaped
23897 <newline>. Any <blank>s immediately before or after the equal sign shall be ignored.

23898 Applications shall select macro names from the set of characters consisting solely of periods,
23899 underscores, digits, and alphabets from the portable character set (see the Base Definitions
23900 volume of IEEE Std 1003.1-2001, Section 6.1, Portable Character Set). A macro name shall not
23901 contain an equals sign. Implementations may allow other characters in macro names as
23902 extensions.

23903 Macros can appear anywhere in the makefile. Macro expansions using the forms `$(string1)` or
23904 `${string1}` shall be replaced by *string2*, as follows:

- 23905 • Macros in target lines shall be evaluated when the target line is read.
- 23906 • Macros in makefile command lines shall be evaluated when the command is executed.
- 23907 • Macros in the string before the equals sign in a macro definition shall be evaluated when the
23908 macro assignment is made.
- 23909 • Macros after the equals sign in a macro definition shall not be evaluated until the defined
23910 macro is used in a rule or command, or before the equals sign in a macro definition.

23911 The parentheses or braces are optional if *string1* is a single character. The macro `$$` shall be
23912 replaced by the single character '\$'. If *string1* in a macro expansion contains a macro
23913 expansion, the results are unspecified.

23914 Macro expansions using the forms `$(string1[:subst1=subst2])` or `${string1[:subst1=subst2]}` can
23915 be used to replace all occurrences of *subst1* with *subst2* when the macro substitution is
23916 performed. The *subst1* to be replaced shall be recognized when it is a suffix at the end of a word
23917 in *string1* (where a *word*, in this context, is defined to be a string delimited by the beginning of
23918 the line, a <blank>, or a <newline>). If *string1* in a macro expansion contains a macro expansion,
23919 the results are unspecified.

23920 Macro expansions in *string1* of macro definition lines shall be evaluated when read. Macro
23921 expansions in *string2* of macro definition lines shall be performed when the macro identified by
23922 *string1* is expanded in a rule or command.

23923 Macro definitions shall be taken from the following sources, in the following logical order,
23924 before the makefile(s) are read.

- 23925 1. Macros specified on the *make* utility command line, in the order specified on the command
23926 line. It is unspecified whether the internal macros defined in **Internal Macros** (on page 619)
23927 are accepted from this source.
- 23928 2. Macros defined by the *MAKEFLAGS* environment variable, in the order specified in the
23929 environment variable. It is unspecified whether the internal macros defined in **Internal
23930 Macros** (on page 619) are accepted from this source.

23931 3. The contents of the environment, excluding the *MAKEFLAGS* and *SHELL* variables and
23932 including the variables with null values.

23933 4. Macros defined in the inference rules built into *make*.

23934 Macro definitions from these sources shall not override macro definitions from a lower-
23935 numbered source. Macro definitions from a single source (for example, the *make* utility
23936 command line, the *MAKEFLAGS* environment variable, or the other environment variables) shall
23937 override previous macro definitions from the same source.

23938 Macros defined in the makefile(s) shall override macro definitions that occur before them in the
23939 makefile(s) and macro definitions from source 4. If the *-e* option is not specified, macros defined
23940 in the makefile(s) shall override macro definitions from source 3. Macros defined in the
23941 makefile(s) shall not override macro definitions from source 1 or source 2.

23942 Before the makefile(s) are read, all of the *make* utility command line options (except *-f* and *-p*)
23943 and *make* utility command line macro definitions (except any for the *MAKEFLAGS* macro), not
23944 already included in the *MAKEFLAGS* macro, shall be added to the *MAKEFLAGS* macro, quoted
23945 in an implementation-defined manner such that when *MAKEFLAGS* is read by another instance
23946 of the *make* command, the original macro's value is recovered. Other implementation-defined
23947 options and macros may also be added to the *MAKEFLAGS* macro. If this modifies the value of
23948 the *MAKEFLAGS* macro, or, if the *MAKEFLAGS* macro is modified at any subsequent time, the
23949 *MAKEFLAGS* environment variable shall be modified to match the new value of the
23950 *MAKEFLAGS* macro. The result of setting *MAKEFLAGS* in the Makefile is unspecified.

23951 Before the makefile(s) are read, all of the *make* utility command line macro definitions (except the
23952 *MAKEFLAGS* macro or the *SHELL* macro) shall be added to the environment of *make*. Other
23953 implementation-defined variables may also be added to the environment of *make*.

23954 The **SHELL** macro shall be treated specially. It shall be provided by *make* and set to the
23955 pathname of the shell command language interpreter (see *sh*). The **SHELL** environment variable
23956 shall not affect the value of the **SHELL** macro. If **SHELL** is defined in the makefile or is specified
23957 on the command line, it shall replace the original value of the **SHELL** macro, but shall not affect
23958 the **SHELL** environment variable. Other effects of defining **SHELL** in the makefile or on the
23959 command line are implementation-defined.

23960 Inference Rules

23961 Inference rules are formatted as follows:

23962 *target* :
23963 <tab>*command*
23964 [<tab>*command*]
23965 ...
23966 *line that does not begin with <tab> or #*

23967 The application shall ensure that the *target* portion is a valid target name (see **Target Rules** (on
23968 page 614)) of the form *.s2* or *.s1.s2* (where *.s1* and *.s2* are suffixes that have been given as
23969 prerequisites of the **.SUFFIXES** special target and *s1* and *s2* do not contain any slashes or
23970 periods.) If there is only one period in the target, it is a single-suffix inference rule. Targets with
23971 two periods are double-suffix inference rules. Inference rules can have only one target before the
23972 colon.

23973 The application shall ensure that the makefile does not specify prerequisites for inference rules;
23974 no characters other than white space shall follow the colon in the first line, except when creating
23975 the *empty rule*, described below. Prerequisites are inferred, as described below.

23976 Inference rules can be redefined. A target that matches an existing inference rule shall overwrite
23977 the old inference rule. An empty rule can be created with a command consisting of simply a
23978 semicolon (that is, the rule still exists and is found during inference rule search, but since it is
23979 empty, execution has no effect). The empty rule can also be formatted as follows:

23980 *rule*: ;

23981 where zero or more <blank>s separate the colon and semicolon.

23982 The *make* utility uses the suffixes of targets and their prerequisites to infer how a target can be
23983 made up-to-date. A list of inference rules defines the commands to be executed. By default, *make*
23984 contains a built-in set of inference rules. Additional rules can be specified in the makefile.

23985 The special target **.SUFFIXES** contains as its prerequisites a list of suffixes that shall be used by
23986 the inference rules. The order in which the suffixes are specified defines the order in which the
23987 inference rules for the suffixes are used. New suffixes shall be appended to the current list by
23988 specifying a **.SUFFIXES** special target in the makefile. A **.SUFFIXES** target with no prerequisites
23989 shall clear the list of suffixes. An empty **.SUFFIXES** target followed by a new **.SUFFIXES** list is
23990 required to change the order of the suffixes.

23991 Normally, the user would provide an inference rule for each suffix. The inference rule to update
23992 a target with a suffix **.s1** from a prerequisite with a suffix **.s2** is specified as a target **.s2.s1**. The
23993 internal macros provide the means to specify general inference rules (see **Internal Macros** (on
23994 page 619)).

23995 When no target rule is found to update a target, the inference rules shall be checked. The suffix
23996 of the target (**.s1**) to be built is compared to the list of suffixes specified by the **.SUFFIXES** special
23997 targets. If the **.s1** suffix is found in **.SUFFIXES**, the inference rules shall be searched in the order
23998 defined for the first **.s2.s1** rule whose prerequisite file (**\$*.s2**) exists. If the target is out-of-date
23999 with respect to this prerequisite, the commands for that inference rule shall be executed.

24000 If the target to be built does not contain a suffix and there is no rule for the target, the single
24001 suffix inference rules shall be checked. The single-suffix inference rules define how to build a
24002 target if a file is found with a name that matches the target name with one of the single suffixes
24003 appended. A rule with one suffix **.s2** is the definition of how to build *target* from *target.s2*. The
24004 other suffix (**.s1**) is treated as null.

24005 XSI A tilde ('~') in the above rules refers to an SCCS file in the current directory. Thus, the rule **.c~.o**
24006 would transform an SCCS C-language source file into an object file (.o). Because the **s.** of the
24007 SCCS files is a prefix, it is incompatible with *make*'s suffix point of view. Hence, the ' ~' is a way
24008 of changing any file reference into an SCCS file reference.

24009 Libraries

24010 If a target or prerequisite contains parentheses, it shall be treated as a member of an archive
24011 library. For the *lib(member.o)* expression *lib* refers to the name of the archive library and *member.o*
24012 to the member name. The application shall ensure that the member is an object file with the **.o**
24013 suffix. The modification time of the expression is the modification time for the member as kept
24014 in the archive library; see *ar*. The **.a** suffix shall refer to an archive library. The **.s2.a** rule shall be
24015 used to update a member in the library from a file with a suffix **.s2**.

24016

Internal Macros

24017

The *make* utility shall maintain five internal macros that can be used in target and inference rules. In order to clearly define the meaning of these macros, some clarification of the terms *target rule*, *inference rule*, *target*, and *prerequisite* is necessary.

24020

Target rules are specified by the user in a makefile for a particular target. Inference rules are user-specified or *make*-specified rules for a particular class of target name. Explicit prerequisites are those prerequisites specified in a makefile on target lines. Implicit prerequisites are those prerequisites that are generated when inference rules are used. Inference rules are applied to implicit prerequisites or to explicit prerequisites that do not have target rules defined for them in the makefile. Target rules are applied to targets specified in the makefile.

24018

24019

24021

Before any target in the makefile is updated, each of its prerequisites (both explicit and implicit) shall be updated. This shall be accomplished by recursively processing each prerequisite. Upon recursion, each prerequisite shall become a target itself. Its prerequisites in turn shall be processed recursively until a target is found that has no prerequisites, at which point the recursion stops. The recursion shall then back up, updating each target as it goes.

24023

24024

24025

In the definitions that follow, the word *target* refers to one of:

24027

- A target specified in the makefile
- An explicit prerequisite specified in the makefile that becomes the target when *make* processes it during recursion
- An implicit prerequisite that becomes a target when *make* processes it during recursion

24029

24030

In the definitions that follow, the word *prerequisite* refers to one of the following:

24032

- An explicit prerequisite specified in the makefile for a particular target
- An implicit prerequisite generated as a result of locating an appropriate inference rule and corresponding file that matches the suffix of the target

24034

24035

The five internal macros are:

24037

\$@ The \$@ shall evaluate to the full target name of the current target, or the archive filename part of a library archive target. It shall be evaluated for both target and inference rules.

24039

24041

For example, in the .c.a inference rule, \$@ represents the out-of-date .a file to be built. Similarly, in a makefile target rule to build lib.a from file.c, \$@ represents the out-of-date lib.a.

24043

24044

\$% The \$% macro shall be evaluated only when the current target is an archive library member of the form libname(member.o). In these cases, \$@ shall evaluate to libname and \$% shall evaluate to member.o. The \$% macro shall be evaluated for both target and inference rules.

24046

24048

For example, in a makefile target rule to build lib.a(file.o), \$% represents file.o, as opposed to \$@, which represents lib.a.

24050

24053

\$? The \$? macro shall evaluate to the list of prerequisites that are newer than the current target. It shall be evaluated for both target and inference rules.

24054

24055

For example, in a makefile target rule to build prog from file1.o, file2.o, and file3.o, and where prog is not out-of-date with respect to file1.o, but is out-of-date with respect to file2.o and file3.o, \$? represents file2.o and file3.o.

24058 \$< In an inference rule, the \$< macro shall evaluate to the filename whose existence
 24059 allowed the inference rule to be chosen for the target. In the .DEFAULT rule, the \$<
 24060 macro shall evaluate to the current target name. The meaning of the \$< macro shall be
 24061 otherwise unspecified.

24062 For example, in the .c.a inference rule, \$< represents the prerequisite .c file.

24063 \$* The \$* macro shall evaluate to the current target name with its suffix deleted. It shall be
 24064 evaluated at least for inference rules.

24065 For example, in the .c.a inference rule, \$*.o represents the out-of-date .o file that
 24066 corresponds to the prerequisite .c file.

24067 Each of the internal macros has an alternative form. When an uppercase 'D' or 'F' is appended
 24068 to any of the macros, the meaning shall be changed to the *directory part* for 'D' and *filename part*
 24069 for 'F'. The directory part is the path prefix of the file without a trailing slash; for the current
 24070 directory, the directory part is '.'. When the \$? macro contains more than one prerequisite
 24071 filename, the \${?D} and \${?F} (or \${?D} and \${?F}) macros expand to a list of directory name parts
 24072 and filename parts respectively.

24073 For the target *lib(member.o)* and the s2.a rule, the internal macros shall be defined as:

24074 \$< member.s2

24075 \$* member

24076 \$@ lib

24077 \$? member.s2

24078 \$% member.o

24079 Default Rules

24080 The default rules for *make* shall achieve results that are the same as if the following were used.
 24081 Implementations that do not support the C-Language Development Utilities option may omit
CC, **CFLAGS**, **YACC**, **YFLAGS**, **LEX**, **LFLAGS**, **LDFLAGS**, and the .c, .y, and .l inference rules.
 24083 Implementations that do not support FORTRAN may omit **FC**, **FFLAGS**, and the .f inference
 24084 rules. Implementations may provide additional macros and rules.

24085 SPECIAL TARGETS

24086 XSI .SCCS_GET: sccs \$(SCCSFLAGS) get \$(SCCSGETFLAGS) \$@
 24087

24088 XSI .SUFFIXES: .o .c .y .l .a .sh .f .c~ .y~ .l~ .sh~ .f~

24089 MACROS

24090 MAKE=make

24091 AR=ar

24092 ARFLAGS=-rv

24093 YACC=yacc

24094 YFLAGS=

24095 LEX=lex

24096 LFLAGS=

24097 LDFLAGS=

24098 CC=c99

24099 CFLAGS=-O

24100 FC=fort77

```

24101      FFLAGS=-O 1
24102 XSI    GET=get
24103      GFLAGS=
24104      SCCSFLAGS=
24105      SCCSGETFLAGS=-s
24106
24107      SINGLE SUFFIX RULES
24108      .c:
24109          $(CC) $(CFLAGS) $(LDFLAGS) -o $@ $<
24110      .f:
24111          $(FC) $(FFLAGS) $(LDFLAGS) -o $@ $<
24112      .sh:
24113          cp $< $@
24114          chmod a+x $@
24115 XSI    .c~:
24116          $(GET) $(GFLAGS) -p $< > $*.c
24117          $(CC) $(CFLAGS) $(LDFLAGS) -o $@ $*.c
24118      .f~:
24119          $(GET) $(GFLAGS) -p $< > $*.f
24120          $(FC) $(FFLAGS) $(LDFLAGS) -o $@ $*.f
24121      .sh~:
24122          $(GET) $(GFLAGS) -p $< > $*.sh
24123          cp $*.sh $@
24124          chmod a+x $@
24125
24126      DOUBLE SUFFIX RULES
24127      .c.o:
24128          $(CC) $(CFLAGS) -c $<
24129      .f.o:
24130          $(FC) $(FFLAGS) -c $<
24131      .y.o:
24132          $(YACC) $(YFLAGS) $<
24133          $(CC) $(CFLAGS) -c y.tab.c
24134          rm -f y.tab.c
24135          mv y.tab.o $@
24136      .l.o:
24137          $(LEX) $(LFLAGS) $<
24138          $(CC) $(CFLAGS) -c lex.yy.c
24139          rm -f lex.yy.c
24140          mv lex.yy.o $@
24141      .y.c:
24142          $(YACC) $(YFLAGS) $<
24143          mv y.tab.c $@
24144      .l.c:
24145          $(LEX) $(LFLAGS) $<

```

```

24146          mv lex.yy.c $@
24147 XSI      .c~.o:
24148          $(GET) $(GFLAGS) -p $< > $*.c
24149          $(CC) $(CFLAGS) -c $*.c
24150      .f~.o:
24151          $(GET) $(GFLAGS) -p $< > $*.f
24152          $(FC) $(FFLAGS) -c $*.f
24153      .y~.o:
24154          $(GET) $(GFLAGS) -p $< > $*.y
24155          $(YACC) $(YFLAGS) $*.y
24156          $(CC) $(CFLAGS) -c y.tab.c
24157          rm -f y.tab.c
24158          mv y.tab.o $@
24159      .l~.o:
24160          $(GET) $(GFLAGS) -p $< > $*.l
24161          $(LEX) $(LFLAGS) $*.l
24162          $(CC) $(CFLAGS) -c lex.yy.c
24163          rm -f lex.yy.c
24164          mv lex.yy.o $@
24165      .y~.c:
24166          $(GET) $(GFLAGS) -p $< > $*.y
24167          $(YACC) $(YFLAGS) $*.y
24168          mv y.tab.c $@
24169      .l~.c:
24170          $(GET) $(GFLAGS) -p $< > $*.l
24171          $(LEX) $(LFLAGS) $*.l
24172          mv lex.yy.c $@
24173
24174      .c.a:
24175          $(CC) -c $(CFLAGS) $<
24176          $(AR) $(ARFLAGS) $@ $*.o
24177          rm -f $*.o
24178      .f.a:
24179          $(FC) -c $(FFLAGS) $<
24180          $(AR) $(ARFLAGS) $@ $*.o
24181          rm -f $*.o

```

24182 EXIT STATUS

24183 When the **-q** option is specified, the *make* utility shall exit with one of the following values:

- 24184 0 Successful completion.
- 24185 1 The target was not up-to-date.
- 24186 >1 An error occurred.

24187 When the **-q** option is not specified, the *make* utility shall exit with one of the following values:

- 24188 0 Successful completion.
- 24189 >0 An error occurred.

24190 CONSEQUENCES OF ERRORS

24191 Default.

24192 APPLICATION USAGE

24193 If there is a source file (such as `./source.c`) and there are two SCCS files corresponding to it
24194 (`./s.source.c` and `./SCCS/s.source.c`), on XSI-conformant systems *make* uses the SCCS file in the
24195 current directory. However, users are advised to use the underlying SCCS utilities (*admin*, *delta*,
24196 *get*, and so on) or the *scs* utility for all source files in a given directory. If both forms are used for
24197 a given source file, future developers are very likely to be confused.

24198 It is incumbent upon portable makefiles to specify the `.POSIX` special target in order to
24199 guarantee that they are not affected by local extensions.

24200 The `-k` and `-S` options are both present so that the relationship between the command line, the
24201 `MAKEFLAGS` variable, and the makefile can be controlled precisely. If the `k` flag is passed in
24202 `MAKEFLAGS` and a command is of the form:

24203 `$(MAKE) -S foo`

24204 then the default behavior is restored for the child *make*.

24205 When the `-n` option is specified, it is always added to `MAKEFLAGS`. This allows a recursive
24206 *make -n target* to be used to see all of the action that would be taken to update *target*.

24207 Because of widespread historical practice, interpreting a '#' number sign inside a variable as
24208 the start of a comment has the unfortunate side effect of making it impossible to place a number
24209 sign in a variable, thus forbidding something like:

24210 `CFLAGS = "-D COMMENT_CHAR='#' "`

24211 Many historical *make* utilities stop chaining together inference rules when an intermediate target
24212 is nonexistent. For example, it might be possible for a *make* to determine that both `.y.c` and `.c.o`
24213 could be used to convert a `.y` to a `.o`. Instead, in this case, *make* requires the use of a `.y.o` rule.

24214 The best way to provide portable makefiles is to include all of the rules needed in the makefile
24215 itself. The rules provided use only features provided by other parts of this volume of
24216 IEEE Std 1003.1-2001. The default rules include rules for optional commands in this volume of
24217 IEEE Std 1003.1-2001. Only rules pertaining to commands that are provided are needed in an
24218 implementation's default set.

24219 Macros used within other macros are evaluated when the new macro is used rather than when
24220 the new macro is defined. Therefore:

24221 `MACRO = value1`
24222 `NEW = $(MACRO)`
24223 `MACRO = value2`

24224 `target:`
24225 `echo $(NEW)`

24226 would produce `value2` and not `value1` since `NEW` was not expanded until it was needed in the
24227 `echo` command line.

24228 Some historical applications have been known to intermix *target_name* and *macro=name* operands
24229 on the command line, expecting that all of the macros are processed before any of the targets are
24230 dealt with. Conforming applications do not do this, although some backwards-compatibility
24231 support may be included in some implementations.

24232 The following characters in filenames may give trouble: '=', ':', '^', '^', and '@'. For
24233 inference rules, the description of \$< and \$? seem similar. However, an example shows the

24234 minor difference. In a makefile containing:

24235 foo.o: foo.h

24236 if **foo.h** is newer than **foo.o**, yet **foo.c** is older than **foo.o**, the built-in rule to make **foo.o** from **foo.c** is used, with \$< equal to **foo.c** and \$? equal to **foo.h**. If **foo.c** is also newer than **foo.o**, \$< is equal to **foo.c** and \$? is equal to **foo.h** **foo.c**.

24239 EXAMPLES

- 24240 1. The following command:
24241 make
24242 makes the first target found in the makefile.
- 24243 2. The following command:
24244 make junk
24245 makes the target **junk**.
- 24246 3. The following makefile says that **pgm** depends on two files, **a.o** and **b.o**, and that they in
24247 turn depend on their corresponding source files (**a.c** and **b.c**), and a common file **incl.h**:

24248 pgm: a.o b.o
24249 c99 a.o b.o -o pgm
24250 a.o: incl.h a.c
24251 c99 -c a.c
24252 b.o: incl.h b.c
24253 c99 -c b.c

- 24254 4. An example for making optimized .o files from .c files is:

24255 .c.o:
24256 c99 -c -O \$*.c

24257 or:

24258 .c.o:
24259 c99 -c -O \$<

- 24260 5. The most common use of the archive interface follows. Here, it is assumed that the source
24261 files are all C-language source:

24262 lib: lib(file1.o) lib(file2.o) lib(file3.o)
24263 @echo lib is now up-to-date

24264 The **.c.a** rule is used to make **file1.o**, **file2.o**, and **file3.o** and insert them into **lib**.

24265 The treatment of escaped <newline>s throughout the makefile is historical practice. For
24266 example, the inference rule:

24267 .c.o\
24268 :

24269 works, and the macro:

24270 f= bar baz\
24271 biz
24272 a:
24273 echo ==\$f==

24274 echoes " ==bar baz biz==".
24275 If \$? were:
24276 /usr/include/stdio.h /usr/include/unistd.h foo.h
24277 then \$(?D) would be:
24278 /usr/include /usr/include .
24279 and \$(?F) would be:
24280 stdio.h unistd.h foo.h
24281 6. The contents of the built-in rules can be viewed by running:
24282 make -p -f /dev/null 2>/dev/null

24283 RATIONALE

24284 The *make* utility described in this volume of IEEE Std 1003.1-2001 is intended to provide the
24285 means for changing portable source code into executables that can be run on an
24286 IEEE Std 1003.1-2001-conforming system. It reflects the most common features present in
24287 System V and BSD *makes*.

24288 Historically, the *make* utility has been an especially fertile ground for vendor and research
24289 organization-specific syntax modifications and extensions. Examples include:

- 24290 • Syntax supporting parallel execution (such as from various multi-processor vendors, GNU,
24291 and others)
- 24292 • Additional “operators” separating targets and their prerequisites (System V, BSD, and
24293 others)
- 24294 • Specifying that command lines containing the strings “\${MAKE}” and “\$(MAKE)” are
24295 executed when the **-n** option is specified (GNU and System V)
- 24296 • Modifications of the meaning of internal macros when referencing libraries (BSD and others)
- 24297 • Using a single instance of the shell for all of the command lines of the target (BSD and others)
- 24298 • Allowing spaces as well as tabs to delimit command lines (BSD)
- 24299 • Adding C preprocessor-style “include” and “ifdef” constructs (System V, GNU, BSD, and
24300 others)
- 24301 • Remote execution of command lines (Sprite and others)
- 24302 • Specifying additional special targets (BSD, System V, and most others)

24303 Additionally, many vendors and research organizations have rethought the basic concepts of
24304 *make*, creating vastly extended, as well as completely new, syntaxes. Each of these versions of
24305 *make* fulfills the needs of a different community of users; it is unreasonable for this volume of
24306 IEEE Std 1003.1-2001 to require behavior that would be incompatible (and probably inferior) to
24307 historical practice for such a community.

24308 In similar circumstances, when the industry has enough sufficiently incompatible formats as to
24309 make them irreconcilable, this volume of IEEE Std 1003.1-2001 has followed one or both of two
24310 courses of action. Commands have been renamed (*cksum*, *echo*, and *pax*) and/or command line
24311 options have been provided to select the desired behavior (*grep*, *od*, and *pax*).

24312 Because the syntax specified for the *make* utility is, by and large, a subset of the syntaxes
24313 accepted by almost all versions of *make*, it was decided that it would be counter-productive to
24314 change the name. And since the makefile itself is a basic unit of portability, it would not be

24315 completely effective to reserve a new option letter, such as *make -P*, to achieve the portable
24316 behavior. Therefore, the special target **.POSIX** was added to the makefile, allowing users to
24317 specify “standard” behavior. This special target does not preclude extensions in the *make* utility,
24318 nor does it preclude such extensions being used by the makefile specifying the target; it does,
24319 however, preclude any extensions from being applied that could alter the behavior of previously
24320 valid syntax; such extensions must be controlled via command line options or new special
24321 targets. It is incumbent upon portable makefiles to specify the **.POSIX** special target in order to
24322 guarantee that they are not affected by local extensions.

24323 The portable version of *make* described in this reference page is not intended to be the state-of-
24324 the-art software generation tool and, as such, some newer and more leading-edge features have
24325 not been included. An attempt has been made to describe the portable makefile in a manner that
24326 does not preclude such extensions as long as they do not disturb the portable behavior described
24327 here.

24328 When the **-n** option is specified, it is always added to **MAKEFLAGS**. This allows a recursive
24329 *make -n target* to be used to see all of the action that would be taken to update *target*.

24330 The definition of **MAKEFLAGS** allows both the System V letter string and the BSD command line
24331 formats. The two formats are sufficiently different to allow implementations to support both
24332 without ambiguity.

24333 Early proposals stated that an “unquoted” number sign was treated as the start of a comment.
24334 The *make* utility does not pay any attention to quotes. A number sign starts a comment
24335 regardless of its surroundings.

24336 The text about “other implementation-defined pathnames may also be tried” in addition to
24337 **./makefile** and **./Makefile** is to allow such extensions as **SCCS/s.Makefile** and other variations.
24338 It was made an implementation-defined requirement (as opposed to unspecified behavior) to
24339 highlight surprising implementations that might select something unexpected like
24340 **/etc/Makefile**. XSI-conformant systems also try **./s.makefile**, **SCCS/s.makefile**, **./s.Makefile**,
24341 and **SCCS/s.Makefile**.

24342 Early proposals contained the macro **NPROC** as a means of specifying that *make* should use *n*
24343 processes to do the work required. While this feature is a valuable extension for many systems, it
24344 is not common usage and could require other non-trivial extensions to makefile syntax. This
24345 extension is not required by this volume of IEEE Std 1003.1-2001, but could be provided as a
24346 compatible extension. The macro **PARALLEL** is used by some historical systems with essentially
24347 the same meaning (but without using a name that is a common system limit value). It is
24348 suggested that implementors recognize the existing use of **NPROC** and/or **PARALLEL** as
24349 extensions to *make*.

24350 The default rules are based on System V. The default **CC=** value is **c99** instead of **cc** because this
24351 volume of IEEE Std 1003.1-2001 does not standardize the utility named **cc**. Thus, every
24352 conforming application would be required to define **CC=c99** to expect to run. There is no
24353 advantage conferred by the hope that the makefile might hit the “preferred” compiler because
24354 this cannot be guaranteed to work. Also, since the portable makescript can only use the **c99**
24355 options, no advantage is conferred in terms of what the script can do. It is a quality-of-
24356 implementation issue as to whether **c99** is as valuable as **cc**.

24357 The **-d** option to *make* is frequently used to produce debugging information, but is too
24358 implementation-defined to add to this volume of IEEE Std 1003.1-2001.

24359 The **-p** option is not passed in **MAKEFLAGS** on most historical implementations and to change
24360 this would cause many implementations to break without sufficiently increased portability.

24361 Commands that begin with a plus sign ('+') are executed even if the **-n** option is present. Based
24362 on the GNU version of *make*, the behavior of **-n** when the plus-sign prefix is encountered has
24363 been extended to apply to **-q** and **-t** as well. However, the System V convention of forcing
24364 command execution with **-n** when the command line of a target contains either of the strings
24365 "`$(MAKE)" or "${MAKE}"`" has not been adopted. This functionality appeared in early
24366 proposals, but the danger of this approach was pointed out with the following example of a
24367 portion of a makefile:

24368

```
subdir:  
24369     cd subdir; rm all_the_files; $(MAKE)
```

24370 The loss of the System V behavior in this case is well-balanced by the safety afforded to other
24371 makefiles that were not aware of this situation. In any event, the command line plus-sign prefix
24372 can provide the desired functionality.

24373 The double colon in the target rule format is supported in BSD systems to allow more than one
24374 target line containing the same target name to have commands associated with it. Since this is
24375 not functionality described in the SVID or XPG3 it has been allowed as an extension, but not
24376 mandated.

24377 The default rules are provided with text specifying that the built-in rules shall be the same as if
24378 the listed set were used. The intent is that implementations should be able to use the rules
24379 without change, but will be allowed to alter them in ways that do not affect the primary
24380 behavior.

24381 The best way to provide portable makefiles is to include all of the rules needed in the makefile
24382 itself. The rules provided use only features provided by other portions of this volume of
24383 IEEE Std 1003.1-2001. The default rules include rules for optional commands in this volume of
24384 IEEE Std 1003.1-2001. Only rules pertaining to commands that are provided are needed in the
24385 default set of an implementation.

24386 One point of discussion was whether to drop the default rules list from this volume of
24387 IEEE Std 1003.1-2001. They provide convenience, but do not enhance portability of applications.
24388 The prime benefit is in portability of users who wish to type *make command* and have the
24389 command build from a **command.c** file.

24390 The historical *MAKESHELL* feature was omitted. In some implementations it is used to let a user
24391 override the shell to be used to run *make* commands. This was confusing; for a portable *make*, the
24392 shell should be chosen by the makefile writer or specified on the *make* command line and not by
24393 a user running *make*.

24394 The *make* utilities in most historical implementations process the prerequisites of a target in left-
24395 to-right order, and the makefile format requires this. It supports the standard idiom used in
24396 many makefiles that produce *yacc* programs; for example:

24397

```
foo: y.tab.o lex.o main.o  
24398     $(CC) $(CFLAGS) -o $@ t.tab.o lex.o main.o
```

24399 In this example, if *make* chose any arbitrary order, the **lex.o** might not be made with the correct
24400 **y.tab.h**. Although there may be better ways to express this relationship, it is widely used
24401 historically. Implementations that desire to update prerequisites in parallel should require an
24402 explicit extension to *make* or the makefile format to accomplish it, as described previously.

24403 The algorithm for determining a new entry for target rules is partially unspecified. Some
24404 historical *makes* allow blank, empty, or comment lines within the collection of commands
24405 marked by leading <tab>s. A conforming makefile must ensure that each command starts with
24406 a <tab>, but implementations are free to ignore blank, empty, and comment lines without
24407 triggering the start of a new entry.

24408 The ASYNCHRONOUS EVENTS section includes having SIGTERM and SIGHUP, along with
24409 the more traditional SIGINT and SIGQUIT, remove the current target unless directed not to do
24410 so. SIGTERM and SIGHUP were added to parallel other utilities that have historically cleaned
24411 up their work as a result of these signals. When *make* receives any signal other than SIGQUIT, it
24412 is required to resend itself the signal it received so that it exits with a status that reflects the
24413 signal. The results from SIGQUIT are partially unspecified because, on systems that create **core**
24414 files upon receipt of SIGQUIT, the **core** from *make* would conflict with a **core** file from the
24415 command that was running when the SIGQUIT arrived. The main concern was to prevent
24416 damaged files from appearing up-to-date when *make* is rerun.

24417 The **.PRECIOUS** special target was extended to affect all targets globally (by specifying no
24418 prerequisites). The **.IGNORE** and **.SILENT** special targets were extended to allow prerequisites;
24419 it was judged to be more useful in some cases to be able to turn off errors or echoing for a list of
24420 targets than for the entire makefile. These extensions to *make* in System V were made to match
24421 historical practice from the BSD *make*.

24422 Macros are not exported to the environment of commands to be run. This was never the case in
24423 any historical *make* and would have serious consequences. The environment is the same as the
24424 environment to *make* except that **MAKEFLAGS** and macros defined on the *make* command line
24425 are added.

24426 Some implementations do not use *system()* for all command lines, as required by the portable
24427 makefile format; as a performance enhancement, they select lines without shell metacharacters
24428 for direct execution by *execve()*. There is no requirement that *system()* be used specifically, but
24429 merely that the same results be achieved. The metacharacters typically used to bypass the direct
24430 *execve()* execution have been any of:

24431 = | ^ () ; & < > * ? [] : \$ ' ' " \ \n

24432 The default in some advanced versions of *make* is to group all the command lines for a target and
24433 execute them using a single shell invocation; the System V method is to pass each line
24434 individually to a separate shell. The single-shell method has the advantages in performance and
24435 the lack of a requirement for many continued lines. However, converting to this newer method
24436 has caused portability problems with many historical makefiles, so the behavior with the POSIX
24437 makefile is specified to be the same as that of System V. It is suggested that the special target
24438 **.ONESHELL** be used as an implementation extension to achieve the single-shell grouping for a
24439 target or group of targets.

24440 Novice users of *make* have had difficulty with the historical need to start commands with a
24441 <tab>. Since it is often difficult to discern differences between <tab>s and <space>s on terminals
24442 or printed listings, confusing bugs can arise. In early proposals, an attempt was made to correct
24443 this problem by allowing leading <blank>s instead of <tab>s. However, implementors reported
24444 many makefiles that failed in subtle ways following this change, and it is difficult to implement
24445 a *make* that unambiguously can differentiate between macro and command lines. There is
24446 extensive historical practice of allowing leading spaces before macro definitions. Forcing macro
24447 lines into column 1 would be a significant backwards-compatibility problem for some makefiles.
24448 Therefore, historical practice was restored.

24449 The System V INCLUDE feature was considered, but not included. This would treat a line that
24450 began in the first column and contained INCLUDE <filename> as an indication to read <filename>
24451 at that point in the makefile. This is difficult to use in a portable way, and it raises concerns
24452 about nesting levels and diagnostics. System V, BSD, GNU, and others have used different
24453 methods for including files.

24454 The System V dynamic dependency feature was not included. It would support:

24455 cat: \$\$@.c
24456 that would expand to;
24457 cat: cat.c
24458 This feature exists only in the new version of System V *make* and, while useful, is not in wide
24459 usage. This means that macros are expanded twice for prerequisites: once at makefile parse time
24460 and once at target update time.
24461 Consideration was given to adding metarules to the POSIX *make*. This would make %.o: %.c the
24462 same as .c.o:. This is quite useful and available from some vendors, but it would cause too many
24463 changes to this *make* to support. It would have introduced rule chaining and new substitution
24464 rules. However, the rules for target names have been set to reserve the '%' and '"' characters.
24465 These are traditionally used to implement metarules and quoting of target names, respectively.
24466 Implementors are strongly encouraged to use these characters only for these purposes.
24467 A request was made to extend the suffix delimiter character from a period to any character. The
24468 metarules feature in newer *makes* solves this problem in a more general way. This volume of
24469 IEEE Std 1003.1-2001 is staying with the more conservative historical definition.
24470 The standard output format for the -p option is not described because it is primarily a
24471 debugging option and because the format is not generally useful to programs. In historical
24472 implementations the output is not suitable for use in generating makefiles. The -p format has
24473 been variable across historical implementations. Therefore, the definition of -p was only to
24474 provide a consistently named option for obtaining *make* script debugging information.
24475 Some historical implementations have not cleared the suffix list with -r.
24476 Implementations should be aware that some historical applications have intermixed *target_name*
24477 and *macro=value* operands on the command line, expecting that all of the macros are processed
24478 before any of the targets are dealt with. Conforming applications do not do this, but some
24479 backwards-compatibility support may be warranted.
24480 Empty inference rules are specified with a semicolon command rather than omitting all
24481 commands, as described in an early proposal. The latter case has no traditional meaning and is
24482 reserved for implementation extensions, such as in GNU *make*.
24483 **FUTURE DIRECTIONS**
24484 None.
24485 **SEE ALSO**
24486 Chapter 2 (on page 29), *ar*, *c99*, *get*, *lex*, *sccs*, *sh*, *yacc*, the System Interfaces volume of
24487 IEEE Std 1003.1-2001, *exec*, *system()*
24488 **CHANGE HISTORY**
24489 First released in Issue 2.
24490 **Issue 5**
24491 The FUTURE DIRECTIONS section is added.
24492 **Issue 6**
24493 This utility is marked as part of the Software Development Utilities option.
24494 The Open Group Corrigendum U029/1 is applied, correcting a typographical error in the
24495 SPECIAL TARGETS section.
24496 In the ENVIRONMENT VARIABLES section, the *PROJECTDIR* description is updated from
24497 "otherwise, the home directory of a user of that name is examined" to "otherwise, the value of
24498 *PROJECTDIR* is treated as a user name and that user's initial working directory is examined".

- 24499 It is specified whether the command line is related to the makefile or to the *make* command, and
24500 the macro processing rules are updated to align with the IEEE P1003.2b draft standard.
- 24501 The normative text is reworded to avoid use of the term “must” for application requirements.
- 24502 PASC Interpretation 1003.2 #193 is applied.

24503 NAME

24504 man — display system documentation

24505 SYNOPSIS

24506 man [-k] name...

24507 DESCRIPTION

24508 The *man* utility shall write information about each of the *name* operands. If *name* is the name of a standard utility, *man* at a minimum shall write a message describing the syntax used by the standard utility, its options, and operands. If more information is available, the *man* utility shall provide it in an implementation-defined manner.

24512 An implementation may provide information for values of *name* other than the standard utilities. Standard utilities that are listed as optional and that are not supported by the implementation either shall cause a brief message indicating that fact to be displayed or shall cause a full display of information as described previously.

24516 OPTIONS

24517 The *man* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section 12.2, Utility Syntax Guidelines.

24519 The following option shall be supported:

24520 **-k** Interpret *name* operands as keywords to be used in searching a utilities summary database that contains a brief purpose entry for each standard utility and write lines from the summary database that match any of the keywords. The keyword search shall produce results that are the equivalent of the output of the following command:

```
24524 grep -Ei '  
24525     name  
24526     name  
24527     ...  
24528     ' summary-database
```

24529 This assumes that the *summary-database* is a text file with a single entry per line; this organization is not required and the example using *grep -Ei* is merely illustrative of the type of search intended. The purpose entry to be included in the database shall consist of a terse description of the purpose of the utility.

24533 OPERANDS

24534 The following operand shall be supported:

24535 *name* A keyword or the name of a standard utility. When **-k** is not specified and *name* does not represent one of the standard utilities, the results are unspecified.

24537 STDIN

24538 Not used.

24539 INPUT FILES

24540 None.

24541 ENVIRONMENT VARIABLES

24542 The following environment variables shall affect the execution of *man*:

24543 *LANG* Provide a default value for the internationalization variables that are unset or null. (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2, Internationalization Variables for the precedence of internationalization variables used to determine the values of locale categories.)

24547	<i>LC_ALL</i>	If set to a non-empty string value, override the values of all the other internationalization variables.
24549	<i>LC_CTYPE</i>	Determine the locale for the interpretation of sequences of bytes of text data as characters (for example, single-byte as opposed to multi-byte characters in arguments and in the summary database). The value of <i>LC_CTYPE</i> need not affect the format of the information written about the <i>name</i> operands.
24553	<i>LC_MESSAGES</i>	Determine the locale that should be used to affect the format and contents of diagnostic messages written to standard error and informative messages written to standard output.
24557 XSI	<i>NLSPATH</i>	Determine the location of message catalogs for the processing of <i>LC_MESSAGES</i> .
24558	<i>PAGER</i>	Determine an output filtering command for writing the output to a terminal. Any string acceptable as a <i>command_string</i> operand to the <i>sh -c</i> command shall be valid. When standard output is a terminal device, the reference page output shall be piped through the command. If the <i>PAGER</i> variable is null or not set, the command shall be either <i>more</i> or another paginator utility documented in the system documentation.
24564	ASYNCHRONOUS EVENTS	
24565		Default.
24566	STDOUT	
24567		The <i>man</i> utility shall write text describing the syntax of the utility <i>name</i> , its options and its operands, or, when -k is specified, lines from the summary database. The format of this text is implementation-defined.
24570	STDERR	
24571		The standard error shall be used only for diagnostic messages.
24572	OUTPUT FILES	
24573		None.
24574	EXTENDED DESCRIPTION	
24575		None.
24576	EXIT STATUS	
24577		The following exit values shall be returned:
24578	0	Successful completion.
24579	>0	An error occurred.
24580	CONSEQUENCES OF ERRORS	
24581		Default.
24582	APPLICATION USAGE	
24583		None.
24584	EXAMPLES	
24585		None.
24586	RATIONALE	
24587		It is recognized that the <i>man</i> utility is only of minimal usefulness as specified. The opinion of the standard developers was strongly divided as to how much or how little information <i>man</i> should be required to provide. They considered, however, that the provision of some portable way of accessing documentation would aid user portability. The arguments against a fuller

24591 specification were:

- 24592 • Large quantities of documentation should not be required on a system that does not have
24593 excess disk space.
- 24594 • The current manual system does not present information in a manner that greatly aids user
24595 portability.
- 24596 • A “better help system” is currently an area in which vendors feel that they can add value to
24597 their POSIX implementations.

24598 The **-f** option was considered, but due to implementation differences, it was not included in this
24599 volume of IEEE Std 1003.1-2001.

24600 The description was changed to be more specific about what has to be displayed for a utility.
24601 The standard developers considered it insufficient to allow a display of only the synopsis
24602 without giving a short description of what each option and operand does.

24603 The “purpose” entry to be included in the database can be similar to the section title (less the
24604 numeric prefix) from this volume of IEEE Std 1003.1-2001 for each utility. These titles are similar
24605 to those used in historical systems for this purpose.

24606 See *mailx* for rationale concerning the default paginator.

24607 The caveat in the *LC_CTYPE* description was added because it is not a requirement that an
24608 implementation provide reference pages for all of its supported locales on each system;
24609 changing *LC_CTYPE* does not necessarily translate the reference page into another language.
24610 This is equivalent to the current state of *LC_MESSAGES* in IEEE Std 1003.1-2001—locale-specific
24611 messages are not yet a requirement.

24612 The historical *MANPATH* variable is not included in POSIX because no attempt is made to
24613 specify naming conventions for reference page files, nor even to mandate that they are files at
24614 all. On some implementations they could be a true database, a hypertext file, or even fixed
24615 strings within the *man* executable. The standard developers considered the portability of
24616 reference pages to be outside their scope of work. However, users should be aware that
24617 *MANPATH* is implemented on a number of historical systems and that it can be used to tailor
24618 the search pattern for reference pages from the various categories (utilities, functions, file
24619 formats, and so on) when the system administrator reveals the location and conventions for
24620 reference pages on the system.

24621 The keyword search can rely on at least the text of the section titles from these utility
24622 descriptions, and the implementation may add more keywords. The term “section titles” refers
24623 to the strings such as:

24624 man – Display system documentation
24625 ps – Report process status

24626 FUTURE DIRECTIONS

24627 None.

24628 SEE ALSO

24629 *more*

24630 CHANGE HISTORY

24631 First released in Issue 4.

24632 **Issue 5**

24633 The FUTURE DIRECTIONS section is added.

24634 NAME

24635 mesg — permit or deny messages

24636 SYNOPSIS

24637 UP mesg [y|n]

24638

24639 DESCRIPTION

24640 The *mesg* utility shall control whether other users are allowed to send messages via *write*, *talk*, or
24641 other utilities to a terminal device. The terminal device affected shall be determined by searching
24642 for the first terminal in the sequence of devices associated with standard input, standard output,
24643 and standard error, respectively. With no arguments, *mesg* shall report the current state without
24644 changing it. Processes with appropriate privileges may be able to send messages to the terminal
24645 independent of the current state.

24646 OPTIONS

24647 None.

24648 OPERANDS

24649 The following operands shall be supported in the POSIX locale:

24650 *y* Grant permission to other users to send messages to the terminal device.

24651 *n* Deny permission to other users to send messages to the terminal device.

24652 STDIN

24653 Not used.

24654 INPUT FILES

24655 None.

24656 ENVIRONMENT VARIABLES

24657 The following environment variables shall affect the execution of *mesg*:

24658 *LANG* Provide a default value for the internationalization variables that are unset or null.
24659 (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2,
24660 Internationalization Variables for the precedence of internationalization variables
24661 used to determine the values of locale categories.)

24662 *LC_ALL* If set to a non-empty string value, override the values of all the other
24663 internationalization variables.

24664 *LC_CTYPE* Determine the locale for the interpretation of sequences of bytes of text data as
24665 characters (for example, single-byte as opposed to multi-byte characters in
24666 arguments).

24667 *LC_MESSAGES*

24668 Determine the locale that should be used to affect the format and contents of
24669 diagnostic messages written (by *mesg*) to standard error.

24670 XSI *NLSPATH* Determine the location of message catalogs for the processing of *LC_MESSAGES*.

24671 ASYNCHRONOUS EVENTS

24672 Default.

24673 STDOUT

24674 If no operand is specified, *mesg* shall display the current terminal state in an unspecified format.

24675 STDERR

24676 The standard error shall be used only for diagnostic messages.

24677 OUTPUT FILES

24678 None.

24679 EXTENDED DESCRIPTION

24680 None.

24681 EXIT STATUS

24682 The following exit values shall be returned:

24683 0 Receiving messages is allowed.

24684 1 Receiving messages is not allowed.

24685 >1 An error occurred.

24686 CONSEQUENCES OF ERRORS

24687 Default.

24688 APPLICATION USAGE

24689 The mechanism by which the message status of the terminal is changed is unspecified.
24690 Therefore, unspecified actions may cause the status of the terminal to change after *mesg* has
24691 successfully completed. These actions may include, but are not limited to: another invocation of
24692 the *mesg* utility, login procedures; invocation of the *stty* utility, invocation of the *chmod* utility or
24693 *chmod()* function, and so on.

24694 EXAMPLES

24695 None.

24696 RATIONALE

24697 The terminal changed by *mesg* is that associated with the standard input, output, or error, rather
24698 than the controlling terminal for the session. This is because users logged in more than once
24699 should be able to change any of their login terminals without having to stop the job running in
24700 those sessions. This is not a security problem involving the terminals of other users because
24701 appropriate privileges would be required to affect the terminal of another user.

24702 The method of checking each of the first three file descriptors in sequence until a terminal is
24703 found was adopted from System V.

24704 The file */dev/tty* is not specified for the terminal device because it was thought to be too
24705 restrictive. Typical environment changes for the *n* operand are that write permissions are
24706 removed for *others* and *group* from the appropriate device. It was decided to leave the actual
24707 description of what is done as unspecified because of potential differences between
24708 implementations.

24709 The format for standard output is unspecified because of differences between historical
24710 implementations. This output is generally not useful to shell scripts (they can use the exit
24711 status), so exact parsing of the output is unnecessary.

24712 FUTURE DIRECTIONS

24713 None.

24714 SEE ALSO

24715 *talk*, *write*

24716 CHANGE HISTORY

24717 First released in Issue 2.

24718 Issue 6

24719 This utility is marked as part of the User Portability Utilities option.

24720 NAME

24721 mkdir — make directories

24722 SYNOPSIS

24723 `mkdir [-p][-m mode] dir...`

24724 DESCRIPTION

24725 The *mkdir* utility shall create the directories specified by the operands, in the order specified.

24726 For each *dir* operand, the *mkdir* utility shall perform actions equivalent to the *mkdir()* function
24727 defined in the System Interfaces volume of IEEE Std 1003.1-2001, called with the following
24728 arguments:

- 24729 1. The *dir* operand is used as the *path* argument.
- 24730 2. The value of the bitwise-inclusive OR of S_IRWXU, S_IRWXG, and S_IRWXO is used as
24731 the *mode* argument. (If the **-m** option is specified, the *mode* option-argument overrides this
24732 default.)

24733 OPTIONS

24734 The *mkdir* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section
24735 12.2, Utility Syntax Guidelines.

24736 The following options shall be supported:

24737 **-m mode** Set the file permission bits of the newly-created directory to the specified *mode*
24738 value. The *mode* option-argument shall be the same as the *mode* operand defined
24739 for the *chmod* utility. In the *symbolic_mode* strings, the *op* characters '+' and '-'
24740 shall be interpreted relative to an assumed initial mode of *a=rwx*; '+' shall add
24741 permissions to the default mode, '-' shall delete permissions from the default
24742 mode.

24743 **-p** Create any missing intermediate pathname components.

24744 For each *dir* operand that does not name an existing directory, effects equivalent to
24745 those caused by the following command shall occur:

24746 `mkdir -p -m $(umask -S),u+wx $(dirname dir) &&`
24747 `mkdir [-m mode] dir`

24748 where the **-m mode** option represents that option supplied to the original
24749 invocation of *mkdir*, if any.

24750 Each *dir* operand that names an existing directory shall be ignored without error.

24751 OPERANDS

24752 The following operand shall be supported:

24753 *dir* A pathname of a directory to be created.

24754 STDIN

24755 Not used.

24756 INPUT FILES

24757 None.

24758 ENVIRONMENT VARIABLES

24759 The following environment variables shall affect the execution of *mkdir*:

24760 **LANG** Provide a default value for the internationalization variables that are unset or null.
24761 (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2,
24762 Internationalization Variables for the precedence of internationalization variables

24763		used to determine the values of locale categories.)
24764	<i>LC_ALL</i>	If set to a non-empty string value, override the values of all the other internationalization variables.
24766	<i>LC_CTYPE</i>	Determine the locale for the interpretation of sequences of bytes of text data as characters (for example, single-byte as opposed to multi-byte characters in arguments).
24769	<i>LC_MESSAGES</i>	Determine the locale that should be used to affect the format and contents of diagnostic messages written to standard error.
24772 XSI	<i>NLSPATH</i>	Determine the location of message catalogs for the processing of <i>LC_MESSAGES</i> .
24773	ASYNCHRONOUS EVENTS	
24774		Default.
24775	STDOUT	
24776		Not used.
24777	STDERR	
24778		The standard error shall be used only for diagnostic messages.
24779	OUTPUT FILES	
24780		None.
24781	EXTENDED DESCRIPTION	
24782		None.
24783	EXIT STATUS	
24784		The following exit values shall be returned:
24785	0	All the specified directories were created successfully or the -p option was specified and all the specified directories now exist.
24786		
24787	>0	An error occurred.
24788	CONSEQUENCES OF ERRORS	
24789		Default.
24790	APPLICATION USAGE	
24791		The default file mode for directories is <i>a=rwx</i> (777 on most systems) with selected permissions removed in accordance with the file mode creation mask. For intermediate pathname components created by <i>mkdir</i> , the mode is the default modified by <i>u+wx</i> so that the subdirectories can always be created regardless of the file mode creation mask; if different ultimate permissions are desired for the intermediate directories, they can be changed afterwards with <i>chmod</i> .
24792		
24793		
24794		
24795		
24796		
24797		Note that some of the requested directories may have been created even if an error occurs.
24798	EXAMPLES	
24799		None.
24800	RATIONALE	
24801		The System V -m option was included to control the file mode.
24802		The System V -p option was included to create any needed intermediate directories and to complement the functionality provided by <i>rmdir</i> for removing directories in the path prefix as they become empty. Because no error is produced if any path component already exists, the -p option is also useful to ensure that a particular directory exists.
24803		
24804		
24805		

24806 The functionality of *mkdir* is described substantially through a reference to the *mkdir()* function
24807 in the System Interfaces volume of IEEE Std 1003.1-2001. For example, by default, the mode of
24808 the directory is affected by the file mode creation mask in accordance with the specified
24809 behavior of the *mkdir()* function. In this way, there is less duplication of effort required for
24810 describing details of the directory creation.

24811 **FUTURE DIRECTIONS**

24812 None.

24813 **SEE ALSO**

24814 *chmod*, *rm*, *rmdir*, *umask*, the System Interfaces volume of IEEE Std 1003.1-2001, *mkdir()*

24815 **CHANGE HISTORY**

24816 First released in Issue 2.

24817 **Issue 5**

24818 The FUTURE DIRECTIONS section is added.

24819 NAME

24820 mkfifo — make FIFO special files

24821 SYNOPSIS

24822 mkfifo [-m mode] file...

24823 DESCRIPTION

24824 The *mkfifo* utility shall create the FIFO special files specified by the operands, in the order
24825 specified.

24826 For each *file* operand, the *mkfifo* utility shall perform actions equivalent to the *mkfifo()* function
24827 defined in the System Interfaces volume of IEEE Std 1003.1-2001, called with the following
24828 arguments:

- 24829 1. The *file* operand is used as the *path* argument.
- 24830 2. The value of the bitwise-inclusive OR of S_IRUSR, S_IWUSR, S_IRGRP, S_IWGRP,
24831 S_IROTH, and S_IWOTH is used as the *mode* argument. (If the **-m** option is specified, the
24832 value of the *mkfifo()* *mode* argument is unspecified, but the FIFO shall at no time have
24833 permissions less restrictive than the **-m** *mode* option-argument.)

24834 OPTIONS

24835 The *mkfifo* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section
24836 12.2, Utility Syntax Guidelines.

24837 The following option shall be supported:

24838 **-m mode** Set the file permission bits of the newly-created FIFO to the specified *mode* value.
24839 The *mode* option-argument shall be the same as the *mode* operand defined for the
24840 *chmod* utility. In the *symbolic_mode* strings, the *op* characters '+' and '-' shall be
24841 interpreted relative to an assumed initial mode of *a=rw*.

24842 OPERANDS

24843 The following operand shall be supported:

24844 *file* A pathname of the FIFO special file to be created.

24845 STDIN

24846 Not used.

24847 INPUT FILES

24848 None.

24849 ENVIRONMENT VARIABLES

24850 The following environment variables shall affect the execution of *mkfifo*:

24851 **LANG** Provide a default value for the internationalization variables that are unset or null.
24852 (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2,
24853 Internationalization Variables for the precedence of internationalization variables
24854 used to determine the values of locale categories.)

24855 **LC_ALL** If set to a non-empty string value, override the values of all the other
24856 internationalization variables.

24857 **LC_CTYPE** Determine the locale for the interpretation of sequences of bytes of text data as
24858 characters (for example, single-byte as opposed to multi-byte characters in
24859 arguments).

24860 *LC_MESSAGES*

24861 Determine the locale that should be used to affect the format and contents of
24862 diagnostic messages written to standard error.

24863 XSI **NLSPATH** Determine the location of message catalogs for the processing of *LC_MESSAGES*.

24864 **ASYNCHRONOUS EVENTS**

24865 Default.

24866 **STDOUT**

24867 Not used.

24868 **STDERR**

24869 The standard error shall be used only for diagnostic messages.

24870 **OUTPUT FILES**

24871 None.

24872 **EXTENDED DESCRIPTION**

24873 None.

24874 **EXIT STATUS**

24875 The following exit values shall be returned:

24876 0 All the specified FIFO special files were created successfully.

24877 >0 An error occurred.

24878 **CONSEQUENCES OF ERRORS**

24879 Default.

24880 **APPLICATION USAGE**

24881 None.

24882 **EXAMPLES**

24883 None.

24884 **RATIONALE**

24885 This utility was added to permit shell applications to create FIFO special files.

24886 The **-m** option was added to control the file mode, for consistency with the similar functionality
24887 provided by the *mkdir* utility.

24888 Early proposals included a **-p** option similar to the *mkdir -p* option that created intermediate
24889 directories leading up to the FIFO specified by the final component. This was removed because
24890 it is not commonly needed and is not common practice with similar utilities.

24891 The functionality of *mkfifo* is described substantially through a reference to the *mkfifo()* function
24892 in the System Interfaces volume of IEEE Std 1003.1-2001. For example, by default, the mode of
24893 the FIFO file is affected by the file mode creation mask in accordance with the specified behavior
24894 of the *mkfifo()* function. In this way, there is less duplication of effort required for describing
24895 details of the file creation.

24896 **FUTURE DIRECTIONS**

24897 None.

24898 **SEE ALSO**

24899 *chmod*, *umask*, the System Interfaces volume of IEEE Std 1003.1-2001, *mkfifo()*

24900 **CHANGE HISTORY**

24901 First released in Issue 3.

24902 NAME

24903 more — display files on a page-by-page basis

24904 SYNOPSIS

24905 UP more [-ceisu][-n number][-p command][-t tagstring][file ...]

24906

24907 DESCRIPTION

24908 The *more* utility shall read files and either write them to the terminal on a page-by-page basis or
24909 filter them to standard output. If standard output is not a terminal device, all input files shall be
24910 copied to standard output in their entirety, without modification, except as specified for the **-s**
24911 option. If standard output is a terminal device, the files shall be written a number of lines (one
24912 screenful) at a time under the control of user commands. See the EXTENDED DESCRIPTION
24913 section.

24914 Certain block-mode terminals do not have all the capabilities necessary to support the complete
24915 *more* definition; they are incapable of accepting commands that are not terminated with a
24916 <newline>. Implementations that support such terminals shall provide an operating mode to
24917 *more* in which all commands can be terminated with a <newline> on those terminals. This mode:

- 24918 • Shall be documented in the system documentation
- 24919 • Shall, at invocation, inform the user of the terminal deficiency that requires the <newline>
24920 usage and provide instructions on how this warning can be suppressed in future invocations
- 24921 • Shall not be required for implementations supporting only fully capable terminals
- 24922 • Shall not affect commands already requiring <newline>s
- 24923 • Shall not affect users on the capable terminals from using *more* as described in this volume of
24924 IEEE Std 1003.1-2001

24925 OPTIONS

24926 The *more* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section
24927 12.2, Utility Syntax Guidelines.

24928 The following options shall be supported:

- 24929 **-c** If a screen is to be written that has no lines in common with the current screen, or
24930 *more* is writing its first screen, *more* shall not scroll the screen, but instead shall
24931 redraw each line of the screen in turn, from the top of the screen to the bottom. In
24932 addition, if *more* is writing its first screen, the screen shall be cleared. This option
24933 may be silently ignored on devices with insufficient terminal capabilities.
- 24934 **-e** By default, *more* shall exit immediately after writing the last line of the last file in
24935 the argument list. If the **-e** option is specified:
 - 24936 1. If there is only a single file in the argument list and that file was completely
24937 displayed on a single screen, *more* shall exit immediately after writing the last
24938 line of that file.
 - 24939 2. Otherwise, *more* shall exit only after reaching end-of-file on the last file in the
24940 argument list twice without an intervening operation. See the EXTENDED
24941 DESCRIPTION section.
- 24942 **-i** Perform pattern matching in searches without regard to case; see the Base
24943 Definitions volume of IEEE Std 1003.1-2001, Section 9.2, Regular Expression
24944 General Requirements.

- 24945 **-n** *number* Specify the number of lines per screenful. The *number* argument is a positive
24946 decimal integer. The **-n** option shall override any values obtained from any other
24947 source.
- 24948 **-p** *command* Each time a screen from a new file is displayed or redisplayed (including as a
24949 result of *more* commands; for example, **:p**), execute the *more* command(s) in the
24950 command arguments in the order specified, as if entered by the user after the first
24951 screen has been displayed. No intermediate results shall be displayed (that is, if the
24952 command is a movement to a screen different from the normal first screen, only
24953 the screen resulting from the command shall be displayed.) If any of the
24954 commands fail for any reason, an informational message to this effect shall be
24955 written, and no further commands specified using the **-p** option shall be executed
24956 for this file.
- 24957 **-s** Behave as if consecutive empty lines were a single empty line.
- 24958 **-t** *tagstring* Write the screenful of the file containing the tag named by the *tagstring* argument.
24959 See the *ctags* utility. The tags feature represented by **-t** *tagstring* and the **:t**
24960 command is optional. It shall be provided on any system that also provides a
24961 conforming implementation of *ctags*; otherwise, the use of **-t** produces undefined
24962 results.
- 24963 The filename resulting from the **-t** option shall be logically added as a prefix to the
24964 list of command line files, as if specified by the user. If the tag named by the
24965 *tagstring* argument is not found, it shall be an error, and *more* shall take no further
24966 action.
- 24967 If the tag specifies a line number, the first line of the display shall contain the
24968 beginning of that line. If the tag specifies a pattern, the first line of the display shall
24969 contain the beginning of the matching text from the first line of the file that
24970 contains that pattern. If the line does not exist in the file or matching text is not
24971 found, an informational message to this effect shall be displayed, and *more* shall
24972 display the default screen as if **-t** had not been specified.
- 24973 If both the **-t** *tagstring* and **-p** *command* options are given, the **-t** *tagstring* shall be
24974 processed first; that is, the file and starting line for the display shall be as specified
24975 by **-t**, and then the **-p** *more* command shall be executed. If the line (matching text)
24976 specified by the **-t** command does not exist (is not found), no **-p** *more* command
24977 shall be executed for this file at any time.
- 24978 **-u** Treat a **<backspace>** as a printable control character, displayed as an
24979 implementation-defined character sequence (see the EXTENDED DESCRIPTION
24980 section), suppressing backspacing and the special handling that produces
24981 underlined or standout mode text on some terminal types. Also, do not ignore a
24982 **<carriage-return>** at the end of a line.

24983 OPERANDS

24984 The following operand shall be supported:

- 24985 **file** A pathname of an input file. If no *file* operands are specified, the standard input
24986 shall be used. If a *file* is '**-**', the standard input shall be read at that point in the
24987 sequence.

24988 STDIN

24989 The standard input shall be used only if no *file* operands are specified, or if a *file* operand is '**-**'.

24990 INPUT FILES

24991 The input files being examined shall be text files. If standard output is a terminal, standard error
 24992 shall be used to read commands from the user. If standard output is a terminal, standard error is
 24993 not readable, and command input is needed, *more* may attempt to obtain user commands from
 24994 the controlling terminal (for example, `/dev/tty`); otherwise, *more* shall terminate with an error
 24995 indicating that it was unable to read user commands. If standard output is not a terminal, no
 24996 error shall result if standard error cannot be opened for reading.

24997 ENVIRONMENT VARIABLES

24998 The following environment variables shall affect the execution of *more*:

24999 **COLUMNS** Override the system-selected horizontal display line size. See the Base Definitions
 25000 volume of IEEE Std 1003.1-2001, Chapter 8, Environment Variables for valid values
 25001 and results when it is unset or null.

25002 **EDITOR** Used by the **v** command to select an editor. See the EXTENDED DESCRIPTION
 25003 section.

25004 **LANG** Provide a default value for the internationalization variables that are unset or null.
 25005 (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2,
 25006 Internationalization Variables for the precedence of internationalization variables
 25007 used to determine the values of locale categories.)

25008 **LC_ALL** If set to a non-empty string value, override the values of all the other
 25009 internationalization variables.

25010 **LC_COLLATE**

25011 Determine the locale for the behavior of ranges, equivalence classes, and multi-
 25012 character collating elements within regular expressions.

25013 **LC_CTYPE** Determine the locale for the interpretation of sequences of bytes of text data as
 25014 characters (for example, single-byte as opposed to multi-byte characters in
 25015 arguments and input files) and the behavior of character classes within regular
 25016 expressions.

25017 **LC_MESSAGES**

25018 Determine the locale that should be used to affect the format and contents of
 25019 diagnostic messages written to standard error and informative messages written to
 25020 standard output.

25021 XSI **NLSPATH** Determine the location of message catalogs for the processing of **LC_MESSAGES**.

25022 **LINES** Override the system-selected vertical screen size, used as the number of lines in a
 25023 screenful. See the Base Definitions volume of IEEE Std 1003.1-2001, Chapter 8,
 25024 Environment Variables for valid values and results when it is unset or null. The **-n**
 25025 option shall take precedence over the **LINES** variable for determining the number
 25026 of lines in a screenful.

25027 **MORE** Determine a string containing options described in the OPTIONS section preceded
 25028 with hyphens and <blank>-separated as on the command line. Any command line
 25029 options shall be processed after those in the **MORE** variable, as if the command
 25030 line were:

25031 *more \$MORE options operands*

25032 The **MORE** variable shall take precedence over the **TERM** and **LINES** variables for
 25033 determining the number of lines in a screenful.

25034 **TERM** Determine the name of the terminal type. If this variable is unset or null, an
25035 unspecified default terminal type is used.

25036 ASYNCHRONOUS EVENTS

25037 Default.

25038 STDOUT

25039 The standard output shall be used to write the contents of the input files.

25040 STDERR

The standard error shall be used for diagnostic messages and user commands (see the INPUT FILES section), and, if standard output is a terminal device, to write a prompting string. The prompting string shall appear on the screen line below the last line of the file displayed in the current screenful. The prompt shall contain the name of the file currently being examined and shall contain an end-of-file indication and the name of the next file, if any, when prompting at the end-of-file. If an error or informational message is displayed, it is unspecified whether it is contained in the prompt. If it is not contained in the prompt, it shall be displayed and then the user shall be prompted for a continuation character, at which point another message or the user prompt may be displayed. The prompt is otherwise unspecified. It is unspecified whether informational messages are written for other user commands.

25051 OUTPUT FILES

25052 None.

25053 EXTENDED DESCRIPTION

The following section describes the behavior of *more* when the standard output is a terminal device. If the standard output is not a terminal device, no options other than **-s** shall have any effect, and all input files shall be copied to standard output otherwise unmodified, at which time *more* shall exit without further action.

25058 The number of lines available per screen shall be determined by the **-n** option, if present, or by
25059 examining values in the environment (see the ENVIRONMENT VARIABLES section). If neither
25060 method yields a number, an unspecified number of lines shall be used.

The maximum number of lines written shall be one less than this number, because the screen line after the last line written shall be used to write a user prompt and user input. If the number of lines in the screen is less than two, the results are undefined. It is unspecified whether user input is permitted to be longer than the remainder of the single line where the prompt has been written.

25066 The number of columns available per line shall be determined by examining values in the
25067 environment (see the ENVIRONMENT VARIABLES section), with a default value as described
25068 in the Base Definitions volume of IEEE Std 1003.1-2001, Chapter 8, Environment Variables.

25069 Lines that are longer than the display shall be folded; the length at which folding occurs is
25070 unspecified, but should be appropriate for the output device. Folding may occur between glyphs
25071 of single characters that take up multiple display columns.

When standard output is a terminal and **-u** is not specified, *more* shall treat <backspace>s and <carriage-return>s specially:

- A character, followed first by a sequence of n <backspace>s (where n is the same as the number of column positions that the character occupies), then by n underscore characters ('_'), shall cause that character to be written as underlined text, if the terminal type supports that. The n underscore characters, followed first by n <backspace>s, then any character with n column positions, shall also cause that character to be written as underlined text, if the terminal type supports that.

- 25080 • A sequence of n <backspace>s (where n is the same as the number of column positions that
25081 the previous character occupies) that appears between two identical printable characters
25082 shall cause the first of those two characters to be written as emboldened text (that is, visually
25083 brighter, standout mode, or inverse-video mode), if the terminal type supports that, and the
25084 second to be discarded. Immediately subsequent occurrences of <backspace>/character pairs
25085 for that same character shall also be discarded. (For example, the sequence "a\b\ba\b\ba\b" is
25086 interpreted as a single emboldened 'a'.)
- 25087 • The *more* utility shall logically discard all other <backspace>s from the line as well as the
25088 character which precedes them, if any.
- 25089 • A <carriage-return> at the end of a line shall be ignored, rather than being written as a non-
25090 printable character, as described in the next paragraph.

25091 It is implementation-defined how other non-printable characters are written. Implementations
25092 should use the same format that they use for the *ex print* command; see the OPTIONS section
25093 within the *ed* utility. It is unspecified whether a multi-column character shall be separated if it
25094 crosses a display line boundary; it shall not be discarded. The behavior is unspecified if the
25095 number of columns on the display is less than the number of columns any single character in the
25096 line being displayed would occupy.

25097 When each new file is displayed (or redisplayed), *more* shall write the first screen of the file.
25098 Once the initial screen has been written, *more* shall prompt for a user command. If the execution
25099 of the user command results in a screen that has lines in common with the current screen, and
25100 the device has sufficient terminal capabilities, *more* shall scroll the screen; otherwise, it is
25101 unspecified whether the screen is scrolled or redrawn.

25102 For all files but the last (including standard input if no file was specified, and for the last file as
25103 well, if the *-e* option was not specified), when *more* has written the last line in the file, *more* shall
25104 prompt for a user command. This prompt shall contain the name of the next file as well as an
25105 indication that *more* has reached end-of-file. If the user command is *f*, <control>-F, <space>, *j*,
25106 <newline>, *d*, <control>-D, or *s*, *more* shall display the next file. Otherwise, if displaying the last
25107 file, *more* shall exit. Otherwise, *more* shall execute the user command specified.

25108 Several of the commands described in this section display a previous screen from the input
25109 stream. In the case that text is being taken from a non-rewindable stream, such as a pipe, it is
25110 implementation-defined how much backwards motion is supported. If a command cannot be
25111 executed because of a limitation on backwards motion, an error message to this effect shall be
25112 displayed, the current screen shall not change, and the user shall be prompted for another
25113 command.

25114 If a command cannot be performed because there are insufficient lines to display, *more* shall alert
25115 the terminal. If a command cannot be performed because there are insufficient lines to display or
25116 a / command fails: if the input is the standard input, the last screen in the file may be displayed;
25117 otherwise, the current file and screen shall not change, and the user shall be prompted for another
25118 command.

25119 The interactive commands in the following sections shall be supported. Some commands can be
25120 preceded by a decimal integer, called *count* in the following descriptions. If not specified with
25121 the command, *count* shall default to 1. In the following descriptions, *pattern* is a basic regular
25122 expression, as described in the Base Definitions volume of IEEE Std 1003.1-2001, Section 9.3,
25123 Basic Regular Expressions. The term "examine" is historical usage meaning "open the file for
25124 viewing"; for example, *more foo* would be expressed as examining file *foo*.

25125 In the following descriptions, unless otherwise specified, *line* is a line in the *more* display, not a
25126 line from the file being examined.

25127 In the following descriptions, the *current position* refers to two things:

- 25128 1. The position of the current line on the screen
25129 2. The line number (in the file) of the current line on the screen

25130 Usually, the line on the screen corresponding to the current position is the third line on the
25131 screen. If this is not possible (there are fewer than three lines to display or this is the first page of
25132 the file, or it is the last page of the file), then the current position is either the first or last line on
25133 the screen as described later.

25134 **Help**

25135 *Synopsis:* h

25136 Write a summary of these commands and other implementation-defined commands. The
25137 behavior shall be as if the *more* utility were executed with the -e option on a file that contained
25138 the summary information. The user shall be prompted as described earlier in this section when
25139 end-of-file is reached. If the user command is one of those specified to continue to the next file,
25140 *more* shall return to the file and screen state from which the h command was executed.

25141 **Scroll Forward One Screenful**

25142 *Synopsis:* [count]f
25143 [count]<control>-F

25144 Scroll forward *count* lines, with a default of one screenful. If *count* is more than the screen size,
25145 only the final screenful shall be written.

25146 **Scroll Backward One Screenful**

25147 *Synopsis:* [count]b
25148 [count]<control>-B

25149 Scroll backward *count* lines, with a default of one screenful (see the -n option). If *count* is more
25150 than the screen size, only the final screenful shall be written.

25151 **Scroll Forward One Line**

25152 *Synopsis:* [count]<space>
25153 [count]j
25154 [count]<newline>

25155 Scroll forward *count* lines. The default *count* for the <space> shall be one screenful; for j and
25156 <newline>, one line. The entire *count* lines shall be written, even if *count* is more than the screen
25157 size.

25158 **Scroll Backward One Line**

25159 *Synopsis:* [count]k

25160 Scroll backward *count* lines. The entire *count* lines shall be written, even if *count* is more than the
25161 screen size.

25162 **Scroll Forward One Half Screenful**

25163 *Synopsis:* [count]d

25164 [count]<control>-D

25165 Scroll forward *count* lines, with a default of one half of the screen size. If *count* is specified, it
25166 shall become the new default for subsequent **d**, <control>-D, and **u** commands.

25167 **Skip Forward One Line**

25168 *Synopsis:* [count]s

25169 Display the screenful beginning with the line *count* lines after the last line on the current screen.
25170 If *count* would cause the current position to be such that less than one screenful would be
25171 written, the last screenful in the file shall be written.

25172 **Scroll Backward One Half Screenful**

25173 *Synopsis:* [count]u

25174 [count]<control>-U

25175 Scroll backward *count* lines, with a default of one half of the screen size. If *count* is specified, it
25176 shall become the new default for subsequent **d**, <control>-D, **u**, and <control>-U commands.
25177 The entire *count* lines shall be written, even if *count* is more than the screen size.

25178 **Go to Beginning of File**

25179 *Synopsis:* [count]g

25180 Display the screenful beginning with line *count*.

25181 **Go to End-of-File**

25182 *Synopsis:* [count]G

25183 If *count* is specified, display the screenful beginning with the line *count*. Otherwise, display the
25184 last screenful of the file.

25185 **Refresh the Screen**

25186 *Synopsis:* r

25187 <control>-L

25188 Refresh the screen.

25189 **Discard and Refresh**

25190 *Synopsis:* R

25191 Refresh the screen, discarding any buffered input. If the current file is non-seekable, buffered
25192 input shall not be discarded and the **R** command shall be equivalent to the **r** command.

25193 **Mark Position**

25194 *Synopsis:* `mletter`

25195 Mark the current position with the letter named by *letter*, where *letter* represents the name of one
25196 of the lowercase letters of the portable character set. When a new file is examined, all marks may
25197 be lost.

25198 **Return to Mark**

25199 *Synopsis:* `' letter`

25200 Return to the position that was previously marked with the letter named by *letter*, making that
25201 line the current position.

25202 **Return to Previous Position**

25203 *Synopsis:* `''`

25204 Return to the position from which the last large movement command was executed (where a
25205 "large movement" is defined as any movement of more than a screenful of lines). If no such
25206 movements have been made, return to the beginning of the file.

25207 **Search Forward for Pattern**

25208 *Synopsis:* `[count]![pattern<newline>`

25209 Display the screenful beginning with the *count*th line containing the pattern. The search shall
25210 start after the first line currently displayed. The null regular expression ('/' followed by a
25211 <newline>) shall repeat the search using the previous regular expression, with a default *count*. If
25212 the character '!' is included, the matching lines shall be those that do not contain the *pattern*. If
25213 no match is found for the *pattern*, a message to that effect shall be displayed.

25214 **Search Backward for Pattern**

25215 *Synopsis:* `[count]?![pattern<newline>`

25216 Display the screenful beginning with the *count*th previous line containing the pattern. The search shall
25217 start on the last line before the first line currently displayed. The null regular
25218 expression ('?' followed by a <newline>) shall repeat the search using the previous regular
25219 expression, with a default *count*. If the character '!' is included, matching lines shall be those
25220 that do not contain the *pattern*. If no match is found for the *pattern*, a message to that effect shall
25221 be displayed.

25222 **Repeat Search**

25223 *Synopsis:* `[count]n`

25224 Repeat the previous search for *count*th line containing the last *pattern* (or not containing the last
25225 *pattern*, if the previous search was "/!" or "?!").

25226 **Repeat Search in Reverse**

25227 *Synopsis:* [*count*]N

25228 Repeat the search in the opposite direction of the previous search for the *count*th line containing
25229 the last *pattern* (or not containing the last *pattern*, if the previous search was "/!" or "?!").

25230 **Examine New File**

25231 *Synopsis:* :e [*filename*]<newline>

25232 Examine a new file. If the *filename* argument is not specified, the current file (see the :n and :p
25233 commands below) shall be re-examined. The *filename* shall be subjected to the process of shell
25234 word expansions (see Section 2.6 (on page 36)); if more than a single pathname results, the
25235 effects are unspecified. If *filename* is a number sign ('#'), the previously examined file shall be
25236 re-examined. If *filename* is not accessible for any reason (including that it is a non-seekable file),
25237 an error message to this effect shall be displayed and the current file and screen shall not change.

25238 **Examine Next File**

25239 *Synopsis:* [*count*]:n

25240 Examine the next file. If a number *count* is specified, the *count*th next file shall be examined. If
25241 *filename* refers to a non-seekable file, the results are unspecified.

25242 **Examine Previous File**

25243 *Synopsis:* [*count*]:p

25244 Examine the previous file. If a number *count* is specified, the *count*th previous file shall be
25245 examined. If *filename* refers to a non-seekable file, the results are unspecified.

25246 **Go to Tag**

25247 *Synopsis:* :t *tagstring*<newline>

25248 If the file containing the tag named by the *tagstring* argument is not the current file, examine the
25249 file, as if the :e command was executed with that file as the argument. Otherwise, or in addition,
25250 display the screenful beginning with the tag, as described for the -t option (see the OPTIONS
25251 section). If the ctags utility is not supported by the system, the use of :t produces undefined
25252 results.

25253 **Invoke Editor**

25254 *Synopsis:* v

25255 Invoke an editor to edit the current file being examined. If standard input is being examined, the
25256 results are unspecified. The name of the editor shall be taken from the environment variable
25257 EDITOR, or shall default to vi. If the last pathname component in EDITOR is either vi or ex, the
25258 editor shall be invoked with a -c *linenumber* command line argument, where *linenumber* is the
25259 line number of the file line containing the display line currently displayed as the first line of the
25260 screen. It is implementation-defined whether line-setting options are passed to editors other
25261 than vi and ex.

25262 When the editor exits, more shall resume with the same file and screen as when the editor was
25263 invoked.

25264 **Display Position**

25265 *Synopsis:* =
25266 <control>-G

25267 Write a message for which the information references the first byte of the line after the last line of
25268 the file on the screen. This message shall include the name of the file currently being examined,
25269 its number relative to the total number of files there are to examine, the line number in the file,
25270 the byte number and the total bytes in the file, and what percentage of the file precedes the
25271 current position. If *more* is reading from standard input, or the file is shorter than a single screen,
25272 the line number, the byte number, the total bytes, and the percentage need not be written.

25273 **Quit**

25274 *Synopsis:* q
25275 :q
25276 zz

25277 Exit *more*.

25278 **EXIT STATUS**

25279 The following exit values shall be returned:

25280 0 Successful completion.
25281 >0 An error occurred.

25282 **CONSEQUENCES OF ERRORS**

25283 If an error is encountered accessing a file when using the :n command, *more* shall attempt to
25284 examine the next file in the argument list, but the final exit status shall be affected. If an error is
25285 encountered accessing a file via the :p command, *more* shall attempt to examine the previous file
25286 in the argument list, but the final exit status shall be affected. If an error is encountered accessing
25287 a file via the :e command, *more* shall remain in the current file and the final exit status shall not
25288 be affected.

25289 **APPLICATION USAGE**

25290 When the standard output is not a terminal, only the -s filter-modification option is effective.
25291 This is based on historical practice. For example, a typical implementation of *man* pipes its
25292 output through *more* -s to squeeze excess white space for terminal users. When *man* is piped to
25293 *lp*, however, it is undesirable for this squeezing to happen.

25294 **EXAMPLES**

25295 The -p allows arbitrary commands to be executed at the start of each file. Examples are:

25296 *more -p G file1 file2*
25297 Examine each file starting with its last screenful.

25298 *more -p 100 file1 file2*
25299 Examine each file starting with line 100 in the current position (usually the third line, so line
25300 98 would be the first line written).

25301 *more -p /100 file1 file2*
25302 Examine each file starting with the first line containing the string "100" in the current
25303 position

25304 **RATIONALE**

25305 The *more* utility, available in BSD and BSD-derived systems, was chosen as the prototype for the
25306 POSIX file display program since it is more widely available than either the public-domain
25307 program *less* or than *pg*, a pager provided in System V. The 4.4 BSD *more* is the model for the

25308 features selected; it is almost fully upwards-compatible from the 4.3 BSD version in wide use
25309 and has become more amenable for *vi* users. Several features originally derived from various file
25310 editors, found in both *less* and *pg*, have been added to this volume of IEEE Std 1003.1-2001 as
25311 they have proved extremely popular with users.

25312 There are inconsistencies between *more* and *vi* that result from historical practice. For example,
25313 the single-character commands **h**, **f**, **b**, and <space> are screen movers in *more*, but cursor
25314 movers in *vi*. These inconsistencies were maintained because the cursor movements are not
25315 applicable to *more* and the powerful functionality achieved without the use of the control key
25316 justifies the differences.

25317 The tags interface has been included in a program that is not a text editor because it promotes
25318 another degree of consistent operation with *vi*. It is conceivable that the paging environment of
25319 *more* would be superior for browsing source code files in some circumstances.

25320 The operating mode referred to for block-mode terminals effectively adds a <newline> to each
25321 Synopsis line that currently has none. So, for example, **d<newline>** would page one screenful.
25322 The mode could be triggered by a command line option, environment variable, or some other
25323 method. The details are not imposed by this volume of IEEE Std 1003.1-2001 because there are so
25324 few systems known to support such terminals. Nevertheless, it was considered that all systems
25325 should be able to support *more* given the exception cited for this small community of terminals
25326 because, in comparison to *vi*, the cursor movements are few and the command set relatively
25327 amenable to the optional <newline>s.

25328 Some versions of *more* provide a shell escaping mechanism similar to the **ex !** command. The
25329 standard developers did not consider that this was necessary in a paginator, particularly given
25330 the wide acceptance of multiple window terminals and job control features. (They chose to
25331 retain such features in the editors and *mailx* because the shell interaction also gives an
25332 opportunity to modify the editing buffer, which is not applicable to *more*.)

25333 The **-p** (position) option replaces the **+** command because of the Utility Syntax Guidelines. In
25334 early proposals, it took a *pattern* argument, but historical *less* provided the *more* general facility of
25335 a command. It would have been desirable to use the same **-c** as *ex* and *vi*, but the letter was
25336 already in use.

25337 The text stating “from a non-rewindable stream … implementations may limit the amount of
25338 backwards motion supported” would allow an implementation that permitted no backwards
25339 motion beyond text already on the screen. It was not possible to require a minimum amount of
25340 backwards motion that would be effective for all conceivable device types. The implementation
25341 should allow the user to back up as far as possible, within device and reasonable memory
25342 allocation constraints.

25343 Historically, non-printable characters were displayed using the ARPA standard mappings,
25344 which are as follows:

- 25345 1. Printable characters are left alone.
- 25346 2. Control characters less than \177 are represented as followed by the character offset from
25347 the '@' character in the ASCII map; for example, \007 is represented as 'G'.
- 25348 3. \177 is represented as followed by '?'.

25349 The display of characters having their eighth bit set was less standard. Existing implementations
25350 use hex (0x00), octal (\000), and a meta-bit display. (The latter displayed characters with their
25351 eighth bit set as the two characters "M-", followed by the seven-bit display as described
25352 previously.) The latter probably has the best claim to historical practice because it was used with
25353 the **-v** option of 4 BSD and 4 BSD-derived versions of the *cat* utility since 1980.

25354 No specific display format is required by IEEE Std 1003.1-2001. Implementations are encouraged
25355 to conform to historic practice in the absence of any strong reason to diverge.

25356 FUTURE DIRECTIONS

25357 None.

25358 SEE ALSO

25359 Chapter 2 (on page 29), *ctags*, *ed*, *ex*, *vi*

25360 CHANGE HISTORY

25361 First released in Issue 4.

25362 Issue 5

25363 The FUTURE DIRECTIONS section is added.

25364 Issue 6

25365 This utility is marked as part of the User Portability Utilities option.

25366 The obsolescent SYNOPSIS is removed.

25367 The utility has been extensively reworked for alignment with the IEEE P1003.2b draft standard:

- Changes have been made as a result of IEEE PASC Interpretations 1003.2 #37 and #109.
- The *more* utility should be able to handle underlined and emboldened displays of characters that are wider than a single column position.

25371 NAME

25372 mv — move files

25373 SYNOPSIS

25374 mv [-fi] *source_file target_file*25375 mv [-fi] *source_file... target_file*

25376 DESCRIPTION

25377 In the first synopsis form, the *mv* utility shall move the file named by the *source_file* operand to the destination specified by the *target_file*. This first synopsis form is assumed when the final operand does not name an existing directory and is not a symbolic link referring to an existing directory.

25381 In the second synopsis form, *mv* shall move each file named by a *source_file* operand to a destination file in the existing directory named by the *target_dir* operand, or referenced if *target_dir* is a symbolic link referring to an existing directory. The destination path for each *source_file* shall be the concatenation of the target directory, a single slash character, and the last pathname component of the *source_file*. This second form is assumed when the final operand names an existing directory.

25387 If any operand specifies an existing file of a type not specified by the System Interfaces volume of IEEE Std 1003.1-2001, the behavior is implementation-defined.

25399 For each *source_file* the following steps shall be taken:

- 25390 1. If the destination path exists, the **-f** option is not specified, and either of the following conditions is true:

25392 a. The permissions of the destination path do not permit writing and the standard input
25393 is a terminal.

25394 b. The **-i** option is specified.

25395 the *mv* utility shall write a prompt to standard error and read a line from standard input. If
25396 the response is not affirmative, *mv* shall do nothing more with the current *source_file* and
25397 go on to any remaining *source_files*.

- 25398 2. The *mv* utility shall perform actions equivalent to the *rename()* function defined in the
25399 System Interfaces volume of IEEE Std 1003.1-2001, called with the following arguments:

25400 a. The *source_file* operand is used as the *old* argument.

25401 b. The destination path is used as the *new* argument.

25402 If this succeeds, *mv* shall do nothing more with the current *source_file* and go on to any
25403 remaining *source_files*. If this fails for any reasons other than those described for the *errno*
25404 [EXDEV] in the System Interfaces volume of IEEE Std 1003.1-2001, *mv* shall write a
25405 diagnostic message to standard error, do nothing more with the current *source_file*, and go
25406 on to any remaining *source_files*.

- 25407 3. If the destination path exists, and it is a file of type directory and *source_file* is not a file of
25408 type directory, or it is a file not of type directory and *source_file* is a file of type directory,
25409 *mv* shall write a diagnostic message to standard error, do nothing more with the current
25410 *source_file*, and go on to any remaining *source_files*.

- 25411 4. If the destination path exists, *mv* shall attempt to remove it. If this fails for any reason, *mv*
25412 shall write a diagnostic message to standard error, do nothing more with the current
25413 *source_file*, and go on to any remaining *source_files*.

25414 5. The file hierarchy rooted in *source_file* shall be duplicated as a file hierarchy rooted in the
25415 destination path. If *source_file* or any of the files below it in the hierarchy are symbolic
25416 links, the links themselves shall be duplicated, including their contents, rather than any
25417 files to which they refer. The following characteristics of each file in the file hierarchy shall
25418 be duplicated:

- 25419 • The time of last data modification and time of last access
25420 • The user ID and group ID
25421 • The file mode

25422 If the user ID, group ID, or file mode of a regular file cannot be duplicated, the file mode
25423 bits S_ISUID and S_ISGID shall not be duplicated.

25424 When files are duplicated to another file system, the implementation may require that the
25425 process invoking *mv* has read access to each file being duplicated.

25426 If the duplication of the file hierarchy fails for any reason, *mv* shall write a diagnostic
25427 message to standard error, do nothing more with the current *source_file*, and go on to any
25428 remaining *source_files*.

25429 If the duplication of the file characteristics fails for any reason, *mv* shall write a diagnostic
25430 message to standard error, but this failure shall not cause *mv* to modify its exit status.

25431 6. The file hierarchy rooted in *source_file* shall be removed. If this fails for any reason, *mv* shall
25432 write a diagnostic message to the standard error, do nothing more with the current
25433 *source_file*, and go on to any remaining *source_files*.

25434 OPTIONS

25435 The *mv* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section 12.2,
25436 Utility Syntax Guidelines.

25437 The following options shall be supported:

- 25438 **-f** Do not prompt for confirmation if the destination path exists. Any previous
25439 occurrence of the **-i** option is ignored.
25440 **-i** Prompt for confirmation if the destination path exists. Any previous occurrence of
25441 the **-f** option is ignored.

25442 Specifying more than one of the **-f** or **-i** options shall not be considered an error. The last option
25443 specified shall determine the behavior of *mv*.

25444 OPERANDS

25445 The following operands shall be supported:

- 25446 *source_file* A pathname of a file or directory to be moved.
25447 *target_file* A new pathname for the file or directory being moved.
25448 *target_dir* A pathname of an existing directory into which to move the input files.

25449 STDIN

25450 The standard input shall be used to read an input line in response to each prompt specified in
25451 the STDERR section. Otherwise, the standard input shall not be used.

25452 INPUT FILES

25453 The input files specified by each *source_file* operand can be of any file type.

25454 ENVIRONMENT VARIABLES

25455 The following environment variables shall affect the execution of *mv*:

25456 **LANG** Provide a default value for the internationalization variables that are unset or null.
25457 (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2,
25458 Internationalization Variables for the precedence of internationalization variables
25459 used to determine the values of locale categories.)

25460 **LC_ALL** If set to a non-empty string value, override the values of all the other
25461 internationalization variables.

25462 *LC_COLLATE*

25463 Determine the locale for the behavior of ranges, equivalence classes, and multi-
25464 character collating elements used in the extended regular expression defined for
25465 the **yesexpr** locale keyword in the *LC_MESSAGES* category.

25466 **LC_CTYPE** Determine the locale for the interpretation of sequences of bytes of text data as
25467 characters (for example, single-byte as opposed to multi-byte characters in
25468 arguments and input files), the behavior of character classes used in the extended
25469 regular expression defined for the **yesexpr** locale keyword in the *LC_MESSAGES*
25470 category.

25471 *LC_MESSAGES*

25472 Determine the locale for the processing of affirmative responses that should be
25473 used to affect the format and contents of diagnostic messages written to standard
25474 error.

25475 XSI **NLSPATH** Determine the location of message catalogs for the processing of *LC_MESSAGES*.

25476 ASYNCHRONOUS EVENTS

25477 Default.

25478 STDOUT

25479 Not used.

25480 STDERR

25481 Prompts shall be written to the standard error under the conditions specified in the
25482 DESCRIPTION section. The prompts shall contain the destination pathname, but their format is
25483 otherwise unspecified. Otherwise, the standard error shall be used only for diagnostic messages.

25484 OUTPUT FILES

25485 The output files may be of any file type.

25486 EXTENDED DESCRIPTION

25487 None.

25488 EXIT STATUS

25489 The following exit values shall be returned:

25490 0 All input files were moved successfully.

25491 >0 An error occurred.

25492 CONSEQUENCES OF ERRORS

25493 If the copying or removal of *source_file* is prematurely terminated by a signal or error, *mv* may
25494 leave a partial copy of *source_file* at the source or destination. The *mv* utility shall not modify
25495 both *source_file* and the destination path simultaneously; termination at any point shall leave
25496 either *source_file* or the destination path complete.

25497 APPLICATION USAGE

25498 Some implementations mark for update the *st_ctime* field of renamed files and some do not.
25499 Applications which make use of the *st_ctime* field may behave differently with respect to
25500 renamed files unless they are designed to allow for either behavior.

25501 EXAMPLES

25502 If the current directory contains only files **a** (of any type defined by the System Interfaces
25503 volume of IEEE Std 1003.1-2001), **b** (also of any type), and a directory **c**:

25504 mv a b c
25505 mv c d

25506 results with the original files **a** and **b** residing in the directory **d** in the current directory.

25507 RATIONALE

25508 Early proposals diverged from the SVID and BSD historical practice in that they required that
25509 when the destination path exists, the **-f** option is not specified, and input is not a terminal, *mv*
25510 fails. This was done for compatibility with *cp*. The current text returns to historical practice. It
25511 should be noted that this is consistent with the *rename()* function defined in the System
25512 Interfaces volume of IEEE Std 1003.1-2001, which does not require write permission on the
25513 target.

25514 For absolute clarity, paragraph (1), describing the behavior of *mv* when prompting for
25515 confirmation, should be interpreted in the following manner:

25516 if (exists AND (NOT f_option) AND
25517 ((not_writable AND input_is_terminal) OR i_option))

25518 The **-i** option exists on BSD systems, giving applications and users a way to avoid accidentally
25519 unlinking files when moving others. When the standard input is not a terminal, the 4.3 BSD *mv*
25520 deletes all existing destination paths without prompting, even when **-i** is specified; this is
25521 inconsistent with the behavior of the 4.3 BSD *cp* utility, which always generates an error when
25522 the file is unwritable and the standard input is not a terminal. The standard developers decided
25523 that use of **-i** is a request for interaction, so when the destination path exists, the utility takes
25524 instructions from whatever responds to standard input.

25525 The *rename()* function is able to move directories within the same file system. Some historical
25526 versions of *mv* have been able to move directories, but not to a different file system. The
25527 standard developers considered that this was an annoying inconsistency, so this volume of
25528 IEEE Std 1003.1-2001 requires directories to be able to be moved even across file systems. There
25529 is no **-R** option to confirm that moving a directory is actually intended, since such an option was
25530 not required for moving directories in historical practice. Requiring the application to specify it
25531 sometimes, depending on the destination, seemed just as inconsistent. The semantics of the
25532 *rename()* function were preserved as much as possible. For example, *mv* is not permitted to
25533 “rename” files to or from directories, even though they might be empty and removable.

25534 Historic implementations of *mv* did not exit with a non-zero exit status if they were unable to
25535 duplicate any file characteristics when moving a file across file systems, nor did they write a
25536 diagnostic message for the user. The former behavior has been preserved to prevent scripts from
25537 breaking; a diagnostic message is now required, however, so that users are alerted that the file
25538 characteristics have changed.

25539 The exact format of the interactive prompts is unspecified. Only the general nature of the
25540 contents of prompts are specified because implementations may desire more descriptive
25541 prompts than those used on historical implementations. Therefore, an application not using the
25542 **-f** option or using the **-i** option relies on the system to provide the most suitable dialog directly
25543 with the user, based on the behavior specified.

25544 When *mv* is dealing with a single file system and *source_file* is a symbolic link, the link itself is
25545 moved as a consequence of the dependence on the *rename()* functionality, per the
25546 DESCRIPTION. Across file systems, this has to be made explicit.

25547 **FUTURE DIRECTIONS**

25548 None.

25549 **SEE ALSO**

25550 *cp*, *In*, the System Interfaces volume of IEEE Std 1003.1-2001, *rename()*

25551 **CHANGE HISTORY**

25552 First released in Issue 2.

25553 **Issue 6**

25554 The *mv* utility is changed to describe processing of symbolic links as specified in the
25555 IEEE P1003.2b draft standard.

25556 The APPLICATION USAGE section is added.

25557 **NAME**

25558 newgrp — change to a new group

25559 **SYNOPSIS**

25560 UP newgrp [-l][group]

25561

25562 **DESCRIPTION**

25563 The *newgrp* utility shall create a new shell execution environment with a new real and effective
25564 group identification. Of the attributes listed in Section 2.12 (on page 61), the new shell execution
25565 environment shall retain the working directory, file creation mask, and exported variables from
25566 the previous environment (that is, open files, traps, unexported variables, alias definitions, shell
25567 functions, and *set* options may be lost). All other aspects of the process environment that are
25568 preserved by the *exec* family of functions defined in the System Interfaces volume of
25569 IEEE Std 1003.1-2001 shall also be preserved by *newgrp*; whether other aspects are preserved is
25570 unspecified.

25571 A failure to assign the new group identifications (for example, for security or password-related
25572 reasons) shall not prevent the new shell execution environment from being created.

25573 The *newgrp* utility shall affect the supplemental groups for the process as follows:

- 25574 • On systems where the effective group ID is normally in the supplementary group list (or
25575 whenever the old effective group ID actually is in the supplementary group list):

25576 — If the new effective group ID is also in the supplementary group list, *newgrp* shall change
25577 the effective group ID.

25578 — If the new effective group ID is not in the supplementary group list, *newgrp* shall add the
25579 new effective group ID to the list, if there is room to add it.

- 25580 • On systems where the effective group ID is not normally in the supplementary group list (or
25581 whenever the old effective group ID is not in the supplementary group list):

25582 — If the new effective group ID is in the supplementary group list, *newgrp* shall delete it.

25583 — If the old effective group ID is not in the supplementary list, *newgrp* shall add it if there is
25584 room.

25585 **Note:** The System Interfaces volume of IEEE Std 1003.1-2001 does not specify whether the effective
25586 group ID of a process is included in its supplementary group list.

25587 With no operands, *newgrp* shall change the effective group back to the groups identified in the
25588 user's user entry, and shall set the list of supplementary groups to that set in the user's group
25589 database entries.

25590 If a password is required for the specified group, and the user is not listed as a member of that
25591 group in the group database, the user shall be prompted to enter the correct password for that
25592 group. If the user is listed as a member of that group, no password shall be requested. If no
25593 password is required for the specified group, it is implementation-defined whether users not
25594 listed as members of that group can change to that group. Whether or not a password is
25595 required, implementation-defined system accounting or security mechanisms may impose
25596 additional authorization restrictions that may cause *newgrp* to write a diagnostic message and
25597 suppress the changing of the group identification.

25598 **OPTIONS**

25599 The *newgrp* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section
25600 12.2, Utility Syntax Guidelines.

25601	The following option shall be supported:	
25602	-l	(The letter ell.) Change the environment to what would be expected if the user actually logged in again.
25604 OPERANDS		
25605	The following operand shall be supported:	
25606	<i>group</i>	A group name from the group database or a non-negative numeric group ID.
25607		Specifies the group ID to which the real and effective group IDs shall be set. If <i>group</i> is a non-negative numeric string and exists in the group database as a group name (see <i>getgrnam()</i>), the numeric group ID associated with that group name shall be used as the group ID.
25611 STDIN		
25612	Not used.	
25613 INPUT FILES		
25614	The file /dev/tty shall be used to read a single line of text for password checking, when one is required.	
25615		
25616 ENVIRONMENT VARIABLES		
25617	The following environment variables shall affect the execution of <i>newgrp</i> :	
25618	<i>LANG</i>	Provide a default value for the internationalization variables that are unset or null. (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2, Internationalization Variables for the precedence of internationalization variables used to determine the values of locale categories.)
25619		
25620		
25621		
25622	<i>LC_ALL</i>	If set to a non-empty string value, override the values of all the other internationalization variables.
25623		
25624	<i>LC_CTYPE</i>	Determine the locale for the interpretation of sequences of bytes of text data as characters (for example, single-byte as opposed to multi-byte characters in arguments).
25625		
25626		
25627	<i>LC_MESSAGES</i>	Determine the locale that should be used to affect the format and contents of diagnostic messages written to standard error.
25628		
25629		
25630 XSI	<i>NLSPATH</i>	Determine the location of message catalogs for the processing of <i>LC_MESSAGES</i> .
25631 ASYNCHRONOUS EVENTS		
25632	Default.	
25633 STDOUT		
25634	Not used.	
25635 STDERR		
25636	The standard error shall be used for diagnostic messages and a prompt string for a password, if one is required. Diagnostic messages may be written in cases where the exit status is not available. See the EXIT STATUS section.	
25637		
25638		
25639 OUTPUT FILES		
25640	None.	
25641 EXTENDED DESCRIPTION		
25642	None.	

25643 EXIT STATUS

25644 If *newgrp* succeeds in creating a new shell execution environment, whether or not the group
25645 identification was changed successfully, the exit status shall be the exit status of the shell.
25646 Otherwise, the following exit value shall be returned:

25647 >0 An error occurred.

25648 CONSEQUENCES OF ERRORS

25649 The invoking shell may terminate.

25650 APPLICATION USAGE

25651 There is no convenient way to enter a password into the group database. Use of group
25652 passwords is not encouraged, because by their very nature they encourage poor security
25653 practices. Group passwords may disappear in the future.

25654 A common implementation of *newgrp* is that the current shell uses *exec* to overlay itself with
25655 *newgrp*, which in turn overlays itself with a new shell after changing group. On some
25656 implementations, however, this may not occur and *newgrp* may be invoked as a subprocess.

25657 The *newgrp* command is intended only for use from an interactive terminal. It does not offer a
25658 useful interface for the support of applications.

25659 The exit status of *newgrp* is generally inapplicable. If *newgrp* is used in a script, in most cases it
25660 successfully invokes a new shell and the rest of the original shell script is bypassed when the
25661 new shell exits. Used interactively, *newgrp* displays diagnostic messages to indicate problems.
25662 But usage such as:

```
25663 newgrp foo  
25664 echo $?
```

25665 is not useful because the new shell might not have access to any status *newgrp* may have
25666 generated (and most historical systems do not provide this status). A zero status echoed here
25667 does not necessarily indicate that the user has changed to the new group successfully. Following
25668 *newgrp* with the *id* command provides a portable means of determining whether the group
25669 change was successful or not.

25670 EXAMPLES

25671 None.

25672 RATIONALE

25673 Most historical implementations use one of the *exec* functions to implement the behavior of
25674 *newgrp*. Errors detected before the *exec* leave the environment unchanged, while errors detected
25675 after the *exec* leave the user in a changed environment. While it would be useful to have *newgrp*
25676 issue a diagnostic message to tell the user that the environment changed, it would be
25677 inappropriate to require this change to some historical implementations.

25678 The password mechanism is allowed in the group database, but how this would be
25679 implemented is not specified.

25680 The *newgrp* utility was retained in this volume of IEEE Std 1003.1-2001, even given the existence
25681 of the multiple group permissions feature in the System Interfaces volume of
25682 IEEE Std 1003.1-2001, for several reasons. First, in some implementations, the group ownership
25683 of a newly created file is determined by the group of the directory in which the file is created, as
25684 allowed by the System Interfaces volume of IEEE Std 1003.1-2001; on other implementations,
25685 the group ownership of a newly created file is determined by the effective group ID. On
25686 implementations of the latter type, *newgrp* allows files to be created with a specific group
25687 ownership. Finally, many implementations use the real group ID in accounting, and on such
25688 systems, *newgrp* allows the accounting identity of the user to be changed.

25689 FUTURE DIRECTIONS

25690 None.

25691 SEE ALSO

25692 Chapter 2 (on page 29), *sh*, the System Interfaces volume of IEEE Std 1003.1-2001, *exec*,
25693 *getgrnam()*

25694 CHANGE HISTORY

25695 First released in Issue 2.

25696 Issue 6

25697 This utility is marked as part of the User Portability Utilities option.

25698 The obsolescent SYNOPSIS is removed.

25699 The text describing supplemental groups is no longer conditional on {NGROUPS_MAX} being
25700 greater than 1. This is because {NGROUPS_MAX} now has a minimum value of 8. This is a FIPS
25701 requirement.

25702 **NAME**

25703 nice — invoke a utility with an altered nice value

25704 **SYNOPSIS**25705 UP nice [-n *increment*] *utility* [*argument...*]

25706

25707 **DESCRIPTION**

25708 The *nice* utility shall invoke a utility, requesting that it be run with a different nice value (see the
25709 Base Definitions volume of IEEE Std 1003.1-2001, Section 3.239, Nice Value). With no options
25710 and only if the user has appropriate privileges, the executed utility shall be run with a nice value
25711 that is some implementation-defined quantity less than or equal to the nice value of the current
25712 process. If the user lacks appropriate privileges to affect the nice value in the requested manner,
25713 the *nice* utility shall not affect the nice value; in this case, a warning message may be written to
25714 standard error, but this shall not prevent the invocation of *utility* or affect the exit status.

25715 **OPTIONS**

25716 The *nice* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section
25717 12.2, Utility Syntax Guidelines.

25718 The following option is supported:

25719 **-n *increment*** A positive or negative decimal integer which shall have the same effect on the
25720 execution of the utility as if the utility had called the *nice()* function with the
25721 numeric value of the *increment* option-argument.

25722 **OPERANDS**

25723 The following operands shall be supported:

25724 **utility** The name of a utility that is to be invoked. If the *utility* operand names any of the
25725 special built-in utilities in Section 2.14 (on page 64), the results are undefined.

25726 **argument** Any string to be supplied as an argument when invoking the utility named by the
25727 *utility* operand.

25728 **STDIN**

25729 Not used.

25730 **INPUT FILES**

25731 None.

25732 **ENVIRONMENT VARIABLES**

25733 The following environment variables shall affect the execution of *nice*:

25734 **LANG** Provide a default value for the internationalization variables that are unset or null.
25735 (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2,
25736 Internationalization Variables for the precedence of internationalization variables
25737 used to determine the values of locale categories.)

25738 **LC_ALL** If set to a non-empty string value, override the values of all the other
25739 internationalization variables.

25740 **LC_CTYPE** Determine the locale for the interpretation of sequences of bytes of text data as
25741 characters (for example, single-byte as opposed to multi-byte characters in
25742 arguments).

25743 **LC_MESSAGES**

25744 Determine the locale that should be used to affect the format and contents of
25745 diagnostic messages written to standard error.

25746 XSI	NLSPATH	Determine the location of message catalogs for the processing of <i>LC_MESSAGES</i> .
25747	PATH	Determine the search path used to locate the utility to be invoked. See the Base Definitions volume of IEEE Std 1003.1-2001, Chapter 8, Environment Variables.
25748		
25749 ASYNCHRONOUS EVENTS		
25750		Default.
25751 STDOUT		
25752		Not used.
25753 STDERR		
25754		The standard error shall be used only for diagnostic messages.
25755 OUTPUT FILES		
25756		None.
25757 EXTENDED DESCRIPTION		
25758		None.
25759 EXIT STATUS		
25760		If <i>utility</i> is invoked, the exit status of <i>nice</i> shall be the exit status of <i>utility</i> ; otherwise, the <i>nice</i> utility shall exit with one of the following values:
25761		
25762	1-125	An error occurred in the <i>nice</i> utility.
25763	126	The utility specified by <i>utility</i> was found but could not be invoked.
25764	127	The utility specified by <i>utility</i> could not be found.
25765 CONSEQUENCES OF ERRORS		
25766		Default.
25767 APPLICATION USAGE		
25768		The only guaranteed portable uses of this utility are:
25769	<i>nice utility</i>	
25770		Run <i>utility</i> with the default lower nice value.
25771	<i>nice -n <positive integer> utility</i>	
25772		Run <i>utility</i> with a lower nice value.
25773		On some implementations they have no discernible effect on the invoked utility and on some others they are exactly equivalent.
25774		
25775		Historical systems have frequently supported the <i><positive integer></i> up to 20. Since there is no
25776		error penalty associated with guessing a number that is too high, users without access to the
25777		system conformance document (to see what limits are actually in place) could use the historical
25778		1 to 20 range or attempt to use very large numbers if the job should be truly low priority.
25779		The nice value of a process can be displayed using the command:
25780		<code>ps -o nice</code>
25781		The <i>command</i> , <i>env</i> , <i>nice</i> , <i>nohup</i> , <i>time</i> , and <i>xargs</i> utilities have been specified to use exit code 127 if
25782		an error occurs so that applications can distinguish “failure to find a utility” from “invoked
25783		utility exited with an error indication”. The value 127 was chosen because it is not commonly
25784		used for other meanings; most utilities use small values for “normal error conditions” and the
25785		values above 128 can be confused with termination due to receipt of a signal. The value 126 was
25786		chosen in a similar manner to indicate that the utility could be found, but not invoked. Some
25787		scripts produce meaningful error messages differentiating the 126 and 127 cases. The distinction
25788		between exit codes 126 and 127 is based on KornShell practice that uses 127 when all attempts to

25789 exec the utility fail with [ENOENT], and uses 126 when any attempt to exec the utility fails for
25790 any other reason.

25791 **EXAMPLES**

25792 None.

25793 **RATIONALE**

25794 The 4.3 BSD version of *nice* does not check whether *increment* is a valid decimal integer. The 2
25795 command *nice -x utility*, for example, would be treated the same as the command *nice --1*
25796 *utility*. If the user does not have appropriate privileges, this results in a “permission denied”
25797 error. This is considered a bug.

25798 When a user without appropriate privileges gives a negative *increment*, System V treats it like
25799 the command *nice -0 utility*, while 4.3 BSD writes a “permission denied” message and does not
25800 run the utility. Neither was considered clearly superior, so the behavior was left unspecified.

25801 The C shell has a built-in version of *nice* that has a different interface from the one described in
25802 this volume of IEEE Std 1003.1-2001.

25803 The term “utility” is used, rather than “command”, to highlight the fact that shell compound
25804 commands, pipelines, and so on, cannot be used. Special built-ins also cannot be used.
25805 However, “utility” includes user application programs and shell scripts, not just utilities defined
25806 in this volume of IEEE Std 1003.1-2001.

25807 Historical implementations of *nice* provide a nice value range of 40 or 41 discrete steps, with the
25808 default nice value being the midpoint of that range. By default, they lower the nice value of the
25809 executed utility by 10.

25810 Some historical documentation states that the *increment* value must be within a fixed range. This
25811 is misleading; the valid *increment* values on any invocation are determined by the current
25812 process nice value, which is not always the default.

25813 The definition of nice value is not intended to suggest that all processes in a system have
25814 priorities that are comparable. Scheduling policy extensions such as the realtime priorities in the
25815 System Interfaces volume of IEEE Std 1003.1-2001 make the notion of a single underlying
25816 priority for all scheduling policies problematic. Some implementations may implement the *nice*-
25817 related features to affect all processes on the system, others to affect just the general time-
25818 sharing activities implied by this volume of IEEE Std 1003.1-2001, and others may have no effect
25819 at all. Because of the use of “implementation-defined” in *nice* and *renice*, a wide range of
25820 implementation strategies are possible.

25821 **FUTURE DIRECTIONS**

25822 None.

25823 **SEE ALSO**

25824 Chapter 2 (on page 29), *renice*, the System Interfaces volume of IEEE Std 1003.1-2001, *nice()*

25825 **CHANGE HISTORY**

25826 First released in Issue 4.

25827 **Issue 6**

25828 This utility is marked as part of the User Portability Utilities option.

25829 The obsolescent SYNOPSIS is removed.

25830 IEEE Std 1003.1-2001/Cor 2-2004, item XCU/TC2/D6/18 is applied, deleting a paragraph of 2
25831 RATIONALE that referred to text no longer in the standard. 2

25832 NAME

25833 nl — line numbering filter

25834 SYNOPSIS

```
25835 XSI    nl [-p][-b type][-d delim][-f type][-h type][-i incr][-l num][-n format]
25836          [-s sep][-v startnum][-w width][file]
```

25837

25838 DESCRIPTION

25839 The *nl* utility shall read lines from the named *file* or the standard input if no *file* is named and
 25840 shall reproduce the lines to standard output. Lines shall be numbered on the left. Additional
 25841 functionality may be provided in accordance with the command options in effect.

25842 The *nl* utility views the text it reads in terms of logical pages. Line numbering shall be reset at
 25843 the start of each logical page. A logical page consists of a header, a body, and a footer section.
 25844 Empty sections are valid. Different line numbering options are independently available for
 25845 header, body, and footer (for example, no numbering of header and footer lines while
 25846 numbering blank lines only in the body).

25847 The starts of logical page sections shall be signaled by input lines containing nothing but the
 25848 following delimiter characters:

25849
25850
25851
25852

Line	Start of
\:\:\:\:	Header
\:\:\:	Body
\:	Footer

25853 Unless otherwise specified, *nl* shall assume the text being read is in a single logical page body.

25854 OPTIONS

25855 The *nl* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section 12.2,
 25856 Utility Syntax Guidelines. Only one file can be named.

25857 The following options shall be supported:

25858 **-b type** Specify which logical page body lines shall be numbered. Recognized *types* and
 25859 their meaning are:

25860 **a** Number all lines.

25861 **t** Number only non-empty lines.

25862 **n** No line numbering.

25863 **pstring** Number only lines that contain the basic regular expression specified in
 25864 *string*.

25865 The default *type* for logical page body shall be **t** (text lines numbered).

25866 **-d delim** Specify the delimiter characters that indicate the start of a logical page section.
 25867 These can be changed from the default characters "\:" to two user-specified
 25868 characters. If only one character is entered, the second character shall remain the
 25869 default character ':'.

25870 **-f type** Specify the same as **b type** except for footer. The default for logical page footer
 25871 shall be **n** (no lines numbered).

25872 **-h type** Specify the same as **b type** except for header. The default *type* for logical page
 25873 header shall be **n** (no lines numbered).

25874	-i incr	Specify the increment value used to number logical page lines. The default shall be 1.
25876	-l num	Specify the number of blank lines to be considered as one. For example, -l 2 results in only the second adjacent blank line being numbered (if the appropriate -h a , -b a , or -f a option is set). The default shall be 1.
25879	-n format	Specify the line numbering format. Recognized values are: In , left justified, leading zeros suppressed; rn , right justified, leading zeros suppressed; rz , right justified, leading zeros kept. The default <i>format</i> shall be rn (right justified).
25882	-p	Specify that numbering should not be restarted at logical page delimiters.
25883	-s sep	Specify the characters used in separating the line number and the corresponding text line. The default <i>sep</i> shall be a <tab>.
25885	-v startnum	Specify the initial value used to number logical page lines. The default shall be 1.
25886	-w width	Specify the number of characters to be used for the line number. The default <i>width</i> shall be 6.

25888 OPERANDS

25889 The following operand shall be supported:

25890 *file* A pathname of a text file to be line-numbered.

25891 STDIN

25892 The standard input is a text file that is used if no *file* operand is given.

25893 INPUT FILES

25894 The input file named by the *file* operand is a text file.

25895 ENVIRONMENT VARIABLES

25896 The following environment variables shall affect the execution of *nl*:

25897	LANG	Provide a default value for the internationalization variables that are unset or null. (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2, Internationalization Variables for the precedence of internationalization variables used to determine the values of locale categories.)
25901	LC_ALL	If set to a non-empty string value, override the values of all the other internationalization variables.
25903	LC_COLLATE	Determine the locale for the behavior of ranges, equivalence classes, and multi-character collating elements within regular expressions.
25906	LC_CTYPE	Determine the locale for the interpretation of sequences of bytes of text data as characters (for example, single-byte as opposed to multi-byte characters in arguments and input files), the behavior of character classes within regular expressions, and for deciding which characters are in character class graph (for the -b t , -f t , and -h t options).
25911	LC_MESSAGES	Determine the locale that should be used to affect the format and contents of diagnostic messages written to standard error.
25914	NLSPATH	Determine the location of message catalogs for the processing of LC_MESSAGES .

25915 **ASYNCHRONOUS EVENTS**

25916 Default.

25917 **STDOUT**

25918 The standard output shall be a text file in the following format:

25919 "%s%s%s", <line number>, <separator>, <input line>

25920 where <line number> is one of the following numeric formats:

25921 %6d When the **rn** format is used (the default; see **-n**).25922 %06d When the **rz** format is used.25923 %-6d When the **ln** format is used.25924 <empty> When line numbers are suppressed for a portion of the page; the <separator> is also
25925 suppressed.25926 In the preceding list, the number 6 is the default width; the **-w** option can change this value.25927 **STDERR**

25928 The standard error shall be used only for diagnostic messages.

25929 **OUTPUT FILES**

25930 None.

25931 **EXTENDED DESCRIPTION**

25932 None.

25933 **EXIT STATUS**

25934 The following exit values shall be returned:

25935 0 Successful completion.

25936 >0 An error occurred.

25937 **CONSEQUENCES OF ERRORS**

25938 Default.

25939 **APPLICATION USAGE**25940 In using the **-d** *delim* option, care should be taken to escape characters that have special meaning
25941 to the command interpreter.25942 **EXAMPLES**

25943 The command:

25944 nl -v 10 -i 10 -d \!+ file1

25945 numbers *file1* starting at line number 10 with an increment of 10. The logical page delimiter is
25946 "**!+**". Note that the '**!**' has to be escaped when using *csh* as a command interpreter because of
25947 its history substitution syntax. For *ksh* and *sh* the escape is not necessary, but does not do any
25948 harm.25949 **RATIONALE**

25950 None.

25951 **FUTURE DIRECTIONS**

25952 None.

25953 **SEE ALSO**

25954 *pr*

25955 **CHANGE HISTORY**

25956 First released in Issue 2.

25957 **Issue 5**

25958 The option [-f *type*] is added to the SYNOPSIS. The option descriptions are presented in
25959 alphabetic order. The description of -bt is changed to “Number only non-empty lines”.

25960 **Issue 6**

25961 The obsolescent behavior allowing the options to be intermingled with the optional *file* operand
25962 is removed.

25963 NAME

25964 nm — write the name list of an object file (**DEVELOPMENT**)

25965 SYNOPSIS

25966 UP SD XSI nm [-APv][-efox][-g|-u][-t *format*] *file...*

25967

25968 DESCRIPTION

25969 This utility shall be provided on systems that support both the User Portability Utilities option
25970 and the Software Development Utilities option. On other systems it is optional. Certain options
25971 are only available on XSI-conformant systems.

25972 The *nm* utility shall display symbolic information appearing in the object file, executable file, or
25973 object-file library named by *file*. If no symbolic information is available for a valid input file, the
25974 *nm* utility shall report that fact, but not consider it an error condition.

25975 XSI The default base used when numeric values are written is unspecified. On XSI-conformant
25976 systems, it shall be decimal.

25977 OPTIONS

25978 The *nm* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section
25979 12.2, Utility Syntax Guidelines.

25980 The following options shall be supported:

25981 **-A** Write the full pathname or library name of an object on each line.

25982 XSI **-e** Write only external (global) and static symbol information.

25983 XSI **-f** Produce full output. Write redundant symbols (.text, .data, and .bss), normally
25984 suppressed.

25985 **-g** Write only external (global) symbol information.

25986 XSI **-o** Write numeric values in octal (equivalent to **-t o**).

25987 **-P** Write information in a portable output format, as specified in the STDOUT section.

25988 **-t *format*** Write each numeric value in the specified format. The format shall be dependent
25989 on the single character used as the *format* option-argument:

25990 XSI **d** The offset is written in decimal (default).

25991 **o** The offset is written in octal.

25992 **x** The offset is written in hexadecimal.

25993 **-u** Write only undefined symbols.

25994 **-v** Sort output by value instead of alphabetically.

25995 XSI **-x** Write numeric values in hexadecimal (equivalent to **-t x**).

25996 OPERANDS

25997 The following operand shall be supported:

25998 *file* A pathname of an object file, executable file, or object-file library.

25999 STDIN

26000 See the INPUT FILES section.

26001 INPUT FILES

26002 The input file shall be an object file, an object-file library whose format is the same as those produced by the *ar* utility for link editing, or an executable file. The *nm* utility may accept additional implementation-defined object library formats for the input file.

26005 ENVIRONMENT VARIABLES

26006 The following environment variables shall affect the execution of *nm*:

26007 *LANG* Provide a default value for the internationalization variables that are unset or null. (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2, Internationalization Variables for the precedence of internationalization variables used to determine the values of locale categories.)

26011 *LC_ALL* If set to a non-empty string value, override the values of all the other internationalization variables.

LC_COLLATE

26014 Determine the locale for character collation information for the symbol-name and symbol-value collation sequences.

26016 *LC_CTYPE* Determine the locale for the interpretation of sequences of bytes of text data as characters (for example, single-byte as opposed to multi-byte characters in arguments).

LC_MESSAGES

26020 Determine the locale that should be used to affect the format and contents of diagnostic messages written to standard error.

26022 XSI *NLSPATH* Determine the location of message catalogs for the processing of *LC_MESSAGES*.

26023 ASYNCHRONOUS EVENTS

26024 Default.

26025 STDOUT

26026 If symbolic information is present in the input files, then for each file or for each member of an archive, the *nm* utility shall write the following information to standard output. By default, the format is unspecified, but the output shall be sorted alphabetically by symbol name:

- Library or object name, if *-A* is specified

- Symbol name

- Symbol type, which shall either be one of the following single characters or an implementation-defined type represented by a single character:

- A Global absolute symbol.

- a Local absolute symbol.

- B Global “bss” (that is, uninitialized data space) symbol.

- b Local bss symbol.

- D Global data symbol.

- d Local data symbol.

- T Global text symbol.

- t Local text symbol.

- U Undefined symbol.

- 26042 • Value of the symbol
- 26043 • The size associated with the symbol, if applicable
- 26044 This information may be supplemented by additional information specific to the
26045 implementation.
- 26046 If the **-P** option is specified, the previous information shall be displayed using the following
26047 portable format. The three versions differ depending on whether **-t d**, **-t o**, or **-t x** was specified,
26048 respectively:
- 26049 "**%s%s %s %d\n**", <library/object name>, <name>, <type>,
26050 <value>, <size>
- 26051 "**%s%s %s %o\n**", <library/object name>, <name>, <type>,
26052 <value>, <size>
- 26053 "**%s%s %s %x %x\n**", <library/object name>, <name>, <type>,
26054 <value>, <size>
- 26055 where <library/object name> shall be formatted as follows:
- 26056 • If **-A** is not specified, <library/object name> shall be an empty string.
- 26057 • If **-A** is specified and the corresponding *file* operand does not name a library:
26058 "**%s:** ", <file>
- 26059 • If **-A** is specified and the corresponding *file* operand names a library. In this case, <object file>
26060 shall name the object file in the library containing the symbol being described:
26061 "**%s[%s]:** ", <file>, <object file>
- 26062 If **-A** is not specified, then if more than one *file* operand is specified or if only one *file* operand is
26063 specified and it names a library, *nm* shall write a line identifying the object containing the
26064 following symbols before the lines containing those symbols, in the form:
- 26065 • If the corresponding *file* operand does not name a library:
26066 "**%s:\n**", <file>
- 26067 • If the corresponding *file* operand names a library; in this case, <object file> shall be the name
26068 of the file in the library containing the following symbols:
26069 "**%s[%s]:\n**", <file>, <object file>
- 26070 If **-P** is specified, but **-t** is not, the format shall be as if **-t x** had been specified.
- 26071 **STDERR**
- 26072 The standard error shall be used only for diagnostic messages.
- 26073 **OUTPUT FILES**
- 26074 None.
- 26075 **EXTENDED DESCRIPTION**
- 26076 None.
- 26077 **EXIT STATUS**
- 26078 The following exit values shall be returned:
- 26079 0 Successful completion.
- 26080 >0 An error occurred.

26081 **CONSEQUENCES OF ERRORS**

26082 Default.

26083 **APPLICATION USAGE**

26084 Mechanisms for dynamic linking make this utility less meaningful when applied to an executable file because a dynamically linked executable may omit numerous library routines that would be found in a statically linked executable.

26087 **EXAMPLES**

26088 None.

26089 **RATIONALE**

26090 Historical implementations of *nm* have used different bases for numeric output and supplied different default types of symbols that were reported. The **-t** *format* option, similar to that used in *od* and *strings*, can be used to specify the numeric base; **-g** and **-u** can be used to restrict the amount of output or the types of symbols included in the output.

26094 The compromise of using **-t** *format* versus using **-d**, **-o**, and other similar options was necessary because of differences in the meaning of **-o** between implementations. The **-o** option from BSD has been provided here as **-A** to avoid confusion with the **-o** from System V (which has been provided here as **-t** and as **-o** on XSI-conformant systems).

26098 The option list was significantly reduced from that provided by historical implementations.

26099 The *nm* description is a subset of both the System V and BSD *nm* utilities with no specified default output.

26101 It was recognized that mechanisms for dynamic linking make this utility less meaningful when applied to an executable file (because a dynamically linked executable file may omit numerous library routines that would be found in a statically linked executable file), but the value of *nm* during software development was judged to outweigh other limitations.

26105 The default output format of *nm* is not specified because of differences in historical implementations. The **-P** option was added to allow some type of portable output format. After a comparison of the different formats used in SunOS, BSD, SVR3, and SVR4, it was decided to create one that did not match the current format of any of these four systems. The format devised is easy to parse by humans, easy to parse in shell scripts, and does not need to vary depending on locale (because no English descriptions are included). All of the systems currently have the information available to use this format.

26112 The format given in *nm* STDOUT uses spaces between the fields, which may be any number of <blank>s required to align the columns. The single-character types were selected to match historical practice, and the requirement that implementation additions also be single characters made parsing the information easier for shell scripts.

26116 **FUTURE DIRECTIONS**

26117 None.

26118 **SEE ALSO**26119 *ar*, *c99*26120 **CHANGE HISTORY**

26121 First released in Issue 2.

26122 **Issue 6**

26123 This utility is marked as supported when both the User Portability Utilities option and the Software Development Utilities option are supported.

26125 NAME

26126 nohup — invoke a utility immune to hangups

26127 SYNOPSIS

26128 nohup *utility* [*argument...*]

26129 DESCRIPTION

26130 The *nohup* utility shall invoke the utility named by the *utility* operand with arguments supplied
26131 as the *argument* operands. At the time the named *utility* is invoked, the SIGHUP signal shall be
26132 set to be ignored.

26133 If the standard output is a terminal, all output written by the named *utility* to its standard output
26134 shall be appended to the end of the file **nohup.out** in the current directory. If **nohup.out** cannot
26135 be created or opened for appending, the output shall be appended to the end of the file
26136 **nohup.out** in the directory specified by the *HOME* environment variable. If neither file can be
26137 created or opened for appending, *utility* shall not be invoked. If a file is created, the file's
26138 permission bits shall be set to S_IRUSR | S_IWUSR.

26139 If the standard error is a terminal, all output written by the named *utility* to its standard error
26140 shall be redirected to the same file descriptor as the standard output.

26141 OPTIONS

26142 None.

26143 OPERANDS

26144 The following operands shall be supported:

26145 *utility* The name of a utility that is to be invoked. If the *utility* operand names any of the
26146 special built-in utilities in Section 2.14 (on page 64), the results are undefined.

26147 *argument* Any string to be supplied as an argument when invoking the utility named by the
26148 *utility* operand.

26149 STDIN

26150 Not used.

26151 INPUT FILES

26152 None.

26153 ENVIRONMENT VARIABLES

26154 The following environment variables shall affect the execution of *nohup*:

26155 *HOME* Determine the pathname of the user's home directory: if the output file **nohup.out**
26156 cannot be created in the current directory, the *nohup* utility shall use the directory
26157 named by *HOME* to create the file.

26158 *LANG* Provide a default value for the internationalization variables that are unset or null.
26159 (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2,
26160 Internationalization Variables for the precedence of internationalization variables
26161 used to determine the values of locale categories.)

26162 *LC_ALL* If set to a non-empty string value, override the values of all the other
26163 internationalization variables.

26164 *LC_CTYPE* Determine the locale for the interpretation of sequences of bytes of text data as
26165 characters (for example, single-byte as opposed to multi-byte characters in
26166 arguments).

26167 *LC_MESSAGES*

26168 Determine the locale that should be used to affect the format and contents of

26169		diagnostic messages written to standard error.
26170	XSI	NLSPATH Determine the location of message catalogs for the processing of <i>LC_MESSAGES</i> .
26171		PATH Determine the search path that is used to locate the utility to be invoked. See the Base Definitions volume of IEEE Std 1003.1-2001, Chapter 8, Environment Variables.
26172		
26173		
26174		ASYNCHRONOUS EVENTS
26175		The <i>nohup</i> utility shall take the standard action for all signals except that SIGHUP shall be ignored.
26176		
26177		STDOUT
26178		If the standard output is not a terminal, the standard output of <i>nohup</i> shall be the standard output generated by the execution of the <i>utility</i> specified by the operands. Otherwise, nothing shall be written to the standard output.
26179		
26180		
26181		STDERR
26182		If the standard output is a terminal, a message shall be written to the standard error, indicating the name of the file to which the output is being appended. The name of the file shall be either nohup.out or \$HOME/nohup.out .
26183		
26184		
26185		OUTPUT FILES
26186		If the standard output is a terminal, all output written by the named <i>utility</i> to the standard output and standard error is appended to the file nohup.out , which is created if it does not already exist.
26187		
26188		
26189		EXTENDED DESCRIPTION
26190		None.
26191		EXIT STATUS
26192		The following exit values shall be returned:
26193	126	The <i>utility</i> specified by <i>utility</i> was found but could not be invoked.
26194	127	An error occurred in the <i>nohup</i> utility or the <i>utility</i> specified by <i>utility</i> could not be found.
26195		
26196		Otherwise, the exit status of <i>nohup</i> shall be that of the <i>utility</i> specified by the <i>utility</i> operand.
26197		CONSEQUENCES OF ERRORS
26198		Default.
26199		APPLICATION USAGE
26200		The <i>command</i> , <i>env</i> , <i>nice</i> , <i>nohup</i> , <i>time</i> , and <i>xargs</i> utilities have been specified to use exit code 127 if an error occurs so that applications can distinguish “failure to find a utility” from “invoked utility exited with an error indication”. The value 127 was chosen because it is not commonly used for other meanings; most utilities use small values for “normal error conditions” and the values above 128 can be confused with termination due to receipt of a signal. The value 126 was chosen in a similar manner to indicate that the utility could be found, but not invoked. Some scripts produce meaningful error messages differentiating the 126 and 127 cases. The distinction between exit codes 126 and 127 is based on KornShell practice that uses 127 when all attempts to <i>exec</i> the utility fail with [ENOENT], and uses 126 when any attempt to <i>exec</i> the utility fails for any other reason.
26201		
26202		
26203		
26204		
26205		
26206		
26207		
26208		
26209		
26210		EXAMPLES
26211		It is frequently desirable to apply <i>nohup</i> to pipelines or lists of commands. This can be done by placing pipelines and command lists in a single file; this file can then be invoked as a utility, and the <i>nohup</i> applies to everything in the file.
26212		
26213		

26214 Alternatively, the following command can be used to apply *nohup* to a complex command:

26215 nohup sh -c '*complex-command-line*'

26216 **RATIONALE**

26217 The 4.3 BSD version ignores SIGTERM and SIGHUP, and if *./nohup.out* cannot be used, it fails
26218 instead of trying to use \$HOME/nohup.out.

26219 The *csh* utility has a built-in version of *nohup* that acts differently from the *nohup* defined in this
26220 volume of IEEE Std 1003.1-2001.

26221 The term *utility* is used, rather than *command*, to highlight the fact that shell compound
26222 commands, pipelines, special built-ins, and so on, cannot be used directly. However, *utility*
26223 includes user application programs and shell scripts, not just the standard utilities.

26224 Historical versions of the *nohup* utility use default file creation semantics. Some more recent
26225 versions use the permissions specified here as an added security precaution.

26226 Some historical implementations ignore SIGQUIT in addition to SIGHUP; others ignore
26227 SIGTERM. An early proposal allowed, but did not require, SIGQUIT to be ignored. Several
26228 reviewers objected that *nohup* should only modify the handling of SIGHUP as required by this
26229 volume of IEEE Std 1003.1-2001.

26230 **FUTURE DIRECTIONS**

26231 None.

26232 **SEE ALSO**

26233 Chapter 2 (on page 29), *sh*, the System Interfaces volume of IEEE Std 1003.1-2001, *signal()*

26234 **CHANGE HISTORY**

26235 First released in Issue 2.

26236 NAME

26237 od — dump files in various formats

26238 SYNOPSIS

26239 od [-v][-A address_base][-j skip][-N count][-t type_string]...
26240 [file...]

26241 XSI od [-bcdosx][file] [+][offset[.][b]]

26242

26243 DESCRIPTION

26244 The *od* utility shall write the contents of its input files to standard output in a user-specified
26245 format.

26246 OPTIONS

26247 The *od* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section 12.2,
26248 XSI Utility Syntax Guidelines, except that the order of presentation of the **-t** options and the
26249 **-bcdosx** options is significant.

26250 The following options shall be supported:

26251 **-A address_base**26252 Specify the input offset base. See the EXTENDED DESCRIPTION section. The
26253 application shall ensure that the *address_base* option-argument is a character. The
26254 characters 'd', 'o', and 'x' specify that the offset base shall be written in
26255 decimal, octal, or hexadecimal, respectively. The character 'n' specifies that the
26256 offset shall not be written.26257 XSI **-b** Interpret bytes in octal. This shall be equivalent to **-t o1**.26258 XSI **-c** Interpret bytes as characters specified by the current setting of the *LC_CTYPE*
26259 category. Certain non-graphic characters appear as C escapes: "NUL=\0",
26260 "BS=\b", "FF=\f", "NL=\n", "CR=\r", "HT=\t"; others appear as 3-digit octal
26261 numbers.26262 XSI **-d** Interpret words (two-byte units) in unsigned decimal. This shall be equivalent to
26263 **-t u2**.26264 **-j skip** Jump over *skip* bytes from the beginning of the input. The *od* utility shall read or
26265 seek past the first *skip* bytes in the concatenated input files. If the combined input
26266 is not at least *skip* bytes long, the *od* utility shall write a diagnostic message to
26267 standard error and exit with a non-zero exit status.26268 By default, the *skip* option-argument shall be interpreted as a decimal number.
26269 With a leading 0x or 0X, the offset shall be interpreted as a hexadecimal number;
26270 otherwise, with a leading '0', the offset shall be interpreted as an octal number.
26271 Appending the character 'b', 'k', or 'm' to offset shall cause it to be interpreted
26272 as a multiple of 512, 1 024, or 1 048 576 bytes, respectively. If the *skip* number is
26273 hexadecimal, any appended 'b' shall be considered to be the final hexadecimal
26274 digit.26275 **-N count** Format no more than *count* bytes of input. By default, *count* shall be interpreted as
26276 a decimal number. With a leading 0x or 0X, *count* shall be interpreted as a
26277 hexadecimal number; otherwise, with a leading '0', it shall be interpreted as an
26278 octal number. If *count* bytes of input (after successfully skipping, if **-j skip** is
26279 specified) are not available, it shall not be considered an error; the *od* utility shall
26280 format the input that is available.

26281 XSI	-o	Interpret <i>words</i> (two-byte units) in octal. This shall be equivalent to -t o2 .
26282 XSI	-s	Interpret <i>words</i> (two-byte units) in signed decimal. This shall be equivalent to -t d2 .
26284	-t type_string	Specify one or more output types. See the EXTENDED DESCRIPTION section. The application shall ensure that the <i>type_string</i> option-argument is a string specifying the types to be used when writing the input data. The string shall consist of the type specification characters a, c, d, f, o, u, and x, specifying named character, character, signed decimal, floating point, octal, unsigned decimal, and hexadecimal, respectively. The type specification characters d, f, o, u, and x can be followed by an optional unsigned decimal integer that specifies the number of bytes to be transformed by each instance of the output type. The type specification character f can be followed by an optional F, D, or L indicating that the conversion should be applied to an item of type float , double , or long double , respectively. The type specification characters d, o, u, and x can be followed by an optional C, S, I, or L indicating that the conversion should be applied to an item of type char , short , int , or long , respectively. Multiple types can be concatenated within the same <i>type_string</i> and multiple -t options can be specified. Output lines shall be written for each type specified in the order in which the type specification characters are specified.
26301	-v	Write all input data. Without the -v option, any number of groups of output lines, which would be identical to the immediately preceding group of output lines (except for the byte offsets), shall be replaced with a line containing only an asterisk ('*').
26305 XSI	-x	Interpret <i>words</i> (two-byte units) in hexadecimal. This shall be equivalent to -t x2 .
26306 XSI		Multiple types can be specified by using multiple -bcdostx options. Output lines are written for each type specified in the order in which the types are specified.
26308	OPERANDS	
26309		The following operands shall be supported:
26310	<i>file</i>	A pathname of a file to be read. If no <i>file</i> operands are specified, the standard input shall be used.
26312		If there are no more than two operands, none of the -A , -j , -N , or -t options is specified, and either of the following is true: the first character of the last operand is a plus sign ('+'), or there are two operands and the first character of the last operand is numeric; the last operand shall be interpreted as an offset operand on XSI-conformant systems. Under these conditions, the results are unspecified on systems that are not XSI-conformant systems.
26318 XSI	[+]offset[.][]b	The <i>offset</i> operand specifies the offset in the file where dumping is to commence. This operand is normally interpreted as octal bytes. If '.' is appended, the offset shall be interpreted in decimal. If 'b' is appended, the offset shall be interpreted in units of 512 bytes.
26322	STDIN	
26323		The standard input shall be used only if no <i>file</i> operands are specified. See the INPUT FILES section.
26324		

26325 INPUT FILES

26326 The input files can be any file type.

26327 ENVIRONMENT VARIABLES

26328 The following environment variables shall affect the execution of *od*:

26329 **LANG** Provide a default value for the internationalization variables that are unset or null.
26330 (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2,
26331 Internationalization Variables for the precedence of internationalization variables
26332 used to determine the values of locale categories.)

26333 **LC_ALL** If set to a non-empty string value, override the values of all the other
26334 internationalization variables.

26335 **LC_CTYPE** Determine the locale for the interpretation of sequences of bytes of text data as
26336 characters (for example, single-byte as opposed to multi-byte characters in
26337 arguments and input files).

26338 LC_MESSAGES

26339 Determine the locale that should be used to affect the format and contents of
26340 diagnostic messages written to standard error.

26341 LC_NUMERIC

26342 Determine the locale for selecting the radix character used when writing floating-
26343 point formatted output.

26344 XSI **NLSPATH** Determine the location of message catalogs for the processing of *LC_MESSAGES*.

26345 ASYNCHRONOUS EVENTS

26346 Default.

26347 STDOUT

26348 See the EXTENDED DESCRIPTION section.

26349 STDERR

26350 The standard error shall be used only for diagnostic messages.

26351 OUTPUT FILES

26352 None.

26353 EXTENDED DESCRIPTION

26354 The *od* utility shall copy sequentially each input file to standard output, transforming the input
26355 XSI data according to the output types specified by the **-t** option or the **-bcdosx** options. If no
26356 output type is specified, the default output shall be as if **-t os** had been specified.

26357 The number of bytes transformed by the output type specifier **c** may be variable depending on
26358 the *LC_CTYPE* category.

26359 The default number of bytes transformed by output type specifiers **d**, **f**, **o**, **u**, and **x** corresponds
26360 to the various C-language types as follows. If the **c99** compiler is present on the system, these
26361 specifiers shall correspond to the sizes used by default in that compiler. Otherwise, these sizes
26362 may vary among systems that conform to IEEE Std 1003.1-2001.

- 26363 • For the type specifier characters **d**, **o**, **u**, and **x**, the default number of bytes shall correspond
26364 to the size of the underlying implementation's basic integer type. For these specifier
26365 characters, the implementation shall support values of the optional number of bytes to be
26366 converted corresponding to the number of bytes in the C-language types **char**, **short**, **int**, and
26367 **long**. These numbers can also be specified by an application as the characters '**C**', '**S**', '**I**',
26368 and '**L**', respectively. The implementation shall also support the values 1, 2, 4, and 8, even if
26369 it provides no C-Language types of those sizes. The implementation shall support the

26370 decimal value corresponding to the C-language type **long long**. The byte order used when
 26371 interpreting numeric values is implementation-defined, but shall correspond to the order in
 26372 which a constant of the corresponding type is stored in memory on the system.

- For the type specifier character **f**, the default number of bytes shall correspond to the number of bytes in the underlying implementation's basic double precision floating-point data type. The implementation shall support values of the optional number of bytes to be converted corresponding to the number of bytes in the C-language types **float**, **double**, and **long double**. These numbers can also be specified by an application as the characters '**F**', '**D**', and '**L**', respectively.

26379 The type specifier character **a** specifies that bytes shall be interpreted as named characters from
 26380 the International Reference Version (IRV) of the ISO/IEC 646:1991 standard. Only the least
 26381 significant seven bits of each byte shall be used for this type specification. Bytes with the values
 26382 listed in the following table shall be written using the corresponding names for those characters.

26383 **Table 4-12** Named Characters in *od*

Value	Name	Value	Name	Value	Name	Value	Name
\000	nul	\001	soh	\002	stx	\003	etx
\004	eot	\005	enq	\006	ack	\007	bel
\010	bs	\011	ht	\012	If or nl [*]	\013	vt
\014	ff	\015	cr	\016	so	\017	si
\020	dle	\021	dcl	\022	dc2	\023	de3
\024	dc4	\025	nak	\026	syn	\027	etb
\030	can	\031	em	\032	sub	\033	esc
\034	fs	\035	gs	\036	rs	\037	us
\040	sp	\177	del				

26395 **Note:** The "\012" value may be written either as **If** or **nl**.

26396 The type specifier character **c** specifies that bytes shall be interpreted as characters specified by
 26397 the current setting of the *LC_CTYPE* locale category. Characters listed in the table in the Base
 26398 Definitions volume of IEEE Std 1003.1-2001, Chapter 5, File Format Notation ('\\', '\a', '\b',
 26399 '\f', '\n', '\r', '\t', '\v') shall be written as the corresponding escape sequences, except
 26400 that backslash shall be written as a single backslash and a NUL shall be written as '\0'. Other
 26401 non-printable characters shall be written as one three-digit octal number for each byte in the
 26402 character. Printable multi-byte characters shall be written in the area corresponding to the first
 26403 byte of the character; the two-character sequence "##" shall be written in the area
 26404 corresponding to each remaining byte in the character, as an indication that the character is
 26405 continued. When either the **-j skip** or **-N count** option is specified along with the **c** type specifier,
 26406 and this results in an attempt to start or finish in the middle of a multi-byte character, the result
 26407 is implementation-defined.

2

26408 The input data shall be manipulated in blocks, where a block is defined as a multiple of the least
 26409 common multiple of the number of bytes transformed by the specified output types. If the least
 26410 common multiple is greater than 16, the results are unspecified. Each input block shall be
 26411 written as transformed by each output type, one per written line, in the order that the output
 26412 types were specified. If the input block size is larger than the number of bytes transformed by
 26413 the output type, the output type shall sequentially transform the parts of the input block, and
 26414 the output from each of the transformations shall be separated by one or more <blank>s.

26415 If, as a result of the specification of the **-N** option or end-of-file being reached on the last input
 26416 file, input data only partially satisfies an output type, the input shall be extended sufficiently

26417 with null bytes to write the last byte of the input.

26418 Unless **-A n** is specified, the first output line produced for each input block shall be preceded by
26419 the input offset, cumulative across input files, of the next byte to be written. The format of the
26420 input offset is unspecified; however, it shall not contain any <blank>s, shall start at the first
26421 character of the output line, and shall be followed by one or more <blank>s. In addition, the
26422 offset of the byte following the last byte written shall be written after all the input data has been
26423 processed, but shall not be followed by any <blank>s.

26424 If no **-A** option is specified, the input offset base is unspecified.

26425 EXIT STATUS

26426 The following exit values shall be returned:

26427 0 All input files were processed successfully.

26428 >0 An error occurred.

26429 CONSEQUENCES OF ERRORS

26430 Default.

26431 APPLICATION USAGE

26432 XSI-conformant applications are warned not to use filenames starting with '+' or a first
26433 operand starting with a numeric character so that the old functionality can be maintained by
26434 implementations, unless they specify one of the **-A**, **-j**, or **-N** options. To guarantee that one of
26435 these filenames is always interpreted as a filename, an application could always specify the
26436 address base format with the **-A** option.

26437 EXAMPLES

26438 If a file containing 128 bytes with decimal values zero to 127, in increasing order, is supplied as
26439 standard input to the command:

26440 `od -A d -t a`

26441 on an implementation using an input block size of 16 bytes, the standard output, independent of
26442 the current locale setting, would be similar to:

26443	0000000 nul	soh	stx	etx	eot	enq	ack	bel	bs	ht	nl	vt	ff	cr	so	si
26444	0000016 dle	dc1	dc2	dc3	dc4	nak	syn	etb	can	em	sub	esc	fs	gs	rs	us
26445	0000032 sp	!	"	#	\$	%	&	'	()	*	+	,	-	.	/	
26446	0000048 0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
26447	0000064 @	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
26448	0000080 P	Q	R	S	T	U	V	W	X	Y	Z	[\]	^	_		
26449	0000096 `	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
26450	0000112 p	q	r	s	t	u	v	w	x	y	z	{ }	~	del		
26451	0000128															

26452 Note that this volume of IEEE Std 1003.1-2001 allows **nl** or **If** to be used as the name for the
26453 ISO/IEC 646:1991 standard IRV character with decimal value 10. The IRV names this character
26454 **If** (line feed), but traditional implementations have referred to this character as newline (**nl**) and
26455 the POSIX locale character set symbolic name for the corresponding character is a <newline>.

26456 The command:

26457 `od -A o -t o2x2x -N 18`

1

26458 on a system with 32-bit words and an implementation using an input block size of 16 bytes
26459 could write 18 bytes in approximately the following format:

```

26460      0000000 032056 031440 041123 042040 052516 044530 020043 031464
26461          342e   3320    4253    4420    554e    4958    2023    3334
26462          342e3320        42534420        554e4958        20233334
26463      0000020 032472
26464          353a
26465          353a0000
26466      0000022
26467      The command:
26468      od -A d -t f -t o4 -t x4 -N 24 -j 0x15
26469      on a system with 64-bit doubles (for example, IEEE Std 754-1985 double precision floating-point
26470      format) would skip 21 bytes of input data and then write 24 bytes in approximately the
26471      following format:
26472      0000000 1.00000000000000e+00 1.57350000000000e+01
26473          07774000000 000000000000 10013674121 35341217270
26474          3ff00000 00000000 402f3851 eb851eb8
26475      0000016 1.40668230000000e+02
26476          10030312542 04370303230
26477          40619562 23e18698
26478      0000024
26479  RATIONALE
26480      The od utility went through several names in early proposals, including hd, xd, and most recently
26481      hexdump. There were several objections to all of these based on the following reasons:
26482      • The hd and xd names conflicted with historical utilities that behaved differently.
26483      • The hexdump description was much more complex than needed for a simple dump utility.
26484      • The od utility has been available on all historical implementations and there was no need to
26485      create a new name for a utility so similar to the historical od utility.
26486      The original reasons for not standardizing historical od were also fairly widespread. Those
26487      reasons are given below along with rationale explaining why the standard developers believe
26488      that this version does not suffer from the indicated problem:
26489      • The BSD and System V versions of od have diverged, and the intersection of features
26490      provided by both does not meet the needs of the user community. In fact, the System V
26491      version only provides a mechanism for dumping octal bytes and shorts, signed and unsigned
26492      decimal shorts, hexadecimal shorts, and ASCII characters. BSD added the ability to dump
26493      floats, doubles, named ASCII characters, and octal, signed decimal, unsigned decimal, and
26494      hexadecimal longs. The version presented here provides more normalized forms for
26495      dumping bytes, shorts, ints, and longs in octal, signed decimal, unsigned decimal, and
26496      hexadecimal; float, double, and long double; and named ASCII as well as current locale
26497      characters.
26498      • It would not be possible to come up with a compatible superset of the BSD and System V
26499      flags that met the requirements of the standard developers. The historical default od output is
26500      the specified default output of this utility. None of the option letters chosen for this version
26501      of od conflict with any of the options to historical versions of od.
26502      • On systems with different sizes for short, int, and long, there was no way to ask for dumps
26503      of ints, even in the BSD version. Because of the way options are named, the name space
26504      could not be extended to solve these problems. This is why the -t option was added (with
26505      type specifiers more closely matched to the printf() formats used in the rest of this volume of

```

IEEE Std 1003.1-2001) and the optional field sizes were added to the d, f, o, u, and x type specifiers. It is also one of the reasons why the historical practice was not mandated as a required obsolescent form of *od*. (Although the old versions of *od* are not listed as an obsolescent form, implementations are urged to continue to recognize the older forms for several more years.) The a, c, f, o, and x types match the meaning of the corresponding format characters in the historical implementations of *od* except for the default sizes of the fields converted. The d format is signed in this volume of IEEE Std 1003.1-2001 to match the *printf()* notation. (Historical versions of *od* used d as a synonym for u in this version. The System V implementation uses s for signed decimal; BSD uses i for signed decimal and s for null-terminated strings.) Other than d and u, all of the type specifiers match format characters in the historical BSD version of **od**.

The sizes of the C-language types **char**, **short**, **int**, **long**, **float**, **double**, and **long double** are used even though it is recognized that there may be zero or more than one compiler for the C language on an implementation and that they may use different sizes for some of these types. (For example, one compiler might use 2 bytes **shorts**, 2 bytes **ints**, and 4 bytes **longs**, while another compiler (or an option to the same compiler) uses 2 bytes **shorts**, 4 bytes **ints**, and 4 bytes **longs**.) Nonetheless, there has to be a basic size known by the implementation for these types, corresponding to the values reported by invocations of the *getconf* utility when called with *system_var* operands {UCHAR_MAX}, {USHORT_MAX}, {UINT_MAX}, and {ULONG_MAX} for the types **char**, **short**, **int**, and **long**, respectively. There are similar constants required by the ISO C standard, but not required by the System Interfaces volume of IEEE Std 1003.1-2001 or this volume of IEEE Std 1003.1-2001. They are {FLT_MANT_DIG}, {DBL_MANT_DIG}, and {LDBL_MANT_DIG} for the types **float**, **double**, and **long double**, respectively. If the optional *c99* utility is provided by the implementation and used as specified by this volume of IEEE Std 1003.1-2001, these are the sizes that would be provided. If an option is used that specifies different sizes for these types, there is no guarantee that the *od* utility is able to interpret binary data output by such a program correctly.

This volume of IEEE Std 1003.1-2001 requires that the numeric values of these lengths be recognized by the *od* utility and that symbolic forms also be recognized. Thus, a conforming application can always look at an array of **unsigned long** data elements using *od -t uL*.

- The method of specifying the format for the address field based on specifying a starting offset in a file unnecessarily tied the two together. The **-A** option now specifies the address base and the **-S** option specifies a starting offset.
- It would be difficult to break the dependence on U.S. ASCII to achieve an internationalized utility. It does not seem to be any harder for *od* to dump characters in the current locale than it is for the *ed* or *sed l* commands. The c type specifier does this without difficulty and is completely compatible with the historical implementations of the c format character when the current locale uses a superset of the ISO/IEC 646:1991 standard as a codeset. The a type specifier (from the BSD a format character) was left as a portable means to dump ASCII (or more correctly ISO/IEC 646:1991 standard (IRV)) so that headers produced by *pax* could be deciphered even on systems that do not use the ISO/IEC 646:1991 standard as a subset of their base codeset.

The use of " ** " as an indication of continuation of a multi-byte character in c specifier output was chosen based on seeing an implementation that uses this method. The continuation bytes have to be marked in a way that is not ambiguous with another single-byte or multi-byte character.

An early proposal used **-S** and **-n**, respectively, for the **-j** and **-N** options eventually selected. These were changed to avoid conflicts with historical implementations.

26554 The original standard specified **-t o2** as the default when no output type was given. This was
26555 changed to **-t oS** (the length of a **short**) to accommodate a supercomputer implementation that
26556 historically used 64 bits as its default (and that defined shorts as 64 bits). This change should not
26557 affect conforming applications. The requirement to support lengths of 1, 2, and 4 was added at
26558 the same time to address an historical implementation that had no two-byte data types in its C
26559 compiler.

26560 The use of a basic integer data type is intended to allow the implementation to choose a word
26561 size commonly used by applications on that architecture.

26562 Previous versions of this standard allowed for implementations with bytes other than eight bits, 2
26563 but this has been modified in this version. 2

26564 FUTURE DIRECTIONS

26565 All option and operand interfaces marked as extensions may be withdrawn in a future version.

26566 SEE ALSO

26567 *c99, sed*

26568 CHANGE HISTORY

26569 First released in Issue 2.

26570 Issue 5

26571 In the description of the **-c** option, the phrase “This is equivalent to **-t c.**” is deleted.

26572 The FUTURE DIRECTIONS section is modified.

26573 Issue 6

26574 The *od* utility is changed to remove the assumption that **short** was a two-byte entity, as per the
26575 revisions in the IEEE P1003.2b draft standard.

26576 The normative text is reworded to avoid use of the term “must” for application requirements.

26577 IEEE Std 1003.1-2001/Cor 1-2002, item XCU/TC1/D6/33 is applied, correcting the examples 1
26578 which used an undefined **-n** option, which should have been **-N**. 1

26579 IEEE Std 1003.1-2001/Cor 2-2004, item XCU/TC2/D6/19 is applied, removing text describing 2
26580 behavior on systems with bytes consisting of more than eight bits. 2

26581 NAME

26582 paste — merge corresponding or subsequent lines of files

26583 SYNOPSIS

26584 paste [-s][-d *list*] *file*...

26585 DESCRIPTION

26586 The *paste* utility shall concatenate the corresponding lines of the given input files, and write the
26587 resulting lines to standard output.26588 The default operation of *paste* shall concatenate the corresponding lines of the input files. The
26589 <newline> of every line except the line from the last input file shall be replaced with a <tab>.26590 If an end-of-file condition is detected on one or more input files, but not all input files, *paste* shall
26591 behave as though empty lines were read from the files on which end-of-file was detected, unless
26592 the **-s** option is specified.

26593 OPTIONS

26594 The *paste* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section
26595 12.2, Utility Syntax Guidelines.

26596 The following options shall be supported:

26597 **-d** *list* Unless a backslash character appears in *list*, each character in *list* is an element
26598 specifying a delimiter character. If a backslash character appears in *list*, the
26599 backslash character and one or more characters following it are an element
26600 specifying a delimiter character as described below. These elements specify one or
26601 more delimiters to use, instead of the default <tab>, to replace the <newline> of
26602 the input lines. The elements in *list* shall be used circularly; that is, when the list is
26603 exhausted the first element from the list is reused. When the **-s** option is specified:

- 26604 • The last <newline> in a file shall not be modified.
-
- 26605 • The delimiter shall be reset to the first element of
- list*
- after each
- file*
- operand is
-
- 26606 processed.

26607 When the **-s** option is not specified:

- 26608 • The <newline>s in the file specified by the last
- file*
- operand shall not be
-
- 26609 modified.
-
- 26610 • The delimiter shall be reset to the first element of
- list*
- each time a line is
-
- 26611 processed from each file.

26612 If a backslash character appears in *list*, it and the character following it shall be
26613 used to represent the following delimiter characters:

26614 \n <newline>.

26615 \t <tab>.

26616 \\ Backslash character.

26617 \0 Empty string (not a null character). If '\0' is immediately followed by the
26618 character 'x', the character 'X', or any character defined by the *LC_CTYPE*
26619 **digit** keyword (see the Base Definitions volume of IEEE Std 1003.1-2001,
26620 Chapter 7, Locale), the results are unspecified.

26621 If any other characters follow the backslash, the results are unspecified.

26622 **-s** Concatenate all of the lines of each separate input file in command line order. The
26623 <newline> of every line except the last line in each input file shall be replaced with

26624 the <tab>, unless otherwise specified by the **-d** option.

26625 OPERANDS

26626 The following operand shall be supported:

26627 *file* A pathname of an input file. If ‘–’ is specified for one or more of the *files*, the
26628 standard input shall be used; the standard input shall be read one line at a time,
26629 circularly, for each instance of ‘–’. Implementations shall support pasting of at
26630 least 12 *file* operands.

26631 STDIN

26632 The standard input shall be used only if one or more *file* operands is ‘–’. See the INPUT FILES
26633 section.

26634 INPUT FILES

26635 The input files shall be text files, except that line lengths shall be unlimited.

26636 ENVIRONMENT VARIABLES

26637 The following environment variables shall affect the execution of *paste*:

26638 *LANG* Provide a default value for the internationalization variables that are unset or null.
26639 (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2,
26640 Internationalization Variables for the precedence of internationalization variables
26641 used to determine the values of locale categories.)

26642 *LC_ALL* If set to a non-empty string value, override the values of all the other
26643 internationalization variables.

26644 *LC_CTYPE* Determine the locale for the interpretation of sequences of bytes of text data as
26645 characters (for example, single-byte as opposed to multi-byte characters in
26646 arguments and input files).

26647 *LC_MESSAGES*

26648 Determine the locale that should be used to affect the format and contents of
26649 diagnostic messages written to standard error.

26650 XSI *NLSPATH* Determine the location of message catalogs for the processing of *LC_MESSAGES*.

26651 ASYNCHRONOUS EVENTS

26652 Default.

26653 STDOUT

26654 Concatenated lines of input files shall be separated by the <tab> (or other characters under the
26655 control of the **-d** option) and terminated by a <newline>.

26656 STDERR

26657 The standard error shall be used only for diagnostic messages.

26658 OUTPUT FILES

26659 None.

26660 EXTENDED DESCRIPTION

26661 None.

26662 EXIT STATUS

26663 The following exit values shall be returned:

26664 0 Successful completion.

26665 >0 An error occurred.

26666 CONSEQUENCES OF ERRORS

26667 If one or more input files cannot be opened when the **-s** option is not specified, a diagnostic
26668 message shall be written to standard error, but no output is written to standard output. If the **-s**
26669 option is specified, the *paste* utility shall provide the default behavior described in Section 1.11
26670 (on page 20).

26671 APPLICATION USAGE

26672 When the escape sequences of the *list* option-argument are used in a shell script, they must be
26673 quoted; otherwise, the shell treats the '\' as a special character.

26674 Conforming applications should only use the specific backslash escaped delimiters presented in
26675 this volume of IEEE Std 1003.1-2001. Historical implementations treat '\x', where 'x' is not in
26676 this list, as 'x', but future implementations are free to expand this list to recognize other
26677 common escapes similar to those accepted by *printf* and other standard utilities.

26678 Most of the standard utilities work on text files. The *cut* utility can be used to turn files with
26679 arbitrary line lengths into a set of text files containing the same data. The *paste* utility can be used
26680 to create (or recreate) files with arbitrary line lengths. For example, if *file* contains long lines:

```
26681 cut -b 1-500 -n file > file1
26682 cut -b 501- -n file > file2
```

26683 creates **file1** (a text file) with lines no longer than 500 bytes (plus the <newline>) and **file2** that
26684 contains the remainder of the data from *file*. Note that **file2** is not a text file if there are lines in
26685 *file* that are longer than 500 + {LINE_MAX} bytes. The original file can be recreated from **file1**
26686 and **file2** using the command:

```
26687 paste -d "\0" file1 file2 > file
```

26688 The commands:

```
26689 paste -d "\0" ...
26690 paste -d "" ...
```

26691 are not necessarily equivalent; the latter is not specified by this volume of IEEE Std 1003.1-2001
26692 and may result in an error. The construct '\0' is used to mean "no separator" because
26693 historical versions of *paste* did not follow the syntax guidelines, and the command:

```
26694 paste -d "" ...
```

26695 could not be handled properly by *getopt()*.

26696 EXAMPLES

26697 1. Write out a directory in four columns:

```
26698 ls | paste - - -
```

26699 2. Combine pairs of lines from a file into single lines:

```
26700 paste -s -d "\t\n" file
```

26701 RATIONALE

26702 None.

26703 FUTURE DIRECTIONS

26704 None.

26705 SEE ALSO

26706 Section 1.11 (on page 20), *cut*, *grep*, *pr*

26707 CHANGE HISTORY

26708 First released in Issue 2.

26709 Issue 6

26710 The normative text is reworded to avoid use of the term “must” for application requirements.

26711 NAME

26712 patch — apply changes to files

26713 SYNOPSIS

```
26714 UP    patch [-blNR][ -c| -e| -n][-d dir][-D define][-i patchfile]
26715           [-o outfile][-p num][-r rejectfile][file]
```

26716

26717 DESCRIPTION

26718 The *patch* utility shall read a source (patch) file containing any of the three forms of difference
 26719 (diff) listings produced by the *diff* utility (normal, context, or in the style of *ed*) and apply those
 26720 differences to a file. By default, *patch* shall read from the standard input.

26721 The *patch* utility shall attempt to determine the type of the *diff* listing, unless overruled by a **-c**,
 26722 **-e**, or **-n** option.

26723 If the patch file contains more than one patch, *patch* shall attempt to apply each of them as if they
 26724 came from separate patch files. (In this case, the application shall ensure that the name of the
 26725 patch file is determinable for each *diff* listing.)

26726 OPTIONS

26727 The *patch* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section
 26728 12.2, Utility Syntax Guidelines.

26729 The following options shall be supported:

26730 **-b** Save a copy of the original contents of each modified file, before the differences are
 26731 applied, in a file of the same name with the suffix **.orig** appended to it. If the file
 26732 already exists, it shall be overwritten; if multiple patches are applied to the same
 26733 file, the **.orig** file shall be written only for the first patch. When the **-o** *outfile* option
 26734 is also specified, *file.orig* shall not be created but, if *outfile* already exists,
 26735 *outfile.orig* shall be created.

26736 **-c** Interpret the patch file as a context difference (the output of the utility *diff* when
 26737 the **-c** or **-C** options are specified).

26738 **-d** *dir* Change the current directory to *dir* before processing as described in the
 26739 EXTENDED DESCRIPTION section.

26740 **-D** *define* Mark changes with one of the following C preprocessor constructs:

```
26741     #ifdef define
26742     ...
26743     #endif
26744     #ifndef define
26745     ...
26746     #endif
```

26747 optionally combined with the C preprocessor construct **#else**. If the patched file is 1
 26748 processed with the C preprocessor, where the macro *define* is defined, the output 1
 26749 shall contain the changes from the patch file; otherwise, the output shall not 1
 26750 contain the patches specified in the patch file. 1

26751 **-e** Interpret the patch file as an *ed* script, rather than a *diff* script.

26752 **-i** *patchfile* Read the patch information from the file named by the pathname *patchfile*, rather
 26753 than the standard input.

26754	-l	(The letter ell.) Cause any sequence of <blank>s in the difference script to match any sequence of <blank>s in the input file. Other characters shall be matched exactly.
26757	-n	Interpret the script as a normal difference.
26758	-N	Ignore patches where the differences have already been applied to the file; by default, already-applied patches shall be rejected.
26760	-o <i>outfile</i>	Instead of modifying the files (specified by the <i>file</i> operand or the difference listings) directly, write a copy of the file referenced by each patch, with the appropriate differences applied, to <i>outfile</i> . Multiple patches for a single file shall be applied to the intermediate versions of the file created by any previous patches, and shall result in multiple, concatenated versions of the file being written to <i>outfile</i> .
26766	-p <i>num</i>	For all pathnames in the patch file that indicate the names of files to be patched, delete <i>num</i> pathname components from the beginning of each pathname. If the pathname in the patch file is absolute, any leading slashes shall be considered the first component (that is, -p 1 shall remove the leading slashes). Specifying -p 0 shall cause the full pathname to be used. If -p is not specified, only the basename (the final pathname component) shall be used.
26772	-R	Reverse the sense of the patch script; that is, assume that the difference script was created from the new version to the old version. The -R option cannot be used with <i>ed</i> scripts. The <i>patch</i> utility shall attempt to reverse each portion of the script before applying it. Rejected differences shall be saved in swapped format. If this option is not specified, and until a portion of the patch file is successfully applied, <i>patch</i> attempts to apply each portion in its reversed sense as well as in its normal sense. If the attempt is successful, the user shall be prompted to determine whether the -R option should be set.
26780	-r <i>rejectfile</i>	Override the default reject filename. In the default case, the reject file shall have the same name as the output file, with the suffix <i>.rej</i> appended to it; see Patch Application (on page 693).

26783 OPERANDS

26784	The following operand shall be supported:
26785	<i>file</i> A pathname of a file to patch.

26786 STDIN

26787 See the INPUT FILES section.

26788 INPUT FILES

26789 Input files shall be text files.

26790 ENVIRONMENT VARIABLES

26791 The following environment variables shall affect the execution of *patch*:

26792	LANG	Provide a default value for the internationalization variables that are unset or null. (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2, Internationalization Variables for the precedence of internationalization variables used to determine the values of locale categories.)
26796	LC_ALL	If set to a non-empty string value, override the values of all the other internationalization variables.

26798	<i>LC_CTYPE</i>	Determine the locale for the interpretation of sequences of bytes of text data as characters (for example, single-byte as opposed to multi-byte characters in arguments and input files).
26801	<i>LC_MESSAGES</i>	Determine the locale that should be used to affect the format and contents of diagnostic messages written to standard error and informative messages written to standard output.
26805 XSI	<i>NLSPATH</i>	Determine the location of message catalogs for the processing of <i>LC_MESSAGES</i> .
26806	<i>LC_TIME</i>	Determine the locale for recognizing the format of file timestamps written by the <i>diff</i> utility in a context-difference input file.
26808	ASYNCHRONOUS EVENTS	
26809		Default.
26810	STDOUT	
26811		Not used.
26812	STDERR	
26813		The standard error shall be used for diagnostic and informational messages.
26814	OUTPUT FILES	
26815		The output of the <i>patch</i> utility, the save files (.orig suffixes), and the reject files (.rej suffixes) shall be text files.
26817	EXTENDED DESCRIPTION	
26818		A patch file may contain patching instructions for more than one file; filenames shall be determined as specified in Filename Determination (on page 693). When the -b option is specified, for each patched file, the original shall be saved in a file of the same name with the suffix .orig appended to it.
26822		For each patched file, a reject file may also be created as noted in Patch Application (on page 693). In the absence of a -r option, the name of this file shall be formed by appending the suffix .rej to the original filename.
26825	Patch File Format	
26826		The patch file shall contain zero or more lines of header information followed by one or more patches. Each patch shall contain zero or more lines of filename identification in the format produced by <i>diff -c</i> , and one or more sets of <i>diff</i> output, which are customarily called <i>hunks</i> .
26829		The <i>patch</i> utility shall recognize the following expression in the header information:
26830	Index: pathname	
26831		The file to be patched is named <i>pathname</i> .
26832		If all lines (including headers) within a patch begin with the same leading sequence of <blank>s, the <i>patch</i> utility shall remove this sequence before proceeding. Within each patch, if the type of difference is context, the <i>patch</i> utility shall recognize the following expressions:
26835	*** filename timestamp	
26836		The patches arose from <i>filename</i> .
26837	-- filename timestamp	
26838		The patches should be applied to <i>filename</i> .
26839		Each hunk within a patch shall be the <i>diff</i> output to change a line range within the original file. The line numbers for successive hunks within a patch shall occur in ascending order.
26840		

26841 Filename Determination

26842 If no *file* operand is specified, *patch* shall perform the following steps to determine the filename
26843 to use:

- 26844 1. If the type of *diff* is context, the *patch* utility shall delete pathname components (as
26845 specified by the **-p** option) from the filename on the line beginning with " *** ", then test
26846 for the existence of this file relative to the current directory (or the directory specified with
26847 the **-d** option). If the file exists, the *patch* utility shall use this filename.
- 26848 2. If the type of *diff* is context, the *patch* utility shall delete the pathname components (as
26849 specified by the **-p** option) from the filename on the line beginning with " --- ", then test
26850 for the existence of this file relative to the current directory (or the directory specified with
26851 the **-d** option). If the file exists, the *patch* utility shall use this filename.
- 26852 3. If the header information contains a line beginning with the string **Index:**, the *patch* utility
26853 shall delete pathname components (as specified by the **-p** option) from this line, then test
26854 for the existence of this file relative to the current directory (or the directory specified with
26855 the **-d** option). If the file exists, the *patch* utility shall use this filename.
- 26856 XSI 4. If an **SCCS** directory exists in the current directory, *patch* shall attempt to perform a **get -e**
26857 **SCCS/s.filename** command to retrieve an editable version of the file. If the file exists, the
26858 *patch* utility shall use this filename.
- 26859 5. The *patch* utility shall write a prompt to standard output and request a filename
26860 interactively from the controlling terminal (for example, **/dev/tty**).

26861 Patch Application

26862 If the **-c**, **-e**, or **-n** option is present, the *patch* utility shall interpret information within each hunk
26863 as a context difference, an *ed* difference, or a normal difference, respectively. In the absence of
26864 any of these options, the *patch* utility shall determine the type of difference based on the format
26865 of information within the hunk.

26866 For each hunk, the *patch* utility shall begin to search for the place to apply the patch at the line
26867 number at the beginning of the hunk, plus or minus any offset used in applying the previous
26868 hunk. If lines matching the hunk context are not found, *patch* shall scan both forwards and
26869 backwards at least 1 000 bytes for a set of lines that match the hunk context.

26870 If no such place is found and it is a context difference, then another scan shall take place,
26871 ignoring the first and last line of context. If that fails, the first two and last two lines of context
26872 shall be ignored and another scan shall be made. Implementations may search more extensively
26873 for installation locations.

26874 If no location can be found, the *patch* utility shall append the hunk to the reject file. The rejected
26875 hunk shall be written in context-difference format regardless of the format of the patch file. If the
26876 input was a normal or *ed*-style difference, the reject file may contain differences with zero lines
26877 of context. The line numbers on the hunks in the reject file may be different from the line
26878 numbers in the patch file since they shall reflect the approximate locations for the failed hunks in
26879 the new file rather than the old one.

26880 If the type of patch is an *ed* diff, the implementation may accomplish the patching by invoking
26881 the *ed* utility.

26882 EXIT STATUS

26883 The following exit values shall be returned:

26884 0 Successful completion.

26885 1 One or more lines were written to a reject file.

26886 >1 An error occurred.

26887 CONSEQUENCES OF ERRORS

26888 Patches that cannot be correctly placed in the file shall be written to a reject file.

26889 APPLICATION USAGE

26890 The **-R** option does not work with *ed* scripts because there is too little information to reconstruct
26891 the reverse operation.

26892 The **-p** option makes it possible to customize a patch file to local user directory structures
26893 without manually editing the patch file. For example, if the filename in the patch file was:

26894 /curds/whey/src/blurf1/blurf1.c

26895 Setting **-p 0** gives the entire pathname unmodified; **-p 1** gives:

26896 curds/whey/src/blurf1/blurf1.c

26897 without the leading slash, **-p 4** gives:

26898 blurf1/blurf1.c

26899 and not specifying **-p** at all gives:

26900 blurf1.c .

26901 EXAMPLES

26902 None.

26903 RATIONALE

26904 Some of the functionality in historical *patch* implementations was not specified. The following
26905 documents those features present in historical implementations that have not been specified.

26906 A deleted piece of functionality was the '+' pseudo-option allowing an additional set of options
26907 and a patch file operand to be given. This was seen as being insufficiently useful to standardize.

26908 In historical implementations, if the string "Prereq:" appeared in the header, the *patch* utility
26909 would search for the corresponding version information (the string specified in the header,
26910 delimited by <blank>s or the beginning or end of a line or the file) anywhere in the original file.
26911 This was deleted as too simplistic and insufficiently trustworthy a mechanism to standardize.
26912 For example, if:

26913 Prereq: 1.2

26914 were in the header, the presence of a delimited 1.2 anywhere in the file would satisfy the
26915 prerequisite.

26916 The following options were dropped from historical implementations of *patch* as insufficiently
26917 useful to standardize:

26918 **-b** The **-b** option historically provided a method for changing the name extension of
26919 the backup file from the default **.orig**. This option has been modified and retained
26920 in this volume of IEEE Std 1003.1-2001.

26921 **-F** The **-F** option specified the number of lines of a context diff to ignore when
26922 searching for a place to install a patch.

26923 **-f** The **-f** option historically caused *patch* not to request additional information from
26924 the user.

26925	-r	The -r option historically provided a method of overriding the extension of the reject file from the default .rej .
26927	-s	The -s option historically caused <i>patch</i> to work silently unless an error occurred.
26928	-x	The -x option historically set internal debugging flags.
26929		In some file system implementations, the saving of a .orig file may produce unwanted results. In the case of 12, 13, or 14-character filenames (on file systems supporting 14-character maximum filenames), the .orig file overwrites the new file. The reject file may also exceed this filename limit. It was suggested, due to some historical practice, that a tilde ('~') suffix be used instead of .orig and some other character instead of the .rej suffix. This was rejected because it is not obvious to the user which file is which. The suffixes .orig and .rej are clearer and more understandable.
26936		The -b option has the opposite sense in some historical implementations—do not save the .orig file. The default case here is not to save the files, making <i>patch</i> behave more consistently with the other standard utilities.
26939		The -w option in early proposals was changed to -l to match historical practice.
26940		The -N option was included because without it, a non-interactive application cannot reject previously applied patches. For example, if a user is piping the output of <i>diff</i> into the <i>patch</i> utility, and the user only wants to patch a file to a newer version non-interactively, the -N option is required.
26944		Changes to the -l option description were proposed to allow matching across <newline>s in addition to just <blank>s. Since this is not historical practice, and since some ambiguities could result, it is suggested that future developments in this area utilize another option letter, such as -L .
26948	FUTURE DIRECTIONS	
26949	None.	
26950	SEE ALSO	
26951	<i>ed</i> , <i>diff</i>	
26952	CHANGE HISTORY	
26953	First released in Issue 4.	
26954	Issue 5	
26955	The FUTURE DIRECTIONS section is added.	
26956	Issue 6	
26957	This utility is marked as part of the User Portability Utilities option.	
26958	The description of the -D option and the steps in Filename Determination (on page 693) are changed to match historical practice as defined in the IEEE P1003.2b draft standard.	
26960	The normative text is reworded to avoid use of the term “must” for application requirements.	
26961	IEEE Std 1003.1-2001/Cor 1-2002, item XCU/TC1/D6/34 is applied, clarifying the way that the <i>patch</i> utility performs ifdef selection for the -D option.	1
26962		

26963 **NAME**

26964 pathchk — check pathnames

26965 **SYNOPSIS**

26966 pathchk [-p] pathname...

26967 **DESCRIPTION**

26968 The *pathchk* utility shall check that one or more pathnames are valid (that is, they could be used
26969 to access or create a file without causing syntax errors) and portable (that is, no filename
26970 truncation results). More extensive portability checks are provided by the **-p** option.

26971 By default, the *pathchk* utility shall check each component of each *pathname* operand based on the
26972 underlying file system. A diagnostic shall be written for each *pathname* operand that:

- 26973 • Is longer than {PATH_MAX} bytes (see **Pathname Variable Values** in the Base Definitions
26974 volume of IEEE Std 1003.1-2001, Chapter 13, Headers, <limits.h>)
- 26975 • Contains any component longer than {NAME_MAX} bytes in its containing directory
- 26976 • Contains any component in a directory that is not searchable
- 26977 • Contains any character in any component that is not valid in its containing directory

26978 The format of the diagnostic message is not specified, but shall indicate the error detected and
26979 the corresponding *pathname* operand.

26980 It shall not be considered an error if one or more components of a *pathname* operand do not exist
26981 as long as a file matching the pathname specified by the missing components could be created
26982 that does not violate any of the checks specified above.

26983 **OPTIONS**

26984 The *pathchk* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section
26985 12.2, Utility Syntax Guidelines.

26986 The following option shall be supported:

26987 **-p** Instead of performing checks based on the underlying file system, write a
26988 diagnostic for each *pathname* operand that:

- 26989 • Is longer than {_POSIX_PATH_MAX} bytes (see **Minimum Values** in the Base
26990 Definitions volume of IEEE Std 1003.1-2001, Chapter 13, Headers, <limits.h>)
- 26991 • Contains any component longer than {_POSIX_NAME_MAX} bytes
- 26992 • Contains any character in any component that is not in the portable filename
26993 character set

26994 **OPERANDS**

26995 The following operand shall be supported:

26996 *pathname* A pathname to be checked.

26997 **STDIN**

26998 Not used.

26999 **INPUT FILES**

27000 None.

27001 **ENVIRONMENT VARIABLES**

27002 The following environment variables shall affect the execution of *pathchk*:

27003 **LANG** Provide a default value for the internationalization variables that are unset or null.
27004 (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2,

27005		Internationalization Variables for the precedence of internationalization variables used to determine the values of locale categories.)
27006		
27007	<i>LC_ALL</i>	If set to a non-empty string value, override the values of all the other internationalization variables.
27008		
27009	<i>LC_CTYPE</i>	Determine the locale for the interpretation of sequences of bytes of text data as characters (for example, single-byte as opposed to multi-byte characters in arguments).
27010		
27011		
27012	<i>LC_MESSAGES</i>	
27013		Determine the locale that should be used to affect the format and contents of diagnostic messages written to standard error.
27014		
27015	XSI <i>NLSPATH</i>	Determine the location of message catalogs for the processing of <i>LC_MESSAGES</i> .
27016	ASYNCHRONOUS EVENTS	
27017		Default.
27018	STDOUT	
27019		Not used.
27020	STDERR	
27021		The standard error shall be used only for diagnostic messages.
27022	OUTPUT FILES	
27023		None.
27024	EXTENDED DESCRIPTION	
27025		None.
27026	EXIT STATUS	
27027		The following exit values shall be returned:
27028		0 All <i>pathname</i> operands passed all of the checks.
27029		>0 An error occurred.
27030	CONSEQUENCES OF ERRORS	
27031		Default.
27032	APPLICATION USAGE	
27033		The <i>test</i> utility can be used to determine whether a given pathname names an existing file; it does not, however, give any indication of whether or not any component of the pathname was truncated in a directory where the <i>_POSIX_NO_TRUNC</i> feature is not in effect. The <i>pathchk</i> utility does not check for file existence; it performs checks to determine whether a pathname does exist or could be created with no pathname component truncation.
27034		
27035		
27036		
27037		
27038		The <i>noclobber</i> option in the shell (see the <i>set</i> special built-in) can be used to atomically create a file. As with all file creation semantics in the System Interfaces volume of IEEE Std 1003.1-2001, it guarantees atomic creation, but still depends on applications to agree on conventions and cooperate on the use of files after they have been created.
27039		
27040		
27041		
27042	EXAMPLES	
27043		To verify that all pathnames in an imported data interchange archive are legitimate and unambiguous on the current system:
27044		
27045		<pre>pax -f archive sed -e '/ == .*\/s///' xargs pathchk</pre>
27046		<pre>if [\$? -eq 0]</pre>
27047		<pre>then</pre>
27048		<pre> pax -r -f archive</pre>

```

27049     else
27050         echo Investigate problems before importing files.
27051         exit 1
27052     fi

```

To verify that all files in the current directory hierarchy could be moved to any system conforming to the System Interfaces volume of IEEE Std 1003.1-2001 that also supports the *pax* utility:

```

27056     find . -print | xargs pathchk -p
27057     if [ $? -eq 0 ]
27058     then
27059         pax -w -f archive .
27060     else
27061         echo Portable archive cannot be created.
27062         exit 1
27063     fi

```

To verify that a user-supplied pathname names a readable file and that the application can create a file extending the given path without truncation and without overwriting any existing file:

```

27066     case $- in
27067         *C*)      reset="" ;;
27068         *)        reset="set +C"
27069             set -C;;
27070     esac
27071     test -r "$path" && pathchk "$path.out" &&
27072         rm "$path.out" > "$path.out"
27073     if [ $? -ne 0 ]; then
27074         printf "%s: %s not found or %s.out fails \
27075 creation checks.\n" $0 "$path" "$path"
27076         $reset      # Reset the noclobber option in case a trap
27077             # on EXIT depends on it.
27078         exit 1
27079     fi
27080     $reset
27081     PROCESSING < "$path" > "$path.out"

```

The following assumptions are made in this example:

1. **PROCESSING** represents the code that is used by the application to use **\$path** once it is verified that **\$path.out** works as intended.
2. The state of the *noclobber* option is unknown when this code is invoked and should be set on exit to the state it was in when this code was invoked. (The **reset** variable is used in this example to restore the initial state.)
3. Note the usage of:

```
    rm "$path.out" > "$path.out"
```

- a. The *pathchk* command has already verified, at this point, that **\$path.out** is not truncated.
- b. With the *noclobber* option set, the shell verifies that **\$path.out** does not already exist before invoking *rm*.

- 27094 c. If the shell succeeded in creating `$path.out`, `rm` removes it so that the application can
27095 create the file again in the **PROCESSING** step.
27096 d. If the **PROCESSING** step wants the file to exist already when it is invoked, the:
27097 `rm "$path.out" > "$path.out"`
27098 should be replaced with:
27099 `> "$path.out"`
27100 which verifies that the file did not already exist, but leaves `$path.out` in place for use
27101 by **PROCESSING**.

27102 RATIONALE

27103 The `pathchk` utility was new for the ISO POSIX-2:1993 standard. It, along with the `set`
27104 `-C(noclobber)` option added to the shell, replaces the `mktemp`, `validfnam`, and `create` utilities that
27105 appeared in early proposals. All of these utilities were attempts to solve several common
27106 problems:

- 27107 • Verify the validity (for several different definitions of “valid”) of a pathname supplied by a
27108 user, generated by an application, or imported from an external source.
- 27109 • Atomically create a file.
- 27110 • Perform various string handling functions to generate a temporary filename.

27111 The `create` utility, included in an early proposal, provided checking and atomic creation in a
27112 single invocation of the utility; these are orthogonal issues and need not be grouped into a single
27113 utility. Note that the `noclobber` option also provides a way of creating a lock for process
27114 synchronization; since it provides an atomic `create`, there is no race between a test for existence
27115 and the following creation if it did not exist.

27116 Having a function like `tmpnam()` in the ISO C standard is important in many high-level
27117 languages. The shell programming language, however, has built-in string manipulation
27118 facilities, making it very easy to construct temporary filenames. The names needed obviously
27119 depend on the application, but are frequently of a form similar to:

27120 `$TMPDIR/application_abbreviation$$.suffix`

27121 In cases where there is likely to be contention for a given suffix, a simple shell **for** or **while** loop
27122 can be used with the shell `noclobber` option to create a file without risk of collisions, as long as
27123 applications trying to use the same filename name space are cooperating on the use of files after
27124 they have been created.

27125 FUTURE DIRECTIONS

27126 None.

27127 SEE ALSO

27128 Section 2.7 (on page 43), `set` (on page 86), `test`

27129 CHANGE HISTORY

27130 First released in Issue 4.

27131 NAME

27132 pax — portable archive interchange

27133 SYNOPSIS

27134 pax [-cdnv][-H -L][-o options][-f archive][-s replstr]...	2
27135 [pattern...]	2
27136 pax -r[-cdiknuv][-H -L][-f archive][-o options]...[-p string]...	
27137 [-s replstr]...[pattern...]	
27138 pax -w[-dituvX][-H -L][-b blocksize][[-a][-f archive][-o options]...]	
27139 [-s replstr]...[-x format][file...]	
27140 pax -r -w[-diklntuvX][-H -L][-o options]...[-p string]...	2
27141 [-s replstr]...[file...] directory	2

27142 DESCRIPTION

27143 The *pax* utility shall read, write, and write lists of the members of archive files and copy
 27144 directory hierarchies. A variety of archive formats shall be supported; see the **-x format** option.

27145 The action to be taken depends on the presence of the **-r** and **-w** options. The four combinations
 27146 of **-r** and **-w** are referred to as the four modes of operation: **list**, **read**, **write**, and **copy** modes,
 27147 corresponding respectively to the four forms shown in the SYNOPSIS section.

27148 **list** In **list** mode (when neither **-r** nor **-w** are specified), *pax* shall write the names of
 27149 the members of the archive file read from the standard input, with pathnames
 27150 matching the specified patterns, to standard output. If a named file is of type
 27151 directory, the file hierarchy rooted at that file shall be listed as well.

27152 **read** In **read** mode (when **-r** is specified, but **-w** is not), *pax* shall extract the members of
 27153 the archive file read from the standard input, with pathnames matching the
 27154 specified patterns. If an extracted file is of type directory, the file hierarchy rooted
 27155 at that file shall be extracted as well. The extracted files shall be created performing
 27156 pathname resolution with the directory in which *pax* was invoked as the current
 27157 working directory.

27158 If an attempt is made to extract a directory when the directory already exists, this
 27159 shall not be considered an error. If an attempt is made to extract a FIFO when the
 27160 FIFO already exists, this shall not be considered an error.

27161 The ownership, access, and modification times, and file mode of the restored files
 27162 are discussed under the **-p** option.

27163 **write** In **write** mode (when **-w** is specified, but **-r** is not), *pax* shall write the contents of
 27164 the *file* operands to the standard output in an archive format. If no *file* operands are
 27165 specified, a list of files to copy, one per line, shall be read from the standard input.
 27166 A file of type directory shall include all of the files in the file hierarchy rooted at the
 27167 file.

27168 **copy** In **copy** mode (when both **-r** and **-w** are specified), *pax* shall copy the *file* operands
 27169 to the destination directory.

27170 If no *file* operands are specified, a list of files to copy, one per line, shall be read
 27171 from the standard input. A file of type directory shall include all of the files in the
 27172 file hierarchy rooted at the file.

27173 The effect of the **copy** shall be as if the copied files were written to an archive file
 27174 and then subsequently extracted, except that there may be hard links between the
 27175 original and the copied files. If the destination directory is a subdirectory of one of

27176 the files to be copied, the results are unspecified. If the destination directory is a
27177 file of a type not defined by the System Interfaces volume of IEEE Std 1003.1-2001,
27178 the results are implementation-defined; otherwise, it shall be an error for the file
27179 named by the *directory* operand not to exist, not be writable by the user, or not be a
27180 file of type directory.

27181 In **read** or **copy** modes, if intermediate directories are necessary to extract an archive member,
27182 *pax* shall perform actions equivalent to the *mkdir()* function defined in the System Interfaces
27183 volume of IEEE Std 1003.1-2001, called with the following arguments:

- 27184 • The intermediate directory used as the *path* argument
- 27185 • The value of the bitwise-inclusive OR of S_IRWXU, S_IRWXG, and S_IRWXO as the *mode*
27186 argument

27187 If any specified *pattern* or *file* operands are not matched by at least one file or archive member,
27188 *pax* shall write a diagnostic message to standard error for each one that did not match and exit
27189 with a non-zero exit status.

27190 The archive formats described in the EXTENDED DESCRIPTION section shall be automatically
27191 detected on input. The default output archive format shall be implementation-defined.

27192 A single archive can span multiple files. The *pax* utility shall determine, in an implementation-
27193 defined manner, what file to read or write as the next file.

27194 If the selected archive format supports the specification of linked files, it shall be an error if these
27195 files cannot be linked when the archive is extracted, except that if the files to be linked are
27196 symbolic links and the system is not capable of making hard links to symbolic links, then
27197 separate copies of the symbolic link shall be created instead. For archive formats that do not
27198 store file contents with each name that causes a hard link, if the file that contains the data is not
27199 extracted during this *pax* session, either the data shall be restored from the original file, or a
27200 diagnostic message shall be displayed with the name of a file that can be used to extract the data.
27201 In traversing directories, *pax* shall detect infinite loops; that is, entering a previously visited
27202 directory that is an ancestor of the last file visited. When it detects an infinite loop, *pax* shall
27203 write a diagnostic message to standard error and shall terminate.

27204 OPTIONS

27205 The *pax* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section
27206 12.2, Utility Syntax Guidelines, except that the order of presentation of the **-o**, **-p**, and **-s** options
27207 is significant.

27208 The following options shall be supported:

- 27209 **-r** Read an archive file from standard input.
- 27210 **-w** Write files to the standard output in the specified archive format.
- 27211 **-a** Append files to the end of the archive. It is implementation-defined which devices
27212 on the system support appending. Additional file formats unspecified by this
27213 volume of IEEE Std 1003.1-2001 may impose restrictions on appending.
- 27214 **-b blocksize** Block the output at a positive decimal integer number of bytes per write to the
27215 archive file. Devices and archive formats may impose restrictions on blocking.
27216 Blocking shall be automatically determined on input. Conforming applications
27217 shall not specify a *blocksize* value larger than 32 256. Default blocking when
27218 creating archives depends on the archive format. (See the **-x** option below.)
- 27219 **-c** Match all file or archive members except those specified by the *pattern* or *file*
27220 operands.

27221	-d	Cause files of type directory being copied or archived or archive members of type directory being extracted or listed to match only the file or archive member itself and not the file hierarchy rooted at the file.
27222		
27223		
27224	-f archive	Specify the pathname of the input or output archive, overriding the default standard input (in list or read modes) or standard output (write mode).
27225		
27226	-H	If a symbolic link referencing a file of type directory is specified on the command line, <i>pax</i> shall archive the file hierarchy rooted in the file referenced by the link, using the name of the link as the root of the file hierarchy. Otherwise, if a symbolic link referencing a file of any other file type which <i>pax</i> can normally archive is specified on the command line, then <i>pax</i> shall archive the file referenced by the link, using the name of the link. The default behavior shall be to archive the symbolic link itself.
27227		
27228		
27229		
27230		
27231		
27232		
27233	-i	Interactively rename files or archive members. For each archive member matching a <i>pattern</i> operand or file matching a <i>file</i> operand, a prompt shall be written to the file /dev/tty . The prompt shall contain the name of the file or archive member, but the format is otherwise unspecified. A line shall then be read from /dev/tty . If this line is blank, the file or archive member shall be skipped. If this line consists of a single period, the file or archive member shall be processed with no modification to its name. Otherwise, its name shall be replaced with the contents of the line. The <i>pax</i> utility shall immediately exit with a non-zero exit status if end-of-file is encountered when reading a response or if /dev/tty cannot be opened for reading and writing.
27234		
27235		
27236		
27237		
27238		
27239		
27240		
27241		
27242		
27243		The results of extracting a hard link to a file that has been renamed during extraction are unspecified.
27244		
27245	-k	Prevent the overwriting of existing files.
27246	-l	(The letter ell.) In copy mode, hard links shall be made between the source and destination file hierarchies whenever possible. If specified in conjunction with -H or -L , when a symbolic link is encountered, the hard link created in the destination file hierarchy shall be to the file referenced by the symbolic link. If specified when neither -H nor -L is specified, when a symbolic link is encountered, the implementation shall create a hard link to the symbolic link in the source file hierarchy or copy the symbolic link to the destination.
27247		
27248		
27249		
27250		
27251		
27252		
27253	-L	If a symbolic link referencing a file of type directory is specified on the command line or encountered during the traversal of a file hierarchy, <i>pax</i> shall archive the file hierarchy rooted in the file referenced by the link, using the name of the link as the root of the file hierarchy. Otherwise, if a symbolic link referencing a file of any other file type which <i>pax</i> can normally archive is specified on the command line or encountered during the traversal of a file hierarchy, <i>pax</i> shall archive the file referenced by the link, using the name of the link. The default behavior shall be to archive the symbolic link itself.
27254		
27255		
27256		
27257		
27258		
27259		
27260		
27261	-n	Select the first archive member that matches each <i>pattern</i> operand. No more than one archive member shall be matched for each pattern (although members of type directory shall still match the file hierarchy rooted at that file).
27262		
27263		
27264	-o options	Provide information to the implementation to modify the algorithm for extracting or writing files. The value of <i>options</i> shall consist of one or more comma-separated keywords of the form:
27265		
27266		

27267	<code>keyword[[[:]=value][, keyword[[[:]=value], ...]</code>	
27268	Some keywords apply only to certain file formats, as indicated with each	
27269	description. Use of keywords that are inapplicable to the file format being	
27270	processed produces undefined results.	
27271	Keywords in the <i>options</i> argument shall be a string that would be a valid portable	
27272	filename as described in the Base Definitions volume of IEEE Std 1003.1-2001,	
27273	Section 3.276, Portable Filename Character Set.	
27274	Note: Keywords are not expected to be filenames, merely to follow the same character	
27275	composition rules as portable filenames.	
27276	Keywords can be preceded with white space. The <i>value</i> field shall consist of zero or	
27277	more characters; within <i>value</i> , the application shall precede any literal comma with	
27278	a backslash, which shall be ignored, but preserves the comma as part of <i>value</i> . A	
27279	comma as the final character, or a comma followed solely by white space as the	
27280	final characters, in <i>options</i> shall be ignored. Multiple -o options can be specified; if	
27281	keywords given to these multiple -o options conflict, the keywords and values	
27282	appearing later in command line sequence shall take precedence and the earlier	
27283	shall be silently ignored. The following keyword values of <i>options</i> shall be	
27284	supported for the file formats as indicated:	
27285	delete=pattern	
27286	(Applicable only to the -x pax format.) When used in write or copy mode, <i>pax</i>	
27287	shall omit from extended header records that it produces any keywords	
27288	matching the string pattern. When used in read or list mode, <i>pax</i> shall ignore	
27289	any keywords matching the string pattern in the extended header records. In	
27290	both cases, matching shall be performed using the pattern matching notation	
27291	described in Section 2.13.1 (on page 62) and Section 2.13.2 (on page 63). For	
27292	example:	
27293	-o delete=security.*	
27294	would suppress security-related information. See pax Extended Header (on	
27295	page 713) for extended header record keyword usage.	
27296	When multiple -o delete=pattern options are specified, the patterns shall be	2
27297	additive; all keywords matching the specified string patterns shall be omitted	2
27298	from extended header records that <i>pax</i> produces.	2
27299	exthdr.name=string	
27300	(Applicable only to the -x pax format.) This keyword allows user control over	
27301	the name that is written into the ustar header blocks for the extended header	
27302	produced under the circumstances described in pax Header Block (on page	
27303	713). The name shall be the contents of <i>string</i> , after the following character	
27304	substitutions have been made:	

27305

27306

27307

<i>string</i> Includes:	Replaced By:
%d	The directory name of the file, equivalent to the result of the <i>dirname</i> utility on the translated pathname.
%f	The filename of the file, equivalent to the result of the <i>basename</i> utility on the translated pathname.
%p	The process ID of the <i>pax</i> process.
%%	A '%' character.

27314

Any other '%' characters in *string* produce undefined results.

27315

27316

If no **-o exthdr.name=string** is specified, *pax* shall use the following default value:

27317

%d/PaxHeaders.%p/%f

1

27318

globexthdr.name=string

27319

27320

27321

27322

27323

27324

27325

(Applicable only to the **-x pax** format.) When used in **write** or **copy** mode with the appropriate options, *pax* shall create global extended header records with **ustar** header blocks that will be treated as regular files by previous versions of *pax*. This keyword allows user control over the name that is written into the **ustar** header blocks for global extended header records. The name shall be the contents of *string*, after the following character substitutions have been made:

27326

27327

<i>string</i> Includes:	Replaced By:
%n	An integer that represents the sequence number of the global extended header record in the archive, starting at 1.
%p	The process ID of the <i>pax</i> process.
%%	A '%' character.

27332

Any other '%' characters in *string* produce undefined results.

27333

27334

If no **-o globexthdr.name=string** is specified, *pax* shall use the following default value:

27335

\$TMPDIR/GlobalHead.%p.%n

1

27336

27337

where \$TMPDIR represents the value of the *TMPDIR* environment variable. If *TMPDIR* is not set, *pax* shall use */tmp*.

27338

invalid=action

27339

27340

27341

27342

27343

27344

(Applicable only to the **-x pax** format.) This keyword allows user control over the action *pax* takes upon encountering values in an extended header record that, in **read** or **copy** mode, are invalid in the destination hierarchy or, in **list** mode, cannot be written in the codeset and current locale of the implementation. The following are invalid values that shall be recognized by *pax*:

27345

27346

27347

- In **read** or **copy** mode, a filename or link name that contains character encodings invalid in the destination hierarchy. (For example, the name may contain embedded NULs.)

27348

27349

- In **read** or **copy** mode, a filename or link name that is longer than the maximum allowed in the destination hierarchy (for either a pathname

27350	component or the entire pathname).				
27351	— In list mode, any character string value (filename, link name, user name, and so on) that cannot be written in the codeset and current locale of the implementation.				
27352					
27353					
27354	The following mutually-exclusive values of the <i>action</i> argument are supported:				
27355					
27356	bypass	In read or copy mode, <i>pax</i> shall bypass the file, causing no change to the destination hierarchy. In list mode, <i>pax</i> shall write all requested valid values for the file, but its method for writing invalid values is unspecified.			
27357					
27358					
27359					
27360	rename	In read or copy mode, <i>pax</i> shall act as if the -i option were in effect for each file with invalid filename or link name values, allowing the user to provide a replacement name interactively. In list mode, <i>pax</i> shall behave identically to the bypass action.			
27361					
27362					
27363					
27364	UTF-8	When used in read , copy , or list mode and a filename, link name, owner name, or any other field in an extended header record cannot be translated from the pax UTF-8 codeset format to the codeset and current locale of the implementation, <i>pax</i> shall use the actual UTF-8 encoding for the name.			
27365					
27366					
27367					
27368					
27369	write	In read or copy mode, <i>pax</i> shall write the file, translating the name, regardless of whether this may overwrite an existing file with a valid name. In list mode, <i>pax</i> shall behave identically to the bypass action.	2		
27370			2		
27371			2		
27372			2		
27373	If no -o invalid= option is specified, <i>pax</i> shall act as if -oinvalid=bypass were specified. Any overwriting of existing files that may be allowed by the -oinvalid= actions shall be subject to permission (-p) and modification time (-u) restrictions, and shall be suppressed if the -k option is also specified.				
27374					
27375					
27376					
27377	linkdata	(Applicable only to the -x pax format.) In write mode, <i>pax</i> shall write the contents of a file to the archive even when that file is merely a hard link to a file whose contents have already been written to the archive.			
27378					
27379					
27380					
27381	listopt=format	This keyword specifies the output format of the table of contents produced when the -v option is specified in list mode. See List Mode Format Specifications (on page 708). To avoid ambiguity, the listopt=format shall be the only or final keyword=value pair in a -o option-argument; all characters in the remainder of the option-argument shall be considered part of the format string. When multiple -olistopt=format options are specified, the format strings shall be considered a single, concatenated string, evaluated in command line order.			
27382					
27383					
27384					
27385					
27386					
27387					
27388					
27389					
27390	times	(Applicable only to the -x pax format.) When used in write or copy mode, <i>pax</i> shall include atime and mtime extended header records for each file. See pax Extended Header File Times (on page 716).	1		
27391					
27392					
27393					
27394	In addition to these keywords, if the -x pax format is specified, any of the keywords and values defined in pax Extended Header (on page 713), including implementation extensions, can be used in -o option-arguments, in either of two				
27395					
27396					

27397

modes:

27398

keyword=value

27399

When used in **write** or **copy** mode, these keyword/value pairs shall be included at the beginning of the archive as **typeflag g** global extended header records. When used in **read** or **list** mode, these keyword/value pairs shall act as if they had been at the beginning of the archive as **typeflag g** global extended header records.

27400

keyword:=value

27401

When used in **write** or **copy** mode, these keyword/value pairs shall be included as records at the beginning of a **typeflag x** extended header for each file. (This shall be equivalent to the equal-sign form except that it creates no **typeflag g** global extended header records.) When used in **read** or **list** mode, these keyword/value pairs shall act as if they were included as records at the end of each extended header; thus, they shall override any global or file-specific extended header record keywords of the same names. For example, in the command:

27413

```
pax -r -o "
gname:=mygroup,
" <archive
```

27416

the group name will be forced to a new value for all files read from the archive.

27418

The precedence of **-o** keywords over various fields in the archive is described in **pax Extended Header Keyword Precedence** (on page 716).

27420

-p string

27421

27422

27423

27424

27425

Specify one or more file characteristic options (privileges). The *string* option-argument shall be a string specifying file characteristics to be retained or discarded on extraction. The string shall consist of the specification characters **a**, **e**, **m**, **o**, and **p**. Other implementation-defined characters can be included. Multiple characteristics can be concatenated within the same string and multiple **-p** options can be specified. The meaning of the specification characters are as follows:

27426

a Do not preserve file access times.

27427

e Preserve the user ID, group ID, file mode bits (see the Base Definitions volume of IEEE Std 1003.1-2001, Section 3.168, File Mode Bits), access time, modification time, and any other implementation-defined file characteristics.

27428

m Do not preserve file modification times.

27429

o Preserve the user ID and group ID.

27430

p Preserve the file mode bits. Other implementation-defined file mode attributes may be preserved.

27431

In the preceding list, “preserve” indicates that an attribute stored in the archive shall be given to the extracted file, subject to the permissions of the invoking process. The access and modification times of the file shall be preserved unless otherwise specified with the **-p** option or not stored in the archive. All attributes that are not preserved shall be determined as part of the normal file creation action (see Section 1.7.1.4 (on page 4)).

27432

If neither the **e** nor the **o** specification character is specified, or the user ID and group ID are not preserved for any reason, *pax* shall not set the **S_ISUID** and **S_ISGID** bits of the file mode.

27443		If the preservation of any of these items fails for any reason, <i>pax</i> shall write a diagnostic message to standard error. Failure to preserve these items shall affect the final exit status, but shall not cause the extracted file to be deleted.
27444		
27445		
27446		If file characteristic letters in any of the <i>string</i> option-arguments are duplicated or conflict with each other, the ones given last shall take precedence. For example, if -p eme is specified, file modification times are preserved.
27447		
27448		
27449	-s replstr	Modify file or archive member names named by <i>pattern</i> or <i>file</i> operands according to the substitution expression <i>replstr</i> , using the syntax of the <i>ed</i> utility. The concepts of “address” and “line” are meaningless in the context of the <i>pax</i> utility, and shall not be supplied. The format shall be:
27450		
27451		
27452		
27453		-s /old/new/[gp]
27454		where as in <i>ed</i> , <i>old</i> is a basic regular expression and <i>new</i> can contain an ampersand, ‘\n’ (where <i>n</i> is a digit) backreferences, or subexpression matching. The <i>old</i> string shall also be permitted to contain <newline>s.
27455		
27456		
27457		Any non-null character can be used as a delimiter (‘/’ shown here). Multiple -s expressions can be specified; the expressions shall be applied in the order specified, terminating with the first successful substitution. The optional trailing ‘g’ is as defined in the <i>ed</i> utility. The optional trailing ‘p’ shall cause successful substitutions to be written to standard error. File or archive member names that substitute to the empty string shall be ignored when reading and writing archives.
27458		
27459		
27460		
27461		
27462		
27463	-t	When reading files from the file system, and if the user has the permissions required by <i>utime()</i> to do so, set the access time of each file read to the access time that it had before being read by <i>pax</i> .
27464		
27465		
27466	-u	Ignore files that are older (having a less recent file modification time) than a pre-existing file or archive member with the same name. In read mode, an archive member with the same name as a file in the file system shall be extracted if the archive member is newer than the file. In write mode, an archive file member with the same name as a file in the file system shall be superseded if the file is newer than the archive member. If -a is also specified, this is accomplished by appending to the archive; otherwise, it is unspecified whether this is accomplished by actual replacement in the archive or by appending to the archive. In copy mode, the file in the destination hierarchy shall be replaced by the file in the source hierarchy or by a link to the file in the source hierarchy if the file in the source hierarchy is newer.
27467		
27468		
27469		
27470		
27471		
27472		
27473		
27474		
27475		
27476	-v	In list mode, produce a verbose table of contents (see the STDOUT section). Otherwise, write archive member pathnames to standard error (see the STDERR section).
27477		
27478		
27479	-x format	Specify the output archive format. The <i>pax</i> utility shall support the following formats:
27480		
27481	cpio	The cpio interchange format; see the EXTENDED DESCRIPTION section. The default <i>blocksize</i> for this format for character special archive files shall be 5 120. Implementations shall support all <i>blocksize</i> values less than or equal to 32 256 that are multiples of 512.
27482		
27483		
27484		
27485	pax	The pax interchange format; see the EXTENDED DESCRIPTION section. The default <i>blocksize</i> for this format for character special archive files shall be 5 120. Implementations shall support all <i>blocksize</i> values less than or equal to 32 256 that are multiples of 512.
27486		
27487		
27488		

27489	ustar	The tar interchange format; see the EXTENDED DESCRIPTION section. The default <i>blocksize</i> for this format for character special archive files shall be 10 240. Implementations shall support all <i>blocksize</i> values less than or equal to 32 256 that are multiples of 512.	
27493		Implementation-defined formats shall specify a default block size as well as any other block sizes supported for character special archive files.	
27495		Any attempt to append to an archive file in a format different from the existing archive format shall cause <i>pax</i> to exit immediately with a non-zero exit status.	
27497		In copy mode, if no -x format is specified, <i>pax</i> shall behave as if -xpax were specified.	
27499	-X	When traversing the file hierarchy specified by a pathname, <i>pax</i> shall not descend into directories that have a different device ID (<i>st_dev</i> ; see the System Interfaces volume of IEEE Std 1003.1-2001, <i>stat()</i>).	
27502		Specifying more than one of the mutually-exclusive options -H and -L shall not be considered an error and the last option specified shall determine the behavior of the utility.	2
27504		The options that operate on the names of files or archive members (-c , -i , -n , -s , -u , and -v) shall interact as follows. In read mode, the archive members shall be selected based on the user-specified <i>pattern</i> operands as modified by the -c , -n , and -u options. Then, any -s and -i options shall modify, in that order, the names of the selected files. The -v option shall write names resulting from these modifications.	
27509		In write mode, the files shall be selected based on the user-specified pathnames as modified by the -n and -u options. Then, any -s and -i options shall modify, in that order, the names of these selected files. The -v option shall write names resulting from these modifications.	
27512		If both the -u and -n options are specified, <i>pax</i> shall not consider a file selected unless it is newer than the file to which it is compared.	
27514	List Mode Format Specifications		
27515		In list mode with the -o listopt=format option, the <i>format</i> argument shall be applied for each selected file. The <i>pax</i> utility shall append a <newline> to the listopt output for each selected file. The <i>format</i> argument shall be used as the <i>format</i> string described in the Base Definitions volume of IEEE Std 1003.1-2001, Chapter 5, File Format Notation, with the exceptions 1. through 5. defined in the EXTENDED DESCRIPTION section of <i>printf</i> , plus the following exceptions:	
27520	6.	The sequence (<i>keyword</i>) can occur before a format conversion specifier. The conversion argument is defined by the value of <i>keyword</i> . The implementation shall support the following keywords:	
27523		— Any of the Field Name entries in Table 4-13 (on page 717) and Table 4-15 (on page 720). The implementation may support the <i>cpio</i> keywords without the leading c_ in addition to the form required by Table 4-16 (on page 721).	
27526		— Any keyword defined for the extended header in pax Extended Header (on page 713).	
27527		— Any keyword provided as an implementation-defined extension within the extended header defined in pax Extended Header (on page 713).	
27529		For example, the sequence "%(charset)s" is the string value of the name of the character set in the extended header.	
27531		The result of the keyword conversion argument shall be the value from the applicable header field or extended header, without any trailing NULs.	

27533 All keyword values used as conversion arguments shall be translated from the UTF-8
27534 encoding to the character set appropriate for the local file system, user database, and so on,
27535 as applicable.

- 27536 7. An additional conversion specifier character, T, shall be used to specify time formats. The T
27537 conversion specifier character can be preceded by the sequence (*keyword*=*subformat*), where
27538 *subformat* is a date format as defined by *date* operands. The default *keyword* shall be **mtime**
27539 and the default subformat shall be:

27540 %b %e %H:%M %Y

- 27541 8. An additional conversion specifier character, M, shall be used to specify the file mode string
27542 as defined in *ls* Standard Output. If (*keyword*) is omitted, the **mode** keyword shall be used.
27543 For example, %.1M writes the single character corresponding to the <entry type> field of the
27544 *ls -l* command.
- 27545 9. An additional conversion specifier character, D, shall be used to specify the device for block
27546 or special files, if applicable, in an implementation-defined format. If not applicable, and
27547 (*keyword*) is specified, then this conversion shall be equivalent to %(*keyword*)u. If not
27548 applicable, and (*keyword*) is omitted, then this conversion shall be equivalent to <space>.

- 27549 10. An additional conversion specifier character, F, shall be used to specify a pathname. The F
27550 conversion character can be preceded by a sequence of comma-separated keywords:

27551 (*keyword*[,*keyword*] . . .)

27552 The values for all the keywords that are non-null shall be concatenated together, each
27553 separated by a '/'. The default shall be (**path**) if the keyword **path** is defined; otherwise,
27554 the default shall be (**prefix.name**).

- 27555 11. An additional conversion specifier character, L, shall be used to specify a symbolic line
27556 expansion. If the current file is a symbolic link, then %L shall expand to:

27557 "%s -> %s", <value of keyword>, <contents of link>

27558 Otherwise, the %L conversion specification shall be the equivalent of %F.

27559 OPERANDS

27560 The following operands shall be supported:

27561 *directory* The destination directory pathname for **copy** mode.

27562 *file* A pathname of a file to be copied or archived.

27563 *pattern* A pattern matching one or more pathnames of archive members. A pattern must
27564 be given in the name-generating notation of the pattern matching notation in
27565 Section 2.13 (on page 62), including the filename expansion rules in Section 2.13.3
27566 (on page 63). The default, if no *pattern* is specified, is to select all members in the
27567 archive.

27568 STDIN

27569 In **write** mode, the standard input shall be used only if no *file* operands are specified. It shall be a
27570 text file containing a list of pathnames, one per line, without leading or trailing <blank>s.

27571 In **list** and **read** modes, if **-f** is not specified, the standard input shall be an archive file.

27572 Otherwise, the standard input shall not be used.

27573 INPUT FILES

27574 The input file named by the *archive* option-argument, or standard input when the archive is read
27575 from there, shall be a file formatted according to one of the specifications in the EXTENDED
27576 DESCRIPTION section or some other implementation-defined format.

27577 The file */dev/tty* shall be used to write prompts and read responses.

27578 ENVIRONMENT VARIABLES

27579 The following environment variables shall affect the execution of *pax*:

27580 *LANG* Provide a default value for the internationalization variables that are unset or null.
27581 (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2,
27582 Internationalization Variables for the precedence of internationalization variables
27583 used to determine the values of locale categories.)

27584 *LC_ALL* If set to a non-empty string value, override the values of all the other
27585 internationalization variables.

27586 *LC_COLLATE*

27587 Determine the locale for the behavior of ranges, equivalence classes, and multi-
27588 character collating elements used in the pattern matching expressions for the
27589 *pattern* operand, the basic regular expression for the *-s* option, and the extended
27590 regular expression defined for the *yesexpr* locale keyword in the *LC_MESSAGES*
27591 category.

27592 *LC_CTYPE* Determine the locale for the interpretation of sequences of bytes of text data as
27593 characters (for example, single-byte as opposed to multi-byte characters in
27594 arguments and input files), the behavior of character classes used in the extended
27595 regular expression defined for the *yesexpr* locale keyword in the *LC_MESSAGES*
27596 category, and pattern matching.

27597 *LC_MESSAGES*

27598 Determine the locale for the processing of affirmative responses that should be
27599 used to affect the format and contents of diagnostic messages written to standard
27600 error.

27601 *LC_TIME* Determine the format and contents of date and time strings when the *-v* option is
27602 specified.

27603 XSI *NLSPATH* Determine the location of message catalogs for the processing of *LC_MESSAGES*.

27604 *TMPDIR* Determine the pathname that provides part of the default global extended header
27605 record file, as described for the *-o globexthdr=* keyword in the OPTIONS section.

27606 *TZ* Determine the timezone used to calculate date and time strings when the *-v* option
27607 is specified. If *TZ* is unset or null, an unspecified default timezone shall be used.

27608 ASYNCHRONOUS EVENTS

27609 Default.

27610 STDOUT

27611 In **write** mode, if *-f* is not specified, the standard output shall be the archive formatted
27612 according to one of the specifications in the EXTENDED DESCRIPTION section, or some other
27613 implementation-defined format (see *-x format*).

27614 In **list** mode, when the *-olistopt=format* has been specified, the selected archive members shall
27615 be written to standard output using the format described under **List Mode Format**
27616 **Specifications** (on page 708). In **list** mode without the *-olistopt=format* option, the table of
27617 contents of the selected archive members shall be written to standard output using the following

27618 format:

27619 "`%s\n`", *<pathname>*

27620 If the **-v** option is specified in **list** mode, the table of contents of the selected archive members
27621 shall be written to standard output using the following formats.

27622 For pathnames representing hard links to previous members of the archive:

27623 "`%sΔ==Δ%s\n`", *<ls -l listing>*, *<linkname>*

27624 For all other pathnames:

27625 "`%s\n`", *<ls -l listing>*

27626 where *<ls -l listing>* shall be the format specified by the *ls* utility with the **-l** option. When
27627 writing pathnames in this format, it is unspecified what is written for fields for which the
27628 underlying archive format does not have the correct information, although the correct number of
27629 `<blank>`-separated fields shall be written.

27630 In **list** mode, standard output shall not be buffered more than a line at a time.

27631 **STDERR**

27632 If **-v** is specified in **read**, **write**, or **copy** modes, *pax* shall write the pathnames it processes to the
27633 standard error output using the following format:

27634 "`%s\n`", *<pathname>*

27635 These pathnames shall be written as soon as processing is begun on the file or archive member,
27636 and shall be flushed to standard error. The trailing *<newline>*, which shall not be buffered, is
27637 written when the file has been read or written.

27638 If the **-s** option is specified, and the replacement string has a trailing '`p`', substitutions shall be
27639 written to standard error in the following format:

27640 "`%sΔ>>Δ%s\n`", *<original pathname>*, *<new pathname>*

27641 In all operating modes of *pax*, optional messages of unspecified format concerning the input
27642 archive format and volume number, the number of files, blocks, volumes, and media parts as
27643 well as other diagnostic messages may be written to standard error.

27644 In all formats, for both standard output and standard error, it is unspecified how non-printable
27645 characters in pathnames or link names are written.

27646 When *pax* is in **read** mode or **list** mode, using the **-xpax** archive format, and a filename, link
27647 name, owner name, or any other field in an extended header record cannot be translated from
27648 the **pax** UTF-8 codeset format to the codeset and current locale of the implementation, *pax* shall
27649 write a diagnostic message to standard error, shall process the file as described for the **-o**
27650 **invalid**=option, and then shall process the next file in the archive.

27651 **OUTPUT FILES**

27652 In **read** mode, the extracted output files shall be of the archived file type. In **copy** mode, the
27653 copied output files shall be the type of the file being copied. In either mode, existing files in the
27654 destination hierarchy shall be overwritten only when all permission (**-p**), modification time (**-u**),
27655 and invalid-value (**-oinvalid=**) tests allow it.

27656 In **write** mode, the output file named by the **-f** option-argument shall be a file formatted
27657 according to one of the specifications in the EXTENDED DESCRIPTION section, or some other
27658 implementation-defined format.

27659 EXTENDED DESCRIPTION

27660 pax Interchange Format

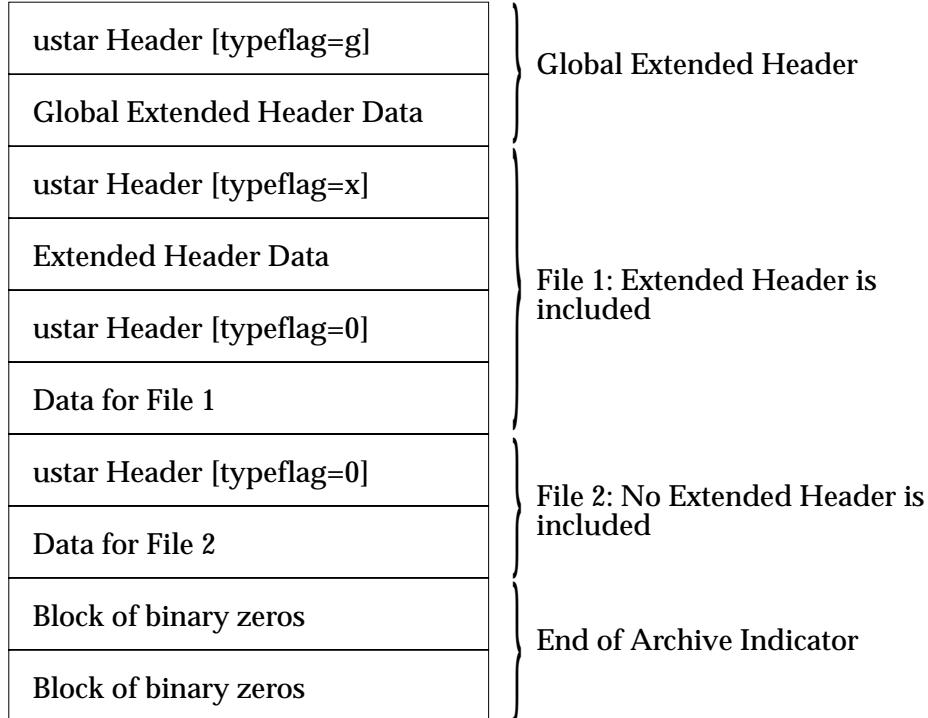
27661 A **pax** archive tape or file produced in the **-xpax** format shall contain a series of blocks. The
 27662 physical layout of the archive shall be identical to the **ustar** format described in **ustar**
 27663 **Interchange Format** (on page 717). Each file archived shall be represented by the following
 27664 sequence:

- 27665 • An optional header block with extended header records. This header block is of the form
 27666 described in **pax Header Block** (on page 713), with a *typeflag* value of **x** or **g**. The extended
 27667 header records, described in **pax Extended Header** (on page 713), shall be included as the
 27668 data for this header block.
- 27669 • A header block that describes the file. Any fields in the preceding optional extended header
 27670 shall override the associated fields in this header block for this file.
- 27671 • Zero or more blocks that contain the contents of the file.

27672 At the end of the archive file there shall be two 512-byte blocks filled with binary zeros,
 27673 interpreted as an end-of-archive indicator.

27674 A schematic of an example archive with global extended header records and two actual files is
 27675 shown in Figure 4-1. In the example, the second file in the archive has no extended header
 27676 preceding it, presumably because it has no need for extended attributes.

27677



27678

Figure 4-1 pax Format Archive Example

27679

pax Header Block

27680

The **pax** header block shall be identical to the **ustar** header block described in **ustar Interchange Format** (on page 717), except that two additional *typeflag* values are defined:

27682

x Represents extended header records for the following file in the archive (which shall have its own **ustar** header block). The format of these extended header records shall be as described in **pax Extended Header**.

27685

g Represents global extended header records for the following files in the archive. The format of these extended header records shall be as described in **pax Extended Header**. Each value shall affect all subsequent files that do not override that value in their own extended header record and until another global extended header record is reached that provides another value for the same field. The *typeflag g* global headers should not be used with interchange media that could suffer partial data loss in transporting the archive.

27691

For both of these types, the *size* field shall be the size of the extended header records in octets. The other fields in the header block are not meaningful to this version of the **pax** utility. However, if this archive is read by a **pax** utility conforming to the ISO POSIX-2:1993 standard, the header block fields are used to create a regular file that contains the extended header records as data. Therefore, header block field values should be selected to provide reasonable file access to this regular file.

27697

A further difference from the **ustar** header block is that data blocks for files of *typeflag 1* (the digit one) (hard link) may be included, which means that the *size* field may be greater than zero. Archives created by **pax -o linkdata** shall include these data blocks with the hard links.

27700

pax Extended Header

27701

A **pax** extended header contains values that are inappropriate for the **ustar** header block because of limitations in that format: fields requiring a character encoding other than that described in the ISO/IEC 646: 1991 standard, fields representing file attributes not described in the **ustar** header, and fields whose format or length do not fit the requirements of the **ustar** header. The values in an extended header add attributes to the following file (or files; see the description of the *typeflag g* header block) or override values in the following header block(s), as indicated in the following list of keywords.

27708

An extended header shall consist of one or more records, each constructed as follows:

27709

"%d %s=%s\n", <length>, <keyword>, <value>

27710

The extended header records shall be encoded according to the ISO/IEC 10646-1: 2000 standard (UTF-8). The *<length>* field, <blank>, equals sign, and <newline> shown shall be limited to the portable character set, as encoded in UTF-8. The *<keyword>* and *<value>* fields can be any UTF-8 characters. The *<length>* field shall be the decimal length of the extended header record in octets, including the trailing <newline>.

27715

The *<keyword>* field shall be one of the entries from the following list or a keyword provided as an implementation extension. Keywords consisting entirely of lowercase letters, digits, and periods are reserved for future standardization. A keyword shall not include an equals sign. (In the following list, the notations "file(s)" or "block(s)" is used to acknowledge that a keyword affects the following single file after a *typeflag x* extended header, but possibly multiple files after *typeflag g*. Any requirements in the list for **pax** to include a record when in **write** or **copy** mode shall apply only when such a record has not already been provided through the use of the **-o** option. When used in **copy** mode, **pax** shall behave as if an archive had been created with applicable extended header records and then extracted.)

27724 **atime** The file access time for the following file(s), equivalent to the value of the *st_atime* member of the **stat** structure for a file, as described by the *stat()* function. The access time shall be restored if the process has the appropriate privilege required to do so. The format of the <value> shall be as described in **pax Extended Header File Times** (on page 716).

27729 **charset** The name of the character set used to encode the data in the following file(s). The entries in the following table are defined to refer to known standards; additional names may be agreed on between the originator and recipient.

<value>	Formal Standard
ISO-IRΔ646Δ1990	ISO/IEC 646: 1990
ISO-IRΔ8859Δ1Δ1998	ISO/IEC 8859-1: 1998
ISO-IRΔ8859Δ2Δ1999	ISO/IEC 8859-2: 1999
ISO-IRΔ8859Δ3Δ1999	ISO/IEC 8859-3: 1999
ISO-IRΔ8859Δ4Δ1998	ISO/IEC 8859-4: 1998
ISO-IRΔ8859Δ5Δ1999	ISO/IEC 8859-5: 1999
ISO-IRΔ8859Δ6Δ1999	ISO/IEC 8859-6: 1999
ISO-IRΔ8859Δ7Δ1987	ISO/IEC 8859-7: 1987
ISO-IRΔ8859Δ8Δ1999	ISO/IEC 8859-8: 1999
ISO-IRΔ8859Δ9Δ1999	ISO/IEC 8859-9: 1999
ISO-IRΔ8859Δ10Δ1998	ISO/IEC 8859-10: 1998
ISO-IRΔ8859Δ13Δ1998	ISO/IEC 8859-13: 1998
ISO-IRΔ8859Δ14Δ1998	ISO/IEC 8859-14: 1998
ISO-IRΔ8859Δ15Δ1999	ISO/IEC 8859-15: 1999
ISO-IRΔ10646Δ2000	ISO/IEC 10646: 2000
ISO-IRΔ10646Δ2000ΔUTF-8	ISO/IEC 10646, UTF-8 encoding
BINARY	None.

27750 The encoding is included in an extended header for information only; when *pax* is used as described in IEEE Std 1003.1-2001, it shall not translate the file data into any other encoding. The **BINARY** entry indicates unencoded binary data.

27753 When used in **write** or **copy** mode, it is implementation-defined whether *pax* includes a **charset** extended header record for a file.

27755 **comment** A series of characters used as a comment. All characters in the <value> field shall be ignored by *pax*.

2

27757 **gid** The group ID of the group that owns the file, expressed as a decimal number using digits from the ISO/IEC 646: 1991 standard. This record shall override the *gid* field in the following header block(s). When used in **write** or **copy** mode, *pax* shall include a *gid* extended header record for each file whose group ID is greater than 2 097 151 (octal 7 777 777).

27762 **gname** The group of the file(s), formatted as a group name in the group database. This record shall override the *gid* and *gname* fields in the following header block(s), and any *gid* extended header record. When used in **read**, **copy**, or **list** mode, *pax* shall translate the name from the UTF-8 encoding in the header record to the character set appropriate for the group database on the receiving system. If any of the UTF-8 characters cannot be translated, and if the **-oinvalid=UTF-8** option is not specified, the results are implementation-defined. When used in **write** or **copy** mode, *pax* shall include a **gname** extended header record for each file whose group name cannot be represented entirely with the letters and digits of the portable character set.

27772	linkpath	The pathname of a link being created to another file, of any type, previously archived. This record shall override the <i>linkname</i> field in the following ustar header block(s). The following ustar header block shall determine the type of link created. If <i>typeflag</i> of the following header block is 1, it shall be a hard link. If <i>typeflag</i> is 2, it shall be a symbolic link and the linkpath value shall be the contents of the symbolic link. The <i>pax</i> utility shall translate the name of the link (contents of the symbolic link) from the UTF-8 encoding to the character set appropriate for the local file system. When used in write or copy mode, <i>pax</i> shall include a linkpath extended header record for each link whose pathname cannot be represented entirely with the members of the portable character set other than NUL.
27782	mtime	The file modification time of the following file(s), equivalent to the value of the <i>st_mtime</i> member of the stat structure for a file, as described in the <i>stat()</i> function. This record shall override the <i>mtime</i> field in the following header block(s). The modification time shall be restored if the process has the appropriate privilege required to do so. The format of the <value> shall be as described in pax Extended Header File Times (on page 716).
27788	path	The pathname of the following file(s). This record shall override the <i>name</i> and <i>prefix</i> fields in the following header block(s). The <i>pax</i> utility shall translate the pathname of the file from the UTF-8 encoding to the character set appropriate for the local file system.
27792		When used in write or copy mode, <i>pax</i> shall include a path extended header record for each file whose pathname cannot be represented entirely with the members of the portable character set other than NUL.
27795	realtime.any	The keywords prefixed by “realtime.” are reserved for future standardization.
27796	security.any	The keywords prefixed by “security.” are reserved for future standardization.
27797	size	The size of the file in octets, expressed as a decimal number using digits from the ISO/IEC 646: 1991 standard. This record shall override the <i>size</i> field in the following header block(s). When used in write or copy mode, <i>pax</i> shall include a size extended header record for each file with a size value greater than 8 589 934 591 (octal 77 777 777 777).
27802	uid	The user ID of the file owner, expressed as a decimal number using digits from the ISO/IEC 646: 1991 standard. This record shall override the <i>uid</i> field in the following header block(s). When used in write or copy mode, <i>pax</i> shall include a uid extended header record for each file whose owner ID is greater than 2 097 151 (octal 7 777 777).
27807	uname	The owner of the following file(s), formatted as a user name in the user database. This record shall override the <i>uid</i> and <i>uname</i> fields in the following header block(s), and any <i>uid</i> extended header record. When used in read , copy , or list mode, <i>pax</i> shall translate the name from the UTF-8 encoding in the header record to the character set appropriate for the user database on the receiving system. If any of the UTF-8 characters cannot be translated, and if the -oinvalid= UTF-8 option is not specified, the results are implementation-defined. When used in write or copy mode, <i>pax</i> shall include a uname extended header record for each file whose user name cannot be represented entirely with the letters and digits of the portable character set.
27817		If the <value> field is zero length, it shall delete any header block field, previously entered extended header value, or global extended header value of the same name.

27819 If a keyword in an extended header record (or in a **-o** option-argument) overrides or deletes a
27820 corresponding field in the **ustar** header block, **pax** shall ignore the contents of that header block
27821 field.

27822 Unlike the **ustar** header block fields, NULs shall not delimit **<value>**s; all characters within the
27823 **<value>** field shall be considered data for the field. None of the length limitations of the **ustar**
27824 header block fields in Table 4-13 (on page 717) shall apply to the extended header records.

27825 **pax Extended Header Keyword Precedence**

27826 This section describes the precedence in which the various header records and fields and
27827 command line options are selected to apply to a file in the archive. When **pax** is used in **read** or
27828 **list** modes, it shall determine a file attribute in the following sequence:

- 27829 1. If **-odelete=keyword-prefix** is used, the affected attributes shall be determined from step 7.,
27830 if applicable, or ignored otherwise.
- 27831 2. If **-okeyword:=** is used, the affected attributes shall be ignored.
- 27832 3. If **-okeyword:=value** is used, the affected attribute shall be assigned the value.
- 27833 4. If there is a **typeflag x** extended header record, the affected attribute shall be assigned the
27834 **<value>**. When extended header records conflict, the last one given in the header shall take
27835 precedence.
- 27836 5. If **-okeyword=value** is used, the affected attribute shall be assigned the value.
- 27837 6. If there is a **typeflag g** global extended header record, the affected attribute shall be
27838 assigned the **<value>**. When global extended header records conflict, the last one given in
27839 the global header shall take precedence.
- 27840 7. Otherwise, the attribute shall be determined from the **ustar** header block.

27841 **pax Extended Header File Times**

27842 The **pax** utility shall write an **mtime** record for each file in **write** or **copy** modes if the file's
27843 modification time cannot be represented exactly in the **ustar** header logical record described in
27844 **ustar Interchange Format** (on page 717). This can occur if the time is out of **ustar** range, or if the
27845 file system of the underlying implementation supports non-integer time granularities and the
27846 time is not an integer. All of these time records shall be formatted as a decimal representation of
27847 the time in seconds since the Epoch. If a period ('.') decimal point character is present, the
27848 digits to the right of the point shall represent the units of a subsecond timing granularity, where
27849 the first digit is tenths of a second and each subsequent digit is a tenth of the previous digit. In
27850 **read** or **copy** mode, the **pax** utility shall truncate the time of a file to the greatest value that is not
27851 greater than the input header file time. In **write** or **copy** mode, the **pax** utility shall output a time
27852 exactly if it can be represented exactly as a decimal number, and otherwise shall generate only
27853 enough digits so that the same time shall be recovered if the file is extracted on a system whose
27854 underlying implementation supports the same time granularity.

27855

ustar Interchange Format

27856

A **ustar** archive tape or file shall contain a series of logical records. Each logical record shall be a fixed-size logical record of 512 octets (see below). Although this format may be thought of as being stored on 9-track industry-standard 12.7 mm (0.5 in) magnetic tape, other types of transportable media are not excluded. Each file archived shall be represented by a header logical record that describes the file, followed by zero or more logical records that give the contents of the file. At the end of the archive file there shall be two 512-octet logical records filled with binary zeros, interpreted as an end-of-archive indicator.

27863

The logical records may be grouped for physical I/O operations, as described under the **-b blocksize** and **-x ustar** options. Each group of logical records may be written with a single operation equivalent to the **write()** function. On magnetic tape, the result of this write shall be a single tape physical block. The last physical block shall always be the full size, so logical records after the two zero logical records may contain undefined data.

27868

The header logical record shall be structured as shown in the following table. All lengths and offsets are in decimal.

27870

Table 4-13 ustar Header Block

Field Name	Octet Offset	Length (in Octets)
<i>name</i>	0	100
<i>mode</i>	100	8
<i>uid</i>	108	8
<i>gid</i>	116	8
<i>size</i>	124	12
<i>mtime</i>	136	12
<i>checksum</i>	148	8
<i>typeflag</i>	156	1
<i>linkname</i>	157	100
<i>magic</i>	257	6
<i>version</i>	263	2
<i>uname</i>	265	32
<i>gname</i>	297	32
<i>devmajor</i>	329	8
<i>devminor</i>	337	8
<i>prefix</i>	345	155

27888

All characters in the header logical record shall be represented in the coded character set of the ISO/IEC 646:1991 standard. For maximum portability between implementations, names should be selected from characters represented by the portable filename character set as octets with the most significant bit zero. If an implementation supports the use of characters outside of slash and the portable filename character set in names for files, users, and groups, one or more implementation-defined encodings of these characters shall be provided for interchange purposes.

27895

However, the **pax** utility shall never create filenames on the local system that cannot be accessed via the procedures described in IEEE Std 1003.1-2001. If a filename is found on the medium that would create an invalid filename, it is implementation-defined whether the data from the file is stored on the file hierarchy and under what name it is stored. The **pax** utility may choose to ignore these files as long as it produces an error indicating that the file is being ignored.

27900

Each field within the header logical record is contiguous; that is, there is no padding used. Each character on the archive medium shall be stored contiguously.

The fields *magic*, *uname*, and *gname* are character strings each terminated by a NUL character. The fields *name*, *linkname*, and *prefix* are NUL-terminated character strings except when all characters in the array contain non-NUL characters including the last character. The *version* field is two octets containing the characters "00" (zero-zero). The *typeflag* contains a single character. All other fields are leading zero-filled octal numbers using digits from the ISO/IEC 646:1991 standard IRV. Each numeric field is terminated by one or more <space> or NUL characters.

The *name* and the *prefix* fields shall produce the pathname of the file. A new pathname shall be formed, if *prefix* is not an empty string (its first character is not NUL), by concatenating *prefix* (up to the first NUL character), a slash character, and *name*; otherwise, *name* is used alone. In either case, *name* is terminated at the first NUL character. If *prefix* begins with a NUL character, it shall be ignored. In this manner, pathnames of at most 256 characters can be supported. If a pathname does not fit in the space provided, *pax* shall notify the user of the error, and shall not store any part of the file—header or data—on the medium.

The *linkname* field, described below, shall not use the *prefix* to produce a pathname. As such, a *linkname* is limited to 100 characters. If the name does not fit in the space provided, *pax* shall notify the user of the error, and shall not attempt to store the link on the medium.

The *mode* field provides 12 bits encoded in the ISO/IEC 646:1991 standard octal digit representation. The encoded bits shall represent the following values:

Table 4-14 ustar mode Field

Bit Value	IEEE Std 1003.1-2001 Bit	Description
04 000	S_ISUID	Set UID on execution.
02 000	S_ISGID	Set GID on execution.
01 000	<reserved>	Reserved for future standardization.
00 400	S_IRUSR	Read permission for file owner class.
00 200	S_IWUSR	Write permission for file owner class.
00 100	S_IXUSR	Execute/search permission for file owner class.
00 040	S_IRGRP	Read permission for file group class.
00 020	S_IWGRP	Write permission for file group class.
00 010	S_IXGRP	Execute/search permission for file group class.
00 004	S_IROTH	Read permission for file other class.
00 002	S_IWOTH	Write permission for file other class.
00 001	S_IXOTH	Execute/search permission for file other class.

When appropriate privilege is required to set one of these mode bits, and the user restoring the files from the archive does not have the appropriate privilege, the mode bits for which the user does not have appropriate privilege shall be ignored. Some of the mode bits in the archive format are not mentioned elsewhere in this volume of IEEE Std 1003.1-2001. If the implementation does not support those bits, they may be ignored.

The *uid* and *gid* fields are the user and group ID of the owner and group of the file, respectively.

The *size* field is the size of the file in octets. If the *typeflag* field is set to specify a file to be of type 1 (a link) or 2 (a symbolic link), the *size* field shall be specified as zero. If the *typeflag* field is set to specify a file of type 5 (directory), the *size* field shall be interpreted as described under the definition of that record type. No data logical records are stored for types 1, 2, or 5. If the *typeflag* field is set to 3 (character special file), 4 (block special file), or 6 (FIFO), the meaning of the *size* field is unspecified by this volume of IEEE Std 1003.1-2001, and no data logical records shall be stored on the medium. Additionally, for type 6, the *size* field shall be ignored when reading. If the *typeflag* field is set to any other value, the number of logical records written following the header shall be $(\text{size}+511)/512$, ignoring any fraction in the result of the division.

27949 The *mtime* field shall be the modification time of the file at the time it was archived. It is the
 27950 ISO/IEC 646: 1991 standard representation of the octal value of the modification time obtained
 27951 from the *stat()* function.

27952 The *cksum* field shall be the ISO/IEC 646: 1991 standard IRV representation of the octal value of
 27953 the simple sum of all octets in the header logical record. Each octet in the header shall be treated
 27954 as an unsigned value. These values shall be added to an unsigned integer, initialized to zero, the
 27955 precision of which is not less than 17 bits. When calculating the checksum, the *cksum* field is
 27956 treated as if it were all spaces.

27957 The *typeflag* field specifies the type of file archived. If a particular implementation does not
 27958 recognize the type, or the user does not have appropriate privilege to create that type, the file
 27959 shall be extracted as if it were a regular file if the file type is defined to have a meaning for the
 27960 *size* field that could cause data logical records to be written on the medium (see the previous
 27961 description for *size*). If conversion to a regular file occurs, the *pax* utility shall produce an error
 27962 indicating that the conversion took place. All of the *typeflag* fields shall be coded in the
 27963 ISO/IEC 646: 1991 standard IRV:

27964 0	Represents a regular file. For backwards-compatibility, a <i>typeflag</i> value of binary zero ('\'0') should be recognized as meaning a regular file when extracting files from the archive. Archives written with this version of the archive file format create regular files with a <i>typeflag</i> value of the ISO/IEC 646: 1991 standard IRV '0'.	2
27968 1	Represents a file linked to another file, of any type, previously archived. Such files are identified by having the same device and file serial numbers, and pathnames that refer to different directory entries. All such files shall be archived as linked files. The linked- to name is specified in the <i>linkname</i> field with a NUL-character terminator if it is less than 100 octets in length.	2
27973 2	Represents a symbolic link. The contents of the symbolic link shall be stored in the <i>linkname</i> field.	2
27975 3 , 4	Represent character special files and block special files respectively. In this case the <i>devmajor</i> and <i>devminor</i> fields shall contain information defining the device, the format of which is unspecified by this volume of IEEE Std 1003.1-2001. Implementations may map the device specifications to their own local specification or may ignore the entry.	2
27979 5	Specifies a directory or subdirectory. On systems where disk allocation is performed on a directory basis, the <i>size</i> field shall contain the maximum number of octets (which may be rounded to the nearest disk block allocation unit) that the directory may hold. A <i>size</i> field of zero indicates no such limiting. Systems that do not support limiting in this manner should ignore the <i>size</i> field.	2
27984 6	Specifies a FIFO special file. Note that the archiving of a FIFO file archives the existence of this file and not its contents.	2
27986 7	Reserved to represent a file to which an implementation has associated some high- performance attribute. Implementations without such extensions should treat this file as a regular file (type 0).	2
27989 A-Z	The letters 'A' to 'Z', inclusive, are reserved for custom implementations. All other values are reserved for future versions of IEEE Std 1003.1-2001.	2
27991 It is unspecified whether files with pathnames that refer to the same directory entry are archived as linked files or as separate files. If they are archived as linked files, this means that attempting to extract both pathnames from the resulting archive will always cause an error (unless the -u option is used) because the link cannot be created.	2	

27995 It is unspecified whether files with the same device and file serial numbers being appended to an archive are treated as linked files to members that were in the archive before the append. 2
 27996

27997 Attempts to archive a socket using **ustar** interchange format shall produce a diagnostic message.
 27998 Handling of other file types is implementation-defined.

27999 The *magic* field is the specification that this archive was output in this archive format. If this field
 28000 contains **ustar** (the five characters from the ISO/IEC 646: 1991 standard IRV shown followed by
 28001 NUL), the *uname* and *gname* fields shall contain the ISO/IEC 646: 1991 standard IRV
 28002 representation of the owner and group of the file, respectively (truncated to fit, if necessary).
 28003 When the file is restored by a privileged, protection-preserving version of the utility, the user
 28004 and group databases shall be scanned for these names. If found, the user and group IDs
 28005 contained within these files shall be used rather than the values contained within the *uid* and *gid*
 28006 fields.

28007 cpio Interchange Format

28008 The octet-oriented **cpio** archive format shall be a series of entries, each comprising a header that
 28009 describes the file, the name of the file, and then the contents of the file.

28010 An archive may be recorded as a series of fixed-size blocks of octets. This blocking shall be used
 28011 only to make physical I/O more efficient. The last group of blocks shall always be at the full
 28012 size.

28013 For the octet-oriented **cpio** archive format, the individual entry information shall be in the order
 28014 indicated and described by the following table; see also the <**cpio.h**> header.

28015 **Table 4-15** Octet-Oriented cpio Archive Entry

28016	Header Field Name	Length (in Octets)	Interpreted as
28017	<i>c_magic</i>	6	Octal number
28018	<i>c_dev</i>	6	Octal number
28019	<i>c_ino</i>	6	Octal number
28020	<i>c_mode</i>	6	Octal number
28021	<i>c_uid</i>	6	Octal number
28022	<i>c_gid</i>	6	Octal number
28023	<i>c_nlink</i>	6	Octal number
28024	<i>c_rdev</i>	6	Octal number
28025	<i>c_mtime</i>	11	Octal number
28026	<i>c_namesize</i>	6	Octal number
28027	<i>c_filesize</i>	11	Octal number
28028	Filename Field Name	Length	Interpreted as
28029	<i>c_name</i>	<i>c_namesize</i>	Pathname string
28030	File Data Field Name	Length	Interpreted as
28031	<i>c_filedata</i>	<i>c_filesize</i>	Data

28032

cpio Header

28033

For each file in the archive, a header as defined previously shall be written. The information in the header fields is written as streams of the ISO/IEC 646: 1991 standard characters interpreted as octal numbers. The octal numbers shall be extended to the necessary length by appending the ISO/IEC 646: 1991 standard IRV zeros at the most-significant-digit end of the number; the result is written to the most-significant digit of the stream of octets first. The fields shall be interpreted as follows:

28034

c_magic Identify the archive as being a transportable archive by containing the identifying value "070707".

28035

c_dev, *c_ino* Contains values that uniquely identify the file within the archive (that is, no files contain the same pair of *c_dev* and *c_ino* values unless they are links to the same file). The values shall be determined in an unspecified manner.

28036

c_mode Contains the file type and access permissions as defined in the following table.

28037

28038

28039

28040

28041

28042

28043

28044

28045

Table 4-16 Values for cpio *c_mode* Field

File Permissions Name	Value	Indicates
C_IRUSR	000 400	Read by owner
C_IWUSR	000 200	Write by owner
C_IXUSR	000 100	Execute by owner
C_IRGRP	000 040	Read by group
C_IWGRP	000 020	Write by group
C_IXGRP	000 010	Execute by group
C_IROTH	000 004	Read by others
C_IWOTH	000 002	Write by others
C_IXOTH	000 001	Execute by others
C_ISUID	004 000	Set <i>uid</i>
C_ISGID	002 000	Set <i>gid</i>
C_ISVTX	001 000	Reserved
File Type Name	Value	Indicates
C_ISDIR	040 000	Directory
C_ISFIFO	010 000	FIFO
C_ISREG	0100 000	Regular file
C_ISLNK	0120 000	Symbolic link
C_ISBLK	060 000	Block special file
C_ISCHR	020 000	Character special file
C_ISSOCK	0140 000	Socket
C_ISCTG	0110 000	Reserved

28059

28060

28061

28062

28063

28064

28065

28066

28067

Directories, FIFOs, symbolic links, and regular files shall be supported on a system conforming to this volume of IEEE Std 1003.1-2001; additional values defined previously are reserved for compatibility with existing systems. Additional file types may be supported; however, such files should not be written to archives intended to be transported to other systems.

28073

c_uid Contains the user ID of the owner.

28074

c_gid Contains the group ID of the group.

28075

c_nlink Contains a number greater than or equal to the number of links in the archive referencing the file. If the *-a* option is used to append to a *cpio* archive, then the *pax* 2

28076

28077		utility need not account for the files in the existing part of the archive when calculating the <i>c_nlink</i> values for the appended part of the archive, and need not alter the <i>c_nlink</i> values in the existing part of the archive if additional files with the same <i>c_dev</i> and <i>c_ino</i> values are appended to the archive.	2
28078			2
28079			2
28080			2
28081	<i>c_rdev</i>	Contains implementation-defined information for character or block special files.	
28082	<i>c_mtime</i>	Contains the latest time of modification of the file at the time the archive was created.	
28083			
28084	<i>c_namesize</i>	Contains the length of the pathname, including the terminating NUL character.	
28085	<i>c_filesize</i>	Contains the length of the file in octets. This shall be the length of the data section following the header structure.	
28086			

28087 cpio Filename

28088 The *c_name* field shall contain the pathname of the file. The length of this field in octets is the
28089 value of *c_namesize*.

28090 If a filename is found on the medium that would create an invalid pathname, it is
28091 implementation-defined whether the data from the file is stored on the file hierarchy and under
28092 what name it is stored.

28093 All characters shall be represented in the ISO/IEC 646:1991 standard IRV. For maximum
28094 portability between implementations, names should be selected from characters represented by
28095 the portable filename character set as octets with the most significant bit zero. If an
28096 implementation supports the use of characters outside the portable filename character set in
28097 names for files, users, and groups, one or more implementation-defined encodings of these
28098 characters shall be provided for interchange purposes. However, the *pax* utility shall never
28099 create filenames on the local system that cannot be accessed via the procedures described
28100 previously in this volume of IEEE Std 1003.1-2001. If a filename is found on the medium that
28101 would create an invalid filename, it is implementation-defined whether the data from the file is
28102 stored on the local file system and under what name it is stored. The *pax* utility may choose to
28103 ignore these files as long as it produces an error indicating that the file is being ignored.

28104 cpio File Data

28105 Following *c_name*, there shall be *c_filesize* octets of data. Interpretation of such data occurs in a
28106 manner dependent on the file. If *c_filesize* is zero, no data shall be contained in *c_filedata*.

28107 When restoring from an archive:

- 28108 • If the user does not have the appropriate privilege to create a file of the specified type, *pax*
28109 shall ignore the entry and write an error message to standard error.
- 28110 • Only regular files have data to be restored. Presuming a regular file meets any selection
28111 criteria that might be imposed on the format-reading utility by the user, such data shall be
28112 restored.
- 28113 • If a user does not have appropriate privilege to set a particular mode flag, the flag shall be
28114 ignored. Some of the mode flags in the archive format are not mentioned elsewhere in this
28115 volume of IEEE Std 1003.1-2001. If the implementation does not support those flags, they
28116 may be ignored.

28117	cpio Special Entries	
28118	FIFO special files, directories, and the trailer shall be recorded with <i>c_filesize</i> equal to zero. For	
28119	other special files, <i>c_filesize</i> is unspecified by this volume of IEEE Std 1003.1-2001. The header for	
28120	the next file entry in the archive shall be written directly after the last octet of the file entry	
28121	preceding it. A header denoting the filename TRAILER!!! shall indicate the end of the archive;	
28122	the contents of octets in the last block of the archive following such a header are undefined.	
28123	EXIT STATUS	
28124	The following exit values shall be returned:	
28125	0 All files were processed successfully.	
28126	>0 An error occurred.	
28127	CONSEQUENCES OF ERRORS	
28128	If <i>pax</i> cannot create a file or a link when reading an archive or cannot find a file when writing an	
28129	archive, or cannot preserve the user ID, group ID, or file mode when the -p option is specified, a	
28130	diagnostic message shall be written to standard error and a non-zero exit status shall be	
28131	returned, but processing shall continue. In the case where <i>pax</i> cannot create a link to a file, <i>pax</i>	
28132	shall not, by default, create a second copy of the file.	
28133	If the extraction of a file from an archive is prematurely terminated by a signal or error, <i>pax</i> may	
28134	have only partially extracted the file or (if the -n option was not specified) may have extracted a	
28135	file of the same name as that specified by the user, but which is not the file the user wanted.	
28136	Additionally, the file modes of extracted directories may have additional bits from the S_IRWXU	
28137	mask set as well as incorrect modification and access times.	
28138	APPLICATION USAGE	
28139	Caution is advised when using the -a option to append to a <i>cpio</i> format archive. If any of the	2
28140	files being appended happen to be given the same <i>c_dev</i> and <i>c_ino</i> values as a file in the existing	2
28141	part of the archive, then they may be treated as links to that file on extraction. Thus, it is risky to	2
28142	use -a with <i>cpio</i> format except when it is done on the same system that the original archive was	2
28143	created on, and with the same <i>pax</i> utility, and in the knowledge that there has been little or no	2
28144	file system activity since the original archive was created that could lead to any of the files	2
28145	appended being given the same <i>c_dev</i> and <i>c_ino</i> values as an unrelated file in the existing part of	2
28146	the archive. Also, when (intentionally) appending additional links to a file in the existing part of	2
28147	the archive, the <i>c_nlink</i> values in the modified archive can be smaller than the number of links to	2
28148	the file in the archive, which may mean that the links are not preserved on extraction.	2
28149	The -p (privileges) option was invented to reconcile differences between historical <i>tar</i> and <i>cpio</i>	
28150	implementations. In particular, the two utilities use -m in diametrically opposed ways. The -p	
28151	option also provides a consistent means of extending the ways in which future file attributes can	
28152	be addressed, such as for enhanced security systems or high-performance files. Although it may	
28153	seem complex, there are really two modes that are most commonly used:	
28154	-p e “Preserve everything”. This would be used by the historical superuser, someone with	
28155	all the appropriate privileges, to preserve all aspects of the files as they are recorded in	
28156	the archive. The e flag is the sum of o and p , and other implementation-defined	
28157	attributes.	
28158	-p p “Preserve” the file mode bits. This would be used by the user with regular privileges	
28159	who wished to preserve aspects of the file other than the ownership. The file times are	
28160	preserved by default, but two other flags are offered to disable these and use the time	
28161	of extraction.	
28162	The one pathname per line format of standard input precludes pathnames containing	
28163	<newline>s. Although such pathnames violate the portable filename guidelines, they may exist	

28164 and their presence may inhibit usage of *pax* within shell scripts. This problem is inherited from
28165 historical archive programs. The problem can be avoided by listing filename arguments on the
28166 command line instead of on standard input.

28167 It is almost certain that appropriate privileges are required for *pax* to accomplish parts of this
28168 volume of IEEE Std 1003.1-2001. Specifically, creating files of type block special or character
28169 special, restoring file access times unless the files are owned by the user (the **-t** option), or
28170 preserving file owner, group, and mode (the **-p** option) all probably require appropriate
28171 privileges.

28172 In **read** mode, implementations are permitted to overwrite files when the archive has multiple
28173 members with the same name. This may fail if permissions on the first version of the file do not
28174 permit it to be overwritten.

28175 The **cpio** and **ustar** formats can only support files up to 8 589 934 592 bytes (8 * 2³⁰) in size.

28176 EXAMPLES

28177 The following command:

```
28178 pax -w -f /dev/rmt/1m .
```

28179 copies the contents of the current directory to tape drive 1, medium density (assuming historical
28180 System V device naming procedures—the historical BSD device name would be **/dev/rmt9**).

28181 The following commands:

```
28182 mkdir newdir  
28183 pax -rw olddir newdir
```

28184 copy the *olddir* directory hierarchy to *newdir*.

```
28185 pax -r -s ',^///*usr//*', -f a.pax
```

28186 reads the archive **a.pax**, with all files rooted in **/usr** in the archive extracted relative to the current
28187 directory.

28188 Using the option:

```
28189 -o listopt="%M %(atime)T %(size)D %(name)s"
```

28190 overrides the default output description in Standard Output and instead writes:

```
28191 -rw-rw--- Jan 12 15:53 1492 /usr/foo/bar
```

28192 Using the options:

```
28193 -o listopt='%L\t%(size)D\n%.7' \  
28194 -o listopt='%(name)s\n%(atime)T\n%T'
```

2

28195 overrides the default output description in Standard Output and instead writes:

```
28196 /usr/foo/bar -> /tmp 1492
```

```
28197 /usr/fo
```

```
28198 Jan 12 1991
```

```
28199 Jan 31 15:53
```

28200 RATIONALE

28201 The *pax* utility was new for the ISO POSIX-2:1993 standard. It represents a peaceful compromise
28202 between advocates of the historical *tar* and *cpio* utilities.

28203 A fundamental difference between *cpio* and *tar* was in the way directories were treated. The *cpio*
28204 utility did not treat directories differently from other files, and to select a directory and its
28205 contents required that each file in the hierarchy be explicitly specified. For *tar*, a directory

28206 matched every file in the file hierarchy it rooted.

28207 The *pax* utility offers both interfaces; by default, directories map into the file hierarchy they root.
28208 The **-d** option causes *pax* to skip any file not explicitly referenced, as *cpio* historically did. The *tar*
28209 *-style* behavior was chosen as the default because it was believed that this was the more
28210 common usage and because *tar* is the more commonly available interface, as it was historically
28211 provided on both System V and BSD implementations.

28212 The data interchange format specification in this volume of IEEE Std 1003.1-2001 requires that
28213 processes with “appropriate privileges” shall always restore the ownership and permissions of
28214 extracted files exactly as archived. If viewed from the historic equivalence between superuser
28215 and “appropriate privileges”, there are two problems with this requirement. First, users running
28216 as superusers may unknowingly set dangerous permissions on extracted files. Second, it is
28217 needlessly limiting, in that superusers cannot extract files and own them as superuser unless the
28218 archive was created by the superuser. (It should be noted that restoration of ownerships and
28219 permissions for the superuser, by default, is historical practice in *cpio*, but not in *tar*.) In order to
28220 avoid these two problems, the *pax* specification has an additional “privilege” mechanism, the **-p**
28221 option. Only a *pax* invocation with the privileges needed, and which has the **-p** option set using
28222 the **e** specification character, has the “appropriate privilege” to restore full ownership and
28223 permission information.

28224 Note also that this volume of IEEE Std 1003.1-2001 requires that the file ownership and access
28225 permissions shall be set, on extraction, in the same fashion as the *creat()* function when provided
28226 with the mode stored in the archive. This means that the file creation mask of the user is applied
28227 to the file permissions.

28228 Users should note that directories may be created by *pax* while extracting files with permissions
28229 that are different from those that existed at the time the archive was created. When extracting
28230 sensitive information into a directory hierarchy that no longer exists, users are encouraged to set
28231 their file creation mask appropriately to protect these files during extraction.

28232 The table of contents output is written to standard output to facilitate pipeline processing.

28233 An early proposal had hard links displaying for all pathnames. This was removed because it
28234 complicates the output of the case where **-v** is not specified and does not match historical *cpio*
28235 usage. The hard-link information is available in the **-v** display.

28236 The description of the **-l** option allows implementations to make hard links to symbolic links.
28237 IEEE Std 1003.1-2001 does not specify any way to create a hard link to a symbolic link, but many
28238 implementations provide this capability as an extension. If there are hard links to symbolic links
28239 when an archive is created, the implementation is required to archive the hard link in the archive
28240 (unless **-H** or **-L** is specified). When in **read** mode and in **copy** mode, implementations
28241 supporting hard links to symbolic links should use them when appropriate.

28242 The archive formats inherited from the POSIX.1-1990 standard have certain restrictions that
28243 have been brought along from historical usage. For example, there are restrictions on the length
28244 of pathnames stored in the archive. When *pax* is used in **copy(-rw)** mode (copying directory
28245 hierarchies), the ability to use extensions from the **-xpax** format overcomes these restrictions.

28246 The default *blocksize* value of 5 120 bytes for *cpio* was selected because it is one of the standard
28247 block-size values for *cpio*, set when the **-B** option is specified. (The other default block-size value
28248 for *cpio* is 512 bytes, and this was considered to be too small.) The default block value of 10 240
28249 bytes for *tar* was selected because that is the standard block-size value for BSD *tar*. The
28250 maximum block size of 32 256 bytes (2^{15} –512 bytes) is the largest multiple of 512 bytes that fits
28251 into a signed 16-bit tape controller transfer register. There are known limitations in some
28252 historical systems that would prevent larger blocks from being accepted. Historical values were
28253 chosen to improve compatibility with historical scripts using *dd* or similar utilities to manipulate

archives. Also, default block sizes for any file type other than character special file has been deleted from this volume of IEEE Std 1003.1-2001 as unimportant and not likely to affect the structure of the resulting archive.

Implementations are permitted to modify the block-size value based on the archive format or the device to which the archive is being written. This is to provide implementations with the opportunity to take advantage of special types of devices, and it should not be used without a great deal of consideration as it almost certainly decreases archive portability.

The intended use of the **-n** option was to permit extraction of one or more files from the archive without processing the entire archive. This was viewed by the standard developers as offering significant performance advantages over historical implementations. The **-n** option in early proposals had three effects; the first was to cause special characters in patterns to not be treated specially. The second was to cause only the first file that matched a pattern to be extracted. The third was to cause *pax* to write a diagnostic message to standard error when no file was found matching a specified pattern. Only the second behavior is retained by this volume of IEEE Std 1003.1-2001, for many reasons. First, it is in general not acceptable for a single option to have multiple effects. Second, the ability to make pattern matching characters act as normal characters is useful for parts of *pax* other than file extraction. Third, a finer degree of control over the special characters is useful because users may wish to normalize only a single special character in a single filename. Fourth, given a more general escape mechanism, the previous behavior of the **-n** option can be easily obtained using the **-s** option or a *sed* script. Finally, writing a diagnostic message when a pattern specified by the user is unmatched by any file is useful behavior in all cases.

In this version, the **-n** was removed from the **copy** mode synopsis of *pax*; it is inapplicable because there are no pattern operands specified in this mode.

There is another method than *pax* for copying subtrees in IEEE Std 1003.1-2001 described as part of the *cp* utility. Both methods are historical practice: *cp* provides a simpler, more intuitive interface, while *pax* offers a finer granularity of control. Each provides additional functionality to the other; in particular, *pax* maintains the hard-link structure of the hierarchy while *cp* does not. It is the intention of the standard developers that the results be similar (using appropriate option combinations in both utilities). The results are not required to be identical; there seemed insufficient gain to applications to balance the difficulty of implementations having to guarantee that the results would be exactly identical.

A single archive may span more than one file. It is suggested that implementations provide informative messages to the user on standard error whenever the archive file is changed.

The **-d** option (do not create intermediate directories not listed in the archive) found in early proposals was originally provided as a complement to the historic **-d** option of *cpio*. It has been deleted.

The **-s** option in early proposals specified a subset of the substitution command from the *ed* utility. As there was no reason for only a subset to be supported, the **-s** option is now compatible with the current *ed* specification. Since the delimiter can be any non-null character, the following usage with single spaces is valid:

```
28295 pax -s " foo bar " ...
```

The **-t** description is worded so as to note that this may cause the access time update caused by some other activity (which occurs while the file is being read) to be overwritten.

The default behavior of *pax* with regard to file modification times is the same as historical implementations of *tar*. It is not the historical behavior of *cpio*.

- 28300 Because the **-i** option uses **/dev/tty**, utilities without a controlling terminal are not able to use
28301 this option.
- 28302 The **-y** option, found in early proposals, has been deleted because a line containing a single
28303 period for the **-i** option has equivalent functionality. The special lines for the **-i** option (a single
28304 period and the empty line) are historical practice in **cpio**.
- 28305 In early drafts, a **-echarmap** option was included to increase portability of files between systems
28306 using different coded character sets. This option was omitted because it was apparent that
28307 consensus could not be formed for it. In this version, the use of UTF-8 should be an adequate
28308 substitute.
- 28309 The **-k** option was added to address international concerns about the dangers involved in the
28310 character set transformations of **-e** (if the target character set were different from the source, the
28311 filenames might be transformed into names matching existing files) and also was made more
28312 general to protect files transferred between file systems with different {NAME_MAX} values
28313 (truncating a filename on a smaller system might also inadvertently overwrite existing files). As
28314 stated, it prevents any overwriting, even if the target file is older than the source. This version
28315 adds more granularity of options to solve this problem by introducing the **-oinvalid=** option—
28316 specifically the UTF-8 action. (Note that an existing file that is named with a UTF-8 encoding is
28317 still subject to overwriting in this case. The **-k** option closes that loophole.)
- 28318 Some of the file characteristics referenced in this volume of IEEE Std 1003.1-2001 might not be
28319 supported by some archive formats. For example, neither the **tar** nor **cpio** formats contain the
28320 file access time. For this reason, the **e** specification character has been provided, intended to
28321 cause all file characteristics specified in the archive to be retained.
- 28322 It is required that extracted directories, by default, have their access and modification times and
28323 permissions set to the values specified in the archive. This has obvious problems in that the
28324 directories are almost certainly modified after being extracted and that directory permissions
28325 may not permit file creation. One possible solution is to create directories with the mode
28326 specified in the archive, as modified by the **umask** of the user, with sufficient permissions to
28327 allow file creation. After all files have been extracted, **pax** would then reset the access and
28328 modification times and permissions as necessary.
- 28329 The list-mode formatting description borrows heavily from the one defined by the **printf** utility.
28330 However, since there is no separate operand list to get conversion arguments, the format was
28331 extended to allow specifying the name of the conversion argument as part of the conversion
28332 specification.
- 28333 The **T** conversion specifier allows time fields to be displayed in any of the date formats. Unlike
28334 the **ls** utility, **pax** does not adjust the format when the date is less than six months in the past.
28335 This makes parsing the output more predictable.
- 28336 The **D** conversion specifier handles the ability to display the major/minor or file size, as with **ls**,
28337 by using **%-8(size)D**.
- 28338 The **L** conversion specifier handles the **ls** display for symbolic links.
- 28339 Conversion specifiers were added to generate existing known types used for **ls**.

28340 **pax Interchange Format**

28341 The new POSIX data interchange format was developed primarily to satisfy international
28342 concerns that the **ustar** and **cpio** formats did not provide for file, user, and group names encoded
28343 in characters outside a subset of the ISO/IEC 646:1991 standard. The standard developers
28344 realized that this new POSIX data interchange format should be very extensible because there
28345 were other requirements they foresaw in the near future:

- 28346 • Support international character encodings and locale information
- 28347 • Support security information (ACLs, and so on)
- 28348 • Support future file types, such as realtime or contiguous files
- 28349 • Include data areas for implementation use
- 28350 • Support systems with words larger than 32 bits and timers with subsecond granularity

28351 The following were not goals for this format because these are better handled by separate
28352 utilities or are inappropriate for a portable format:

- 28353 • Encryption
- 28354 • Compression
- 28355 • Data translation between locales and codesets
- 28356 • *inode* storage

28357 The format chosen to support the goals is an extension of the **ustar** format. Of the two formats
28358 previously available, only the **ustar** format was selected for extensions because:

- 28359 • It was easier to extend in an upwards-compatible way. It offered version flags and header
28360 block type fields with room for future standardization. The **cpio** format, while possessing a
28361 more flexible file naming methodology, could not be extended without breaking some
28362 theoretical implementation or using a dummy filename that could be a legitimate filename.
- 28363 • Industry experience since the original “tar wars” fought in developing the ISO POSIX-1
28364 standard has clearly been in favor of the **ustar** format, which is generally the default output
28365 format selected for *pax* implementations on new systems.

28366 The new format was designed with one additional goal in mind: reasonable behavior when an
28367 older *tar* or *pax* utility happened to read an archive. Since the POSIX.1-1990 standard mandated
28368 that a “format-reading utility” had to treat unrecognized *typeflag* values as regular files, this
28369 allowed the format to include all the extended information in a pseudo-regular file that preceded
28370 each real file. An option is given that allows the archive creator to set up reasonable names for
28371 these files on the older systems. Also, the normative text suggests that reasonable file access
28372 values be used for this **ustar** header block. Making these header files inaccessible for convenient
28373 reading and deleting would not be reasonable. File permissions of 600 or 700 are suggested.

28374 The **ustar** *typeflag* field was used to accommodate the additional functionality of the new format
28375 rather than magic or version because the POSIX.1-1990 standard (and, by reference, the previous
28376 version of *pax*), mandated the behavior of the format-reading utility when it encountered an
28377 unknown *typeflag*, but was silent about the other two fields.

28378 Early proposals of the first revision to IEEE Std 1003.1-2001 contained a proposed archive format
28379 that was based on compatibility with the standard for tape files (ISO 1001, similar to the format
28380 used historically on many mainframes and minicomputers). This format was overly complex
28381 and required considerable overhead in volume and header records. Furthermore, the standard
28382 developers felt that it would not be acceptable to the community of POSIX developers, so it was
28383 later changed to be a format more closely related to historical practice on POSIX systems.

28384 The prefix and name split of pathnames in **ustar** was replaced by the single path extended
28385 header record for simplicity.

28386 The concept of a global extended header (*typeflagg*) was controversial. If this were applied to an
28387 archive being recorded on magnetic tape, a few unreadable blocks at the beginning of the tape
28388 could be a serious problem; a utility attempting to extract as many files as possible from a
28389 damaged archive could lose a large percentage of file header information in this case. However,
28390 if the archive were on a reliable medium, such as a CD-ROM, the global extended header offers
28391 considerable potential size reductions by eliminating redundant information. Thus, the text
28392 warns against using the global method for unreliable media and provides a method for
28393 implanting global information in the extended header for each file, rather than in the *typeflag g*
28394 records.

28395 No facility for data translation or filtering on a per-file basis is included because the standard
28396 developers could not invent an interface that would allow this in an efficient manner. If a filter,
28397 such as encryption or compression, is to be applied to all the files, it is more efficient to apply the
28398 filter to the entire archive as a single file. The standard developers considered interfaces that
28399 would invoke a shell script for each file going into or out of the archive, but the system overhead
28400 in this approach was considered to be too high.

28401 One such approach would be to have **filter=** records that give a pathname for an executable.
28402 When the program is invoked, the file and archive would be open for standard input/output
28403 and all the header fields would be available as environment variables or command-line
28404 arguments. The standard developers did discuss such schemes, but they were omitted from
28405 IEEE Std 1003.1-2001 due to concerns about excessive overhead. Also, the program itself would
28406 need to be in the archive if it were to be used portably.

28407 There is currently no portable means of identifying the character set(s) used for a file in the file
28408 system. Therefore, **pax** has not been given a mechanism to generate charset records
28409 automatically. The only portable means of doing this is for the user to write the archive using the
28410 **-ocharset=string** command line option. This assumes that all of the files in the archive use the
28411 same encoding. The “implementation-defined” text is included to allow for a system that can
28412 identify the encodings used for each of its files.

28413 The table of standards that accompanies the charset record description is acknowledged to be
28414 very limited. Only a limited number of character set standards is reasonable for maximal
28415 interchange. Any character set is, of course, possible by prior agreement. It was suggested that
28416 EBCDIC be listed, but it was omitted because it is not defined by a formal standard. Formal
28417 standards, and then only those with reasonably large followings, can be included here, simply as
28418 a matter of practicality. The *<value>*s represent names of officially registered character sets in the
28419 format required by the ISO 2375: 1985 standard.

28420 The normal comma or *<blank>*-separated list rules are not followed in the case of keyword
28421 options to allow ease of argument parsing for *getopts*.

28422 Further information on character encodings is in **pax Archive Character Set Encoding/Decoding**
28423 (on page 731).

28424 The standard developers have reserved keyword name space for vendor extensions. It is
28425 suggested that the format to be used is:

28426 *VENDOR.keyword*

28427 where **VENDOR** is the name of the vendor or organization in all uppercase letters. It is further
28428 suggested that the keyword following the period be named differently than any of the standard
28429 keywords so that it could be used for future standardization, if appropriate, by omitting the
28430 **VENDOR** prefix.

28431 The <length> field in the extended header record was included to make it simpler to step
28432 through the records, even if a record contains an unknown format (to a particular *pax*) with
28433 complex interactions of special characters. It also provides a minor integrity checkpoint within
28434 the records to aid a program attempting to recover files from a damaged archive.

28435 There are no extended header versions of the *devmajor* and *devminor* fields because the
28436 unspecified format **ustar** header field should be sufficient. If they are not, vendor-specific
28437 extended keywords (such as *VENDOR.devmajor*) should be used.

28438 Device and *i*-number labeling of files was not adopted from *cpio*; files are interchanged strictly
28439 on a symbolic name basis, as in **ustar**.

28440 Just as with the **ustar** format descriptions, the new format makes no special arrangements for
28441 multi-volume archives. Each of the *pax* archive types is assumed to be inside a single POSIX file
28442 and splitting that file over multiple volumes (diskettes, tape cartridges, and so on), processing
28443 their labels, and mounting each in the proper sequence are considered to be implementation
28444 details that cannot be described portably.

28445 The **pax** format is intended for interchange, not only for backup on a single (family of) systems.
28446 It is not as densely packed as might be possible for backup:

- 28447 • It contains information as coded characters that could be coded in binary.
- 28448 • It identifies extended records with name fields that could be omitted in favor of a fixed-field
28449 layout.
- 28450 • It translates names into a portable character set and identifies locale-related information,
28451 both of which are probably unnecessary for backup.

28452 The requirements on restoring from an archive are slightly different from the historical wording,
28453 allowing for non-monolithic privilege to bring forward as much as possible. In particular,
28454 attributes such as "high performance file" might be broadly but not universally granted while
28455 set-user-ID or *chown()* might be much more restricted. There is no implication in
28456 IEEE Std 1003.1-2001 that the security information be honored after it is restored to the file
28457 hierarchy, in spite of what might be improperly inferred by the silence on that topic. That is a
28458 topic for another standard.

28459 Links are recorded in the fashion described here because a link can be to any file type. It is
28460 desirable in general to be able to restore part of an archive selectively and restore all of those
28461 files completely. If the data is not associated with each link, it is not possible to do this.
28462 However, the data associated with a file can be large, and when selective restoration is not
28463 needed, this can be a significant burden. The archive is structured so that files that have no
28464 associated data can always be restored by the name of any link name of any link, and the user
28465 may choose whether data is recorded with each instance of a file that contains data. The format
28466 permits mixing of both types of links in a single archive; this can be done for special needs, and
28467 *pax* is expected to interpret such archives on input properly, despite the fact that there is no *pax*
28468 option that would force this mixed case on output. (When **-o linkdata** is used, the output must
28469 contain the duplicate data, but the implementation is free to include it or omit it when **-o**
28470 **linkdata** is not used.)

28471 The time values are included as extended header records for those implementations needing
28472 more than the eleven octal digits allowed by the **ustar** format. Portable file timestamps cannot be
28473 negative. If *pax* encounters a file with a negative timestamp in **copy** or **write** mode, it can reject
28474 the file, substitute a non-negative timestamp, or generate a non-portable timestamp with a
28475 leading '-'. Even though some implementations can support finer file-time granularities than
28476 seconds, the normative text requires support only for seconds since the Epoch because the
28477 ISO POSIX-1 standard states them that way. The **ustar** format includes only *mtime*; the new

format adds *atime* and *ctime* for symmetry. The *atime* access time restored to the file system will be affected by the **-p a** and **-p e** options. The *ctime* creation time (actually *inode* modification time) is described with “appropriate privilege” so that it can be ignored when writing to the file system. POSIX does not provide a portable means to change file creation time. Nothing is intended to prevent a non-portable implementation of *pax* from restoring the value.

The *gid*, *size*, and *uid* extended header records were included to allow expansion beyond the sizes specified in the regular *tar* header. New file system architectures are emerging that will exhaust the 12-digit size field. There are probably not many systems requiring more than 8 digits for user and group IDs, but the extended header values were included for completeness, allowing overrides for all of the decimal values in the *tar* header.

The standard developers intended to describe the effective results of *pax* with regard to file ownerships and permissions; implementations are not restricted in timing or sequencing the restoration of such, provided the results are as specified.

Much of the text describing the extended headers refers to use in “**write or copy modes**”. The **copy** mode references are due to the normative text: “The effect of the copy shall be as if the copied files were written to an archive file and then subsequently extracted ...”. There is certainly no way to test whether *pax* is actually generating the extended headers in **copy** mode, but the effects must be as if it had.

28496 **pax Archive Character Set Encoding/Decoding**

There is a need to exchange archives of files between systems of different native codesets. Filenames, group names, and user names must be preserved to the fullest extent possible when an archive is read on the receiving platform. Translation of the contents of files is not within the scope of the *pax* utility.

There will also be the need to represent characters that are not available on the receiving platform. These unsupported characters cannot be automatically folded to the local set of characters due to the chance of collisions. This could result in overwriting previous extracted files from the archive or pre-existing files on the system.

For these reasons, the codeset used to represent characters within the extended header records of the *pax* archive must be sufficiently rich to handle all commonly used character sets. The fields requiring translation include, at a minimum, filenames, user names, group names, and link pathnames. Implementations may wish to have localized extended keywords that use non-portable characters.

28510 The standard developers considered the following options:

- 28511 • The archive creator specifies the well-defined name of the source codeset. The receiver must
28512 then recognize the codeset name and perform the appropriate translations to the destination
28513 codeset.
- 28514 • The archive creator includes within the archive the character mapping table for the source
28515 codeset used to encode extended header records. The receiver must then read the character
28516 mapping table and perform the appropriate translations to the destination codeset.
- 28517 • The archive creator translates the extended header records in the source codeset into a
28518 canonical form. The receiver must then perform the appropriate translations to the
28519 destination codeset.

28520 The approach that incorporates the name of the source codeset poses the problem of codeset
28521 name registration, and makes the archive useless to *pax* archive decoders that do not recognize
28522 that codeset.

Because parts of an archive may be corrupted, the standard developers felt that including the character map of the source codeset was too fragile. The loss of this one key component could result in making the entire archive useless. (The difference between this and the global extended header decision was that the latter has a workaround—duplicating extended header records on unreliable media—but this would be too burdensome for large character set maps.)

Both of the above approaches also put an undue burden on the *pax* archive receiver to handle the cross-product of all source and destination codesets.

To simplify the translation from the source codeset to the canonical form and from the canonical form to the destination codeset, the standard developers decided that the internal representation should be a stateless encoding. A stateless encoding is one where each codepoint has the same meaning, without regard to the decoder being in a specific state. An example of a stateful encoding would be the Japanese Shift-JIS; an example of a stateless encoding would be the ISO/IEC 646: 1991 standard (equivalent to 7-bit ASCII).

For these reasons, the standard developers decided to adopt a canonical format for the representation of file information strings. The obvious, well-endorsed candidate is the ISO/IEC 10646-1: 2000 standard (based in part on Unicode), which can be used to represent the characters of virtually all standardized character sets. The standard developers initially agreed upon using UCS2 (16-bit Unicode) as the internal representation. This repertoire of characters provides a sufficiently rich set to represent all commonly-used codesets.

However, the standard developers found that the 16-bit Unicode representation had some problems. It forced the issue of standardizing byte ordering. The 2-byte length of each character made the extended header records twice as long for the case of strings coded entirely from historical 7-bit ASCII. For these reasons, the standard developers chose the UTF-8 defined in the ISO/IEC 10646-1: 2000 standard. This multi-byte representation encodes UCS2 or UCS4 characters reliably and deterministically, eliminating the need for a canonical byte ordering. In addition, NUL octets and other characters possibly confusing to POSIX file systems do not appear, except to represent themselves. It was realized that certain national codesets take up more space after the encoding, due to their placement within the UCS range; it was felt that the usefulness of the encoding of the names outweighs the disadvantage of size increase for file, user, and group names.

The encoding of UTF-8 is as follows:

28554	UCS4 Hex Encoding	UTF-8 Binary Encoding
28555	00000000-0000007F	0xxxxxxxxx
28556	00000080-000007FF	110xxxxx 10xxxxxx
28557	00000800-0000FFFF	1110xxxx 10xxxxxx 10xxxxxx
28558	00010000-001FFFFF	11110xxx 10xxxxxx 10xxxxxx 10xxxxxx
28559	00200000-03FFFFFF	111110xx 10xxxxxx 10xxxxxx 10xxxxxx 10xxxxxx
28560	04000000-7FFFFFFF	1111110x 10xxxxxx 10xxxxxx 10xxxxxx 10xxxxxx 10xxxxxx

where each 'x' represents a bit value from the character being translated.

28562

ustar Interchange Format

28563

The description of the **ustar** format reflects numerous enhancements over pre-1988 versions of the historical **tar** utility. The goal of these changes was not only to provide the functional enhancements desired, but also to retain compatibility between new and old versions. This compatibility has been retained. Archives written using the old archive format are compatible with the new format.

28568

Implementors should be aware that the previous file format did not include a mechanism to archive directory type files. For this reason, the convention of using a filename ending with slash was adopted to specify a directory on the archive.

28571

The total size of the *name* and *prefix* fields have been set to meet the minimum requirements for {PATH_MAX}. If a pathname will fit within the *name* field, it is recommended that the pathname be stored there without the use of the *prefix* field. Although the name field is known to be too small to contain {PATH_MAX} characters, the value was not changed in this version of the archive file format to retain backwards-compatibility, and instead the prefix was introduced. Also, because of the earlier version of the format, there is no way to remove the restriction on the *linkname* field being limited in size to just that of the *name* field.

28578

The *size* field is required to be meaningful in all implementation extensions, although it could be zero. This is required so that the data blocks can always be properly counted.

28580

It is suggested that if device special files need to be represented that cannot be represented in the standard format, that one of the extension types (A-Z) be used, and that the additional information for the special file be represented as data and be reflected in the *size* field.

28583

Attempting to restore a special file type, where it is converted to ordinary data and conflicts with an existing filename, need not be specially detected by the utility. If run as an ordinary user, **pax** should not be able to overwrite the entries in, for example, **/dev** in any case (whether the file is converted to another type or not). If run as a privileged user, it should be able to do so, and it would be considered a bug if it did not. The same is true of ordinary data files and similarly named special files; it is impossible to anticipate the needs of the user (who could really intend to overwrite the file), so the behavior should be predictable (and thus regular) and rely on the protection system as required.

28591

The value 7 in the *typeflag* field is intended to define how contiguous files can be stored in a **ustar** archive. IEEE Std 1003.1-2001 does not require the contiguous file extension, but does define a standard way of archiving such files so that all conforming systems can interpret these file types in a meaningful and consistent manner. On a system that does not support extended file types, the **pax** utility should do the best it can with the file and go on to the next.

28596

The file protection modes are those conventionally used by the **ls** utility. This is extended beyond the usage in the ISO POSIX-2 standard to support the “shared text” or “sticky” bit. It is intended that the conformance document should not document anything beyond the existence of and support of such a mode. Further extensions are expected to these bits, particularly with overloading the set-user-ID and set-group-ID flags.

28601 **cpio Interchange Format**

28602 The reference to appropriate privilege in the **cpio** format refers to an error on standard output;
28603 the **ustar** format does not make comparable statements.

28604 The model for this format was the historical System V *cpio-c* data interchange format. This
28605 model documents the portable version of the **cpio** format and not the binary version. It has the
28606 flexibility to transfer data of any type described within IEEE Std 1003.1-2001, yet is extensible to
28607 transfer data types specific to extensions beyond IEEE Std 1003.1-2001 (for example, contiguous
28608 files). Because it describes existing practice, there is no question of maintaining upwards-
28609 compatibility.

28610 **cpio Header**

28611 There has been some concern that the size of the *c_ino* field of the header is too small to handle
28612 those systems that have very large *inode* numbers. However, the *c_ino* field in the header is used
28613 strictly as a hard-link resolution mechanism for archives. It is not necessarily the same value as
28614 the *inode* number of the file in the location from which that file is extracted.

28615 The name *c_magic* is based on historical usage.

28616 **cpio Filename**

28617 For most historical implementations of the *cpio* utility, {PATH_MAX} octets can be used to
28618 describe the pathname without the addition of any other header fields (the NUL character
28619 would be included in this count). {PATH_MAX} is the minimum value for pathname size,
28620 documented as 256 bytes. However, an implementation may use *c_namesize* to determine the
28621 exact length of the pathname. With the current description of the <cpio.h> header, this
28622 pathname size can be as large as a number that is described in six octal digits.

28623 Two values are documented under the *c_mode* field values to provide for extensibility for known
28624 file types:

28625 **0110 000** Reserved for contiguous files. The implementation may treat the rest of the
28626 information for this archive like a regular file. If this file type is undefined, the
28627 implementation may create the file as a regular file.

28628 This provides for extensibility of the **cpio** format while allowing for the ability to read old
28629 archives. Files of an unknown type may be read as “regular files” on some implementations. On
28630 a system that does not support extended file types, the *pax* utility should do the best it can with
28631 the file and go on to the next.

28632 **FUTURE DIRECTIONS**

28633 None.

28634 **SEE ALSO**

28635 Chapter 2 (on page 29), *cp*, *ed*, *getopts*, *ls*, *printf*, the Base Definitions volume of
28636 IEEE Std 1003.1-2001, <cpio.h>, the System Interfaces volume of IEEE Std 1003.1-2001, *chown()*,
28637 *creat()*, *mkdir()*, *mkfifo()*, *stat()*, *utime()*, *write()*

28638 **CHANGE HISTORY**

28639 First released in Issue 4.

28640 **Issue 5**

28641 A note is added to the APPLICATION USAGE indicating that the **cpio** and **tar** formats can only
28642 support files up to 8 gigabytes in size.

28643 Issue 6

- 28644 The *pax* utility is aligned with the IEEE P1003.2b draft standard:
- 28645 • Support has been added for symbolic links in the options and interchange formats.
 - 28646 • A new format has been devised, based on extensions to **ustar**.
 - 28647 • References to the “extended” **tar** and **cpio** formats derived from the POSIX.1-1990 standard
28648 have been changed to remove the “extended” adjective because this could cause confusion
28649 with the extended *tar* header added in this revision. (All references to *tar* are actually to
28650 **ustar**.)
- 28651 The **TZ** entry is added to the ENVIRONMENT VARIABLES section.
- 28652 IEEE PASC Interpretation 1003.2 #168 is applied, clarifying that *mkdir()* and *mkfifo()* calls can
28653 ignore an [EEXIST] error when extracting an archive.
- 28654 IEEE PASC Interpretation 1003.2 #180 is applied, clarifying how extracted files are created when
28655 in **read** mode.
- 28656 IEEE PASC Interpretation 1003.2 #181 is applied, clarifying the description of the **-t** option.
- 28657 IEEE PASC Interpretation 1003.2 #195 is applied.
- 28658 IEEE PASC Interpretation 1003.2 #206 is applied, clarifying the handling of links for the **-H**, **-L**,
28659 and **-I** options.
- 28660 IEEE Std 1003.1-2001/Cor 1-2002, item XCU/TC1/D6/35 is applied, adding the process ID of 1
28661 the *pax* process into certain fields. This change provides a method for the implementation to 1
28662 ensure that different instances of *pax* extracting a file named **/a/b/foo** will not collide when 1
28663 processing the extended header information associated with **foo**. 1
- 28664 IEEE Std 1003.1-2001/Cor 1-2002, item XCU/TC1/D6/36 is applied, changing **-x B** to **-x pax** in 1
28665 the OPTIONS section. 1
- 28666 IEEE Std 1003.1-2001/Cor 2-2004, item XCU/TC2/D6/20 is applied, updating the SYNOPSIS to 2
28667 be consistent with the normative text. 2
- 28668 IEEE Std 1003.1-2001/Cor 2-2004, item XCU/TC2/D6/21 is applied, updating the 2
28669 DESCRIPTION to describe the behavior when files to be linked are symbolic links and the 2
28670 system is not capable of making hard links to symbolic links. 2
- 28671 IEEE Std 1003.1-2001/Cor 2-2004, item XCU/TC2/D6/22 is applied, updating the OPTIONS 2
28672 section to describe the behavior for how multiple **-odelete=pattern** options are to be handled. 2
- 28673 IEEE Std 1003.1-2001/Cor 2-2004, item XCU/TC2/D6/23 is applied, updating the **write** option 2
28674 within the OPTIONS section. 2
- 28675 IEEE Std 1003.1-2001/Cor 2-2004, item XCU/TC2/D6/24 is applied, adding a paragraph into the 2
28676 OPTIONS section that states that specifying more than one of the mutually-exclusive options 2
28677 (**-H** and **-L**) is not considered an error and that the last option specified will determine the 2
28678 behavior of the utility. 2
- 28679 IEEE Std 1003.1-2001/Cor 2-2004, item XCU/TC2/D6/25 is applied, removing the *ctime* 2
28680 paragraph within the EXTENDED DESCRIPTION. There is a contradiction in the definition of 2
28681 the *ctime* keyword for the *pax* extended header, in that the *st_ctime* member of the **stat** structure 2
28682 does not refer to a file creation time. No field in the standard **stat** structure from <sys/stat.h> 2
28683 includes a file creation time. 2
- 28684 IEEE Std 1003.1-2001/Cor 2-2004, item XCU/TC2/D6/26 is applied, making it clear that *typeflag* 2
28685 1 (**ustar** Interchange Format) applies not only to files that are hard-linked, but also to files that 2

28686	are aliased via symlinks.	2
28687	IEEE Std 1003.1-2001/Cor 2-2004, item XCU/TC2/D6/27 is applied, clarifying the <i>cpio c_nlink</i> field.	2
28688		2

28689 NAME

28690 pr — print files

28691 SYNOPSIS

```
28692     pr [+page][-column][-adFmrt][-e[char][gap]][-h header][-i[char][gap]]
28693 XSI      [-l lines][-n[char][width]][-o offset][-s[char]][-w width][-fp]
28694      [file...]
```

28695 DESCRIPTION

28696 The *pr* utility is a printing and pagination filter. If multiple input files are specified, each shall be
 28697 read, formatted, and written to standard output. By default, the input shall be separated into 66-line pages, each with:

- 28699 • A 5-line header that includes the page number, date, time, and the pathname of the file
- 28700 • A 5-line trailer consisting of blank lines

28701 If standard output is associated with a terminal, diagnostic messages shall be deferred until the
 28702 *pr* utility has completed processing.

28703 When options specifying multi-column output are specified, output text columns shall be of
 28704 equal width; input lines that do not fit into a text column shall be truncated. By default, text
 28705 columns shall be separated with at least one <blank>.

28706 OPTIONS

28707 The *pr* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section 12.2,
 28708 Utility Syntax Guidelines, except that: the *page* option has a '+' delimiter; *page* and *column* can
 28709 be multi-digit numbers; some of the option-arguments are optional; and some of the option-
 28710 arguments cannot be specified as separate arguments from the preceding option letter. In
 28711 particular, the *-s* option does not allow the option letter to be separated from its argument, and
 28712 the options *-e*, *-i*, and *-n* require that both arguments, if present, not be separated from the
 28713 option letter.

28714 The following options shall be supported. In the following option descriptions, *column*, *lines*,
 28715 *offset*, *page*, and *width* are positive decimal integers; *gap* is a non-negative decimal integer.

- 28716 **+page** Begin output at page number *page* of the formatted input.
- 28717 **-column** Produce multi-column output that is arranged in *column* columns (the default shall
 28718 be 1) and is written down each column in the order in which the text is received
 28719 from the input file. This option should not be used with *-m*. The options *-e* and *-i*
 28720 shall be assumed for multiple text-column output. Whether or not text columns
 28721 are produced with identical vertical lengths is unspecified, but a text column shall
 28722 never exceed the length of the page (see the *-l* option). When used with *-t*, use the
 28723 minimum number of lines to write the output.
- 28724 **-a** Modify the effect of the *-column* option so that the columns are filled across the
 28725 page in a round-robin order (for example, when *column* is 2, the first input line
 28726 heads column 1, the second heads column 2, the third is the second line in column
 28727 1, and so on).
- 28728 **-d** Produce output that is double-spaced; append an extra <newline> following every
 28729 <newline> found in the input.
- 28730 **-e[char][gap]** Expand each input <tab> to the next greater column position specified by the
 28731 formula $n * gap + 1$, where *n* is an integer > 0 . If *gap* is zero or is omitted, it shall
 28732 default to 8. All <tab>s in the input shall be expanded into the appropriate number
 28733 of <space>s. If any non-digit character, *char*, is specified, it shall be used as the
 28734

28735		input <tab>.
28736 XSI	-f	Use a <form-feed> for new pages, instead of the default behavior that uses a sequence of <newline>s. Pause before beginning the first page if the standard output is associated with a terminal.
28737		
28738		
28739	-F	Use a <form-feed> for new pages, instead of the default behavior that uses a sequence of <newline>s.
28740		
28741	-h header	Use the string <i>header</i> to replace the contents of the <i>file</i> operand in the page header.
28742	-i[char][gap]	In output, replace multiple <space>s with <tab>s wherever two or more adjacent <space>s reach column positions <i>gap</i> +1, 2* <i>gap</i> +1, 3* <i>gap</i> +1, and so on. If <i>gap</i> is zero or is omitted, default tab settings at every eighth column position shall be assumed. If any non-digit character, <i>char</i> , is specified, it shall be used as the output <tab>.
28743		
28744		
28745		
28746		
28747	-l lines	Override the 66-line default and reset the page length to <i>lines</i> . If <i>lines</i> is not greater than the sum of both the header and trailer depths (in lines), the <i>pr</i> utility shall suppress both the header and trailer, as if the -t option were in effect.
28748		
28749		
28750	-m	Merge files. Standard output shall be formatted so the <i>pr</i> utility writes one line from each file specified by a <i>file</i> operand, side by side into text columns of equal fixed widths, in terms of the number of column positions. Implementations shall support merging of at least nine <i>file</i> operands.
28751		
28752		
28753		
28754	-n[char][width]	Provide <i>width</i> -digit line numbering (default for <i>width</i> shall be 5). The number shall occupy the first <i>width</i> column positions of each text column of default output or each line of -m output. If <i>char</i> (any non-digit character) is given, it shall be appended to the line number to separate it from whatever follows (default for <i>char</i> is a <tab>).
28755		
28756		
28757		
28758		
28759		
28760	-o offset	Each line of output shall be preceded by offset <space>s. If the -o option is not specified, the default offset shall be zero. The space taken is in addition to the output line width (see the -w option below).
28761		
28762		
28763	-p	Pause before beginning each page if the standard output is directed to a terminal (<i>pr</i> shall write an <alert> to standard error and wait for a <carriage-return> to be read on /dev/tty).
28764		
28765		
28766	-r	Write no diagnostic reports on failure to open files.
28767	-s[char]	Separate text columns by the single character <i>char</i> instead of by the appropriate number of <space>s (default for <i>char</i> shall be <tab>).
28768		
28769	-t	Write neither the five-line identifying header nor the five-line trailer usually supplied for each page. Quit writing after the last line of each file without spacing to the end of the page.
28770		
28771		
28772	-w width	Set the width of the line to <i>width</i> column positions for multiple text-column output only. If the -w option is not specified and the -s option is not specified, the default width shall be 72. If the -w option is not specified and the -s option is specified, the default width shall be 512.
28773		
28774		
28775		
28776		For single column output, input lines shall not be truncated.

28777 OPERANDS

28778 The following operand shall be supported:

28779 *file* A pathname of a file to be written. If no *file* operands are specified, or if a *file* operand is ‘–’, the standard input shall be used.

28781 STDIN

28782 The standard input shall be used only if no *file* operands are specified, or if a *file* operand is ‘–’.
28783 See the INPUT FILES section.

28784 INPUT FILES

28785 The input files shall be text files.

28786 The file /dev/tty shall be used to read responses required by the –p option.

28787 ENVIRONMENT VARIABLES

28788 The following environment variables shall affect the execution of *pr*:28789 *LANG* Provide a default value for the internationalization variables that are unset or null.
28790 (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2,
28791 Internationalization Variables for the precedence of internationalization variables
28792 used to determine the values of locale categories.)28793 *LC_ALL* If set to a non-empty string value, override the values of all the other
28794 internationalization variables.28795 *LC_CTYPE* Determine the locale for the interpretation of sequences of bytes of text data as
28796 characters (for example, single-byte as opposed to multi-byte characters in
28797 arguments and input files) and which characters are defined as printable (character
28798 class **print**). Non-printable characters are still written to standard output, but are
28799 not counted for the purpose for column-width and line-length calculations.28800 *LC_MESSAGES*28801 Determine the locale that should be used to affect the format and contents of
28802 diagnostic messages written to standard error.28803 *LC_TIME* Determine the format of the date and time for use in writing header lines.28804 XSI *NLSPATH* Determine the location of message catalogs for the processing of *LC_MESSAGES*.28805 *TZ* Determine the timezone used to calculate date and time strings written in header
28806 lines. If *TZ* is unset or null, an unspecified default timezone shall be used.

28807 ASYNCHRONOUS EVENTS

28808 If *pr* receives an interrupt while writing to a terminal, it shall flush all accumulated error
28809 messages to the screen before terminating.

28810 STDOUT

28811 The *pr* utility output shall be a paginated version of the original file (or files). This pagination
28812 shall be accomplished using either <form-feed>s or a sequence of <newline>s, as controlled by
28813 XSI the –F or –f option. Page headers shall be generated unless the –t option is specified. The page
28814 headers shall be of the form:

28815 "\n\n%s %s Page %d\n\n\n", <output of date>, <file>, <page number>

28816 In the POSIX locale, the <output of date> field, representing the date and time of last modification
28817 of the input file (or the current date and time if the input file is standard input), shall be
28818 equivalent to the output of the following command as it would appear if executed at the given
28819 time:

28820 date "+%b %e %H:%M %Y"
28821 without the trailing <newline>, if the page being written is from standard input. If the page
28822 being written is not from standard input, in the POSIX locale, the same format shall be used, but
28823 the time used shall be the modification time of the file corresponding to *file* instead of the current
28824 time. When the *LC_TIME* locale category is not set to the POSIX locale, a different format and
28825 order of presentation of this field may be used.

28826 If the standard input is used instead of a *file* operand, the <*file*> field shall be replaced by a null
28827 string.

28828 If the **-h** option is specified, the <*file*> field shall be replaced by the *header* argument.

28829 **STDERR**

28830 The standard error shall be used for diagnostic messages and for alerting the terminal when **-p**
28831 is specified.

28832 **OUTPUT FILES**

28833 None.

28834 **EXTENDED DESCRIPTION**

28835 None.

28836 **EXIT STATUS**

28837 The following exit values shall be returned:

28838 0 Successful completion.

28839 >0 An error occurred.

28840 **CONSEQUENCES OF ERRORS**

28841 Default.

28842 **APPLICATION USAGE**

28843 None.

28844 **EXAMPLES**

28845 1. Print a numbered list of all files in the current directory:

28846 ls -a | pr -n -h "Files in \$(pwd)."

28847 2. Print **file1** and **file2** as a double-spaced, three-column listing headed by "file list":

28848 pr -3d -h "file list" file1 file2

28849 3. Write **file1** on **file2**, expanding tabs to columns 10, 19, 28, ...:

28850 pr -e9 -t <file1 >file2

28851 **RATIONALE**

28852 This utility is one of those that does not follow the Utility Syntax Guidelines because of its
28853 historical origins. The standard developers could have added new options that obeyed the
28854 guidelines (and marked the old options obsolescent) or devised an entirely new utility; there are
28855 examples of both actions in this volume of IEEE Std 1003.1-2001. Because of its widespread use
28856 by historical applications, the standard developers decided to exempt this version of *pr* from
28857 many of the guidelines.

28858 Implementations are required to accept option-arguments to the **-h**, **-l**, **-o**, and **-w** options
28859 whether presented as part of the same argument or as a separate argument to *pr*, as suggested by
28860 the Utility Syntax Guidelines. The **-n** and **-s** options, however, are specified as in historical
28861 practice because they are frequently specified without their optional arguments. If a <blank>

28862 were allowed before the option-argument in these cases, a *file* operand could mistakenly be
28863 interpreted as an option-argument in historical applications.

28864 The text about the minimum number of lines in multi-column output was included to ensure
28865 that a best effort is made in balancing the length of the columns. There are known historical
28866 implementations in which, for example, 60-line files are listed by *pr -2* as one column of 56 lines
28867 and a second of 4. Although this is not a problem when a full page with headers and trailers is
28868 produced, it would be relatively useless when used with *-t*.

28869 Historical implementations of the *pr* utility have differed in the action taken for the *-f* option.
28870 BSD uses it as described here for the *-F* option; System V uses it to change trailing <newline>s
28871 on each page to a <form-feed> and, if standard output is a TTY device, sends an <alert> to
28872 standard error and reads a line from /dev/tty before the first page. There were strong arguments
28873 from both sides of this issue concerning historical practice and as a result the *-F* option was
28874 added. XSI-conformant systems support the System V historical actions for the *-f* option.

28875 The <output of date> field in the *-l* format is specified only for the POSIX locale. As noted, the
28876 format can be different in other locales. No mechanism for defining this is present in this volume
28877 of IEEE Std 1003.1-2001, as the appropriate vehicle is a message catalog; that is, the format
28878 should be specified as a “message”.

28879 FUTURE DIRECTIONS

28880 None.

28881 SEE ALSO

28882 *expand, lp*

28883 CHANGE HISTORY

28884 First released in Issue 2.

28885 Issue 6

28886 The following new requirements on POSIX implementations derive from alignment with the
28887 Single UNIX Specification:

- 28888 • The *-p* option is added.

28889 The normative text is reworded to avoid use of the term “must” for application requirements.

28890 NAME

28891 printf — write formatted output

28892 SYNOPSIS

28893 `printf format[argument...]`

28894 DESCRIPTION

28895 The *printf* utility shall write formatted operands to the standard output. The *argument* operands
28896 shall be formatted under control of the *format* operand.

28897 OPTIONS

28898 None.

28899 OPERANDS

28900 The following operands shall be supported:

28901 *format* A string describing the format to use to write the remaining operands. See the
28902 EXTENDED DESCRIPTION section.

28903 *argument* The strings to be written to standard output, under the control of *format*. See the
28904 EXTENDED DESCRIPTION section.

28905 STDIN

28906 Not used.

28907 INPUT FILES

28908 None.

28909 ENVIRONMENT VARIABLES

28910 The following environment variables shall affect the execution of *printf*:

28911 *LANG* Provide a default value for the internationalization variables that are unset or null.
28912 (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2,
28913 Internationalization Variables for the precedence of internationalization variables
28914 used to determine the values of locale categories.)

28915 *LC_ALL* If set to a non-empty string value, override the values of all the other
28916 internationalization variables.

28917 *LC_CTYPE* Determine the locale for the interpretation of sequences of bytes of text data as
28918 characters (for example, single-byte as opposed to multi-byte characters in
28919 arguments).

28920 *LC_MESSAGES*

28921 Determine the locale that should be used to affect the format and contents of
28922 diagnostic messages written to standard error.

28923 *LC_NUMERIC*

28924 Determine the locale for numeric formatting. It shall affect the format of numbers
28925 written using the e, E, f, g, and G conversion specifier characters (if supported).

28926 XSI **NLSPATH** Determine the location of message catalogs for the processing of *LC_MESSAGES*.

28927 ASYNCHRONOUS EVENTS

28928 Default.

28929 STDOOUT

28930 See the EXTENDED DESCRIPTION section.

28931 **STDERR**

28932 The standard error shall be used only for diagnostic messages.

28933 **OUTPUT FILES**

28934 None.

28935 **EXTENDED DESCRIPTION**

28936 The *format* operand shall be used as the *format* string described in the Base Definitions volume of
28937 IEEE Std 1003.1-2001, Chapter 5, File Format Notation with the following exceptions:

- 28938 1. A <space> in the format string, in any context other than a flag of a conversion
28939 specification, shall be treated as an ordinary character that is copied to the output.
- 28940 2. A 'Δ' character in the format string shall be treated as a 'Δ' character, not as a <space>.
- 28941 3. In addition to the escape sequences shown in the Base Definitions volume of
28942 IEEE Std 1003.1-2001, Chapter 5, File Format Notation ('\\', '\a', '\b', '\f', '\n',
28943 '\r', '\t', '\v'), "\ddd", where *ddd* is a one, two, or three-digit octal number, shall be
28944 written as a byte with the numeric value specified by the octal number.
- 28945 4. The implementation shall not precede or follow output from the d or u conversion
28946 specifiers with <blank>s not specified by the *format* operand.
- 28947 5. The implementation shall not precede output from the o conversion specifier with zeros
28948 not specified by the *format* operand.
- 28949 6. The e, E, f, g, and G conversion specifiers need not be supported.
- 28950 7. An additional conversion specifier character, b, shall be supported as follows. The
28951 argument shall be taken to be a string that may contain backslash-escape sequences. The
28952 following backslash-escape sequences shall be supported:
 - 28953 — The escape sequences listed in the Base Definitions volume of IEEE Std 1003.1-2001,
28954 Chapter 5, File Format Notation ('\\', '\a', '\b', '\f', '\n', '\r', '\t', '\v'),
28955 which shall be converted to the characters they represent
 - 28956 — "\0ddd", where *ddd* is a zero, one, two, or three-digit octal number that shall be
28957 converted to a byte with the numeric value specified by the octal number
 - 28958 — '\c', which shall not be written and shall cause *printf* to ignore any remaining
28959 characters in the string operand containing it, any remaining string operands, and any
28960 additional characters in the *format* operand

28961 The interpretation of a backslash followed by any other sequence of characters is
28962 unspecified.

28963 Bytes from the converted string shall be written until the end of the string or the number of
28964 bytes indicated by the precision specification is reached. If the precision is omitted, it shall
28965 be taken to be infinite, so all bytes up to the end of the converted string shall be written.

- 28966 8. For each conversion specification that consumes an argument, the next argument operand
28967 shall be evaluated and converted to the appropriate type for the conversion as specified
28968 below.
- 28969 9. The *format* operand shall be reused as often as necessary to satisfy the argument operands.
28970 Any extra c or s conversion specifiers shall be evaluated as if a null string argument were
28971 supplied; other extra conversion specifications shall be evaluated as if a zero argument
28972 were supplied. If the *format* operand contains no conversion specifications and *argument*
28973 operands are present, the results are unspecified.

28974 10. If a character sequence in the *format* operand begins with a '%' character, but does not
28975 form a valid conversion specification, the behavior is unspecified.

28976 The *argument* operands shall be treated as strings if the corresponding conversion specifier is b,
28977 c, or s; otherwise, it shall be evaluated as a C constant, as described by the ISO C standard, with
28978 the following extensions:

- 28979 • A leading plus or minus sign shall be allowed.
- 28980 • If the leading character is a single-quote or double-quote, the value shall be the numeric
28981 value in the underlying codeset of the character following the single-quote or double-quote.

28982 If an argument operand cannot be completely converted into an internal value appropriate to
28983 the corresponding conversion specification, a diagnostic message shall be written to standard
28984 error and the utility shall not exit with a zero exit status, but shall continue processing any
28985 remaining operands and shall write the value accumulated at the time the error was detected to
28986 standard output.

28987 It is not considered an error if an argument operand is not completely used for a c or s
28988 conversion or if a string operand's first or second character is used to get the numeric value of a
28989 character.

28990 EXIT STATUS

28991 The following exit values shall be returned:

- 28992 0 Successful completion.
- 28993 >0 An error occurred.

28994 CONSEQUENCES OF ERRORS

28995 Default.

28996 APPLICATION USAGE

28997 The floating-point formatting conversion specifications of *printf()* are not required because all
28998 arithmetic in the shell is integer arithmetic. The awk utility performs floating-point calculations
28999 and provides its own printf function. The bc utility can perform arbitrary-precision floating-
29000 point arithmetic, but does not provide extensive formatting capabilities. (This printf utility
29001 cannot really be used to format bc output; it does not support arbitrary precision.)
29002 Implementations are encouraged to support the floating-point conversions as an extension.

29003 Note that this printf utility, like the printf() function defined in the System Interfaces volume of
29004 IEEE Std 1003.1-2001 on which it is based, makes no special provision for dealing with multi-
29005 byte characters when using the %c conversion specification or when a precision is specified in a
29006 %b or %s conversion specification. Applications should be extremely cautious using either of
29007 these features when there are multi-byte characters in the character set.

29008 No provision is made in this volume of IEEE Std 1003.1-2001 which allows field widths and
29009 precisions to be specified as '*' since the '*' can be replaced directly in the *format* operand
29010 using shell variable substitution. Implementations can also provide this feature as an extension
29011 if they so choose.

29012 Hexadecimal character constants as defined in the ISO C standard are not recognized in the
29013 *format* operand because there is no consistent way to detect the end of the constant. Octal
29014 character constants are limited to, at most, three octal digits, but hexadecimal character
29015 constants are only terminated by a non-hex-digit character. In the ISO C standard, the "##"
29016 concatenation operator can be used to terminate a constant and follow it with a hexadecimal
29017 character to be written. In the shell, concatenation occurs before the printf utility has a chance to
29018 parse the end of the hexadecimal constant.

29019 The %b conversion specification is not part of the ISO C standard; it has been added here as a
 29020 portable way to process backslash escapes expanded in string operands as provided by the *echo*
 29021 utility. See also the APPLICATION USAGE section of *echo* (on page 333) for ways to use *printf* as
 29022 a replacement for all of the traditional versions of the *echo* utility.

29023 If an argument cannot be parsed correctly for the corresponding conversion specification, the
 29024 *printf* utility is required to report an error. Thus, overflow and extraneous characters at the end
 29025 of an argument being used for a numeric conversion shall be reported as errors.

29026 EXAMPLES

29027 To alert the user and then print and read a series of prompts:

```
29028 printf "\aPlease fill in the following: \nName: "
29029 read name
29030 printf "Phone number: "
29031 read phone
```

29032 To read out a list of right and wrong answers from a file, calculate the percentage correctly, and
 29033 print them out. The numbers are right-justified and separated by a single <tab>. The percentage
 29034 is written to one decimal place of accuracy:

```
29035 while read right wrong ; do
29036     percent=$(echo "scale=1;($right*100)/($right+$wrong)" | bc)
29037     printf "%2d right\t%2d wrong\t(%s%%)\n" \
29038         $right $wrong $percent
29039 done < database_file
```

29040 The command:

```
29041 printf "%5d%4d\n" 1 21 321 4321 54321
```

29042 produces:

```
29043     1   21
29044     3214321
29045     54321   0
```

29046 Note that the *format* operand is used three times to print all of the given strings and that a '0'
 29047 was supplied by *printf* to satisfy the last %4d conversion specification.

29048 The *printf* utility is required to notify the user when conversion errors are detected while
 29049 producing numeric output; thus, the following results would be expected on an implementation
 29050 with 32-bit two's-complement integers when %d is specified as the *format* operand:

29051 Argument	29052 Standard Output	29053 Diagnostic Output
5a	5	printf: "5a" not completely converted
9999999999	2147483647	printf: "9999999999" arithmetic overflow
-9999999999	-2147483648	printf: "-9999999999" arithmetic overflow
ABC	0	printf: "ABC" expected numeric value

29057 The diagnostic message format is not specified, but these examples convey the type of
 29058 information that should be reported. Note that the value shown on standard output is what
 29059 would be expected as the return value from the *strtol()* function as defined in the System
 29060 Interfaces volume of IEEE Std 1003.1-2001. A similar correspondence exists between %u and
 29061 *strtoul()* and %e, %f, and %g (if the implementation supports floating-point conversions) and
 29062 *strtod()*.

29063 In a locale using the ISO/IEC 646: 1991 standard as the underlying codeset, the command:
29064 `printf "%d\n" 3 +3 -3 \'3 \'+"3 "'-3"`
29065 produces:
29066 3 Numeric value of constant 3
29067 3 Numeric value of constant 3
29068 -3 Numeric value of constant -3
29069 51 Numeric value of the character '3' in the ISO/IEC 646: 1991 standard codeset
29070 43 Numeric value of the character '+' in the ISO/IEC 646: 1991 standard codeset
29071 45 Numeric value of the character '-' in the ISO/IEC 646: 1991 standard codeset
29072 Note that in a locale with multi-byte characters, the value of a character is intended to be the
29073 value of the equivalent of the `wchar_t` representation of the character as described in the System
29074 Interfaces volume of IEEE Std 1003.1-2001.

29075 RATIONALE

29076 The *printf* utility was added to provide functionality that has historically been provided by *echo*.
29077 However, due to irreconcilable differences in the various versions of *echo* extant, the version has
29078 few special features, leaving those to this new *printf* utility, which is based on one in the Ninth
29079 Edition system.

29080 The EXTENDED DESCRIPTION section almost exactly matches the *printf()* function in the
29081 ISO C standard, although it is described in terms of the file format notation in the Base
29082 Definitions volume of IEEE Std 1003.1-2001, Chapter 5, File Format Notation.

29083 FUTURE DIRECTIONS

29084 None.

29085 SEE ALSO

29086 *awk*, *bc*, *echo*, the System Interfaces volume of IEEE Std 1003.1-2001, *printf()*

29087 CHANGE HISTORY

29088 First released in Issue 4.

29089 **NAME**29090 prs — print an SCCS file (**DEVELOPMENT**)29091 **SYNOPSIS**29092 XSI prs [-a][-d *dataspec*] [-r[*SID*]] *file...*29093 XSI prs [-e| -l] -c *cutoff* [-d *dataspec*] *file...*29094 XSI prs [-e| -l] -r[*SID*][-d *dataspec*] *file...*29095 **DESCRIPTION**29096 The *prs* utility shall write to standard output parts or all of an SCCS file in a user-supplied
29097 format.29098 **OPTIONS**29099 The *prs* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section 12.2,
29100 Utility Syntax Guidelines, except that the **-r** option has an optional option-argument. This
29101 optional option-argument cannot be presented as a separate argument. The following options
29102 shall be supported:29103 **-d *dataspec*** Specify the output data specification. The *dataspec* shall be a string consisting of
29104 SCCS file *data keywords* (see **Data Keywords** (on page 748)) interspersed with
29105 optional user-supplied text.29106 **-r[*SID*]** Specify the SCCS identification string (SID) of a delta for which information is
29107 desired. If no *SID* option-argument is specified, the SID of the most recently
29108 created delta shall be assumed.29109 **-e** Request information for all deltas created earlier than and including the delta
29110 designated via the **-r** option or the date-time given by the **-c** option.29111 **-l** Request information for all deltas created later than and including the delta
29112 designated via the **-r** option or the date-time given by the **-c** option.29113 **-c *cutoff*** Indicate the *cutoff* date-time, in the form:

29114 YY[MM[DD[HH[MM[SS]]]]]

29115 For the YY component, values in the range [69,99] shall refer to years 1969 to 1999
29116 inclusive, and values in the range [00,68] shall refer to years 2000 to 2068 inclusive.29117 **Note:** It is expected that in a future version of IEEE Std 1003.1-2001 the default
29118 century inferred from a 2-digit year will change. (This would apply to all
29119 commands accepting a 2-digit year as input.)29120 No changes (deltas) to the SCCS file that were created after the specified *cutoff*
29121 date-time shall be included in the output. Units omitted from the date-time default
29122 to their maximum possible values; for example, **-c 7502** is equivalent to
29123 **-c 750228235959**.29124 **-a** Request writing of information for both removed—that is, *delta type=R* (see
29125 *rmdel*)—and existing—that is, *delta type=D*,—deltas. If the **-a** option is not
29126 specified, information for existing deltas only shall be provided.29127 **OPERANDS**

29128 The following operand shall be supported:

29129 **file** A pathname of an existing SCCS file or a directory. If *file* is a directory, the *prs*
29130 utility shall behave as though each file in the directory were specified as a named
29131 file, except that non-SCCS files (last component of the pathname does not begin
29132 with **s.**) and unreadable files shall be silently ignored.

29133 If exactly one *file* operand appears, and it is ‘–’, the standard input shall be read;
29134 each line of the standard input shall be taken to be the name of an SCCS file to be
29135 processed. Non-SCCS files and unreadable files shall be silently ignored.

29136 **STDIN**

29137 The standard input shall be a text file used only when the *file* operand is specified as ‘–’. Each
29138 line of the text file shall be interpreted as an SCCS pathname.

29139 **INPUT FILES**

29140 Any SCCS files displayed are files of an unspecified format.

29141 **ENVIRONMENT VARIABLES**

29142 The following environment variables shall affect the execution of *prs*:

29143 *LANG* Provide a default value for the internationalization variables that are unset or null.
29144 (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2,
29145 Internationalization Variables for the precedence of internationalization variables
29146 used to determine the values of locale categories.)

29147 *LC_ALL* If set to a non-empty string value, override the values of all the other
29148 internationalization variables.

29149 *LC_CTYPE* Determine the locale for the interpretation of sequences of bytes of text data as
29150 characters (for example, single-byte as opposed to multi-byte characters in
29151 arguments and input files).

29152 *LC_MESSAGES*

29153 Determine the locale that should be used to affect the format and contents of
29154 diagnostic messages written to standard error.

29155 *NLSPATH* Determine the location of message catalogs for the processing of *LC_MESSAGES*.

29156 **ASYNCHRONOUS EVENTS**

29157 Default.

29158 **STDOUT**

29159 The standard output shall be a text file whose format is dependent on the data keywords
29160 specified with the **-d** option.

29161 **Data Keywords**

29162 Data keywords specify which parts of an SCCS file shall be retrieved and output. All parts of an
29163 SCCS file have an associated data keyword. A data keyword may appear in a *dataspec* multiple
29164 times.

29165 The information written by *prs* shall consist of:

- 29166 1. The user-supplied text
- 29167 2. Appropriate values (extracted from the SCCS file) substituted for the recognized data
29168 keywords in the order of appearance in the *dataspec*

29169 The format of a data keyword value shall either be simple (‘S’), in which keyword substitution
29170 is direct, or multi-line (‘M’).

29171 User-supplied text shall be any text other than recognized data keywords. A *<tab>* shall be
29172 specified by ‘\t’ and *<newline>* by ‘\n’. When the **-r** option is not specified, the default
29173 *dataspec* shall be:

29174 :PN: :\n\n

29175 and the following *dataspec* shall be used for each selected delta:

29176 :Dt:\t:DL:\nMRs:\n:MR:COMMENTS:\n:C:

29177

29178

29179

SCCS File Data Keywords				
Keyword	Data Item	File Section	Value	Format
:Dt:	Delta information	Delta Table	See below*	S
:DL:	Delta line statistics	"	:Li:/:Ld:/:Lu:	S
:Li:	Lines inserted by Delta	"	nnnnn***	S
:Ld:	Lines deleted by Delta	"	nnnnn***	S
:Lu:	Lines unchanged by Delta	"	nnnnn***	S
:DT:	Delta type	"	D or R	S
:I:	SCCS ID string (SID)	"	See below**	S
:R:	Release number	"	nnnn	S
:L:	Level number	"	nnnn	S
:B:	Branch number	"	nnnn	S
:S:	Sequence number	"	nnnn	S
:D:	Date delta created	"	:Dy:/:Dm:/:Dd:	S
:Dy:	Year delta created	"	nn	S
:Dm:	Month delta created	"	nn	S
:Dd:	Day delta created	"	nn	S
:T:	Time delta created	"	:Th:::Tm:::Ts:	S
:Th:	Hour delta created	"	nn	S
:Tm:	Minutes delta created	"	nn	S
:Ts:	Seconds delta created	"	nn	S
:P:	Programmer who created Delta	"	logname	S
:DS:	Delta sequence number	"	nnnn	S
:DP:	Predecessor Delta sequence number	"	nnnn	S
:DI:	Sequence number of deltas included, excluded, or ignored	"	:Dn:/:Dx:/:Dg:	S
:Dn:	Deltas included (sequence #)	"	:DS: :DS: ...	S
:Dx:	Deltas excluded (sequence #)	"	:DS: :DS: ...	S
:Dg:	Deltas ignored (sequence #)	"	:DS: :DS: ...	S
:MR:	MR numbers for delta	"	text	M
:C:	Comments for delta	"	text	M
:UN:	User names	User Names	text	M
:FL:	Flag list	Flags	text	M
:Y:	Module type flag	"	text	S
:MF:	MR validation flag	"	yes or no	S
:MP:	MR validation program name	"	text	S
:KF:	Keyword error, warning flag	"	yes or no	S
:KV:	Keyword validation string	"	text	S
:BF:	Branch flag	"	yes or no	S
:J:	Joint edit flag	"	yes or no	S
:LK:	Locked releases	"	:R: ...	S
:Q:	User-defined keyword	"	text	S
:M:	Module name	"	text	S

29222

29223

SCCS File Data Keywords				
Keyword	Data Item	File Section	Value	Format
:FB:	Floor boundary	"	:R:	S
:CB:	Ceiling boundary	"	:R:	S
:Ds:	Default SID	"	:I:	S
:ND:	Null delta flag	"	yes or no	S
:FD:	File descriptive text	Comments	text	M
:BD:	Body	Body	text	M
:GB:	Gotten body	"	text	M
:W:	A form of what string	N/A	:Z::M:\t:I:	S
:A:	A form of what string	N/A	:Z::Y::M::I::Z:	S
:Z:	what string delimiter	N/A	@(#)	S
:F:	SCCS filename	N/A	text	S
:PN:	SCCS file pathname	N/A	text	S

29237

* :Dt:=:DT: :I: :D: :T: :P: :DS: :DP:

29238

** :R::L::B::S: if the delta is a branch delta (:BF:=yes)
:R::L: if the delta is not a branch delta (:BF:=no)

29240

*** The line statistics are capped at 99 999. For example, if 100 000 lines were unchanged in a certain revision, :Lu: shall produce the value 99 999.

29242 **STDERR**

29243

The standard error shall be used only for diagnostic messages.

29244 **OUTPUT FILES**

29245

None.

29246 **EXTENDED DESCRIPTION**

29247

None.

29248 **EXIT STATUS**

29249

The following exit values shall be returned:

29250

0 Successful completion.

29251

>0 An error occurred.

29252 **CONSEQUENCES OF ERRORS**

29253

Default.

29254 **APPLICATION USAGE**

29255

None.

29256 **EXAMPLES**

29257

1. The following example:

29258

prs -d "User Names for :F: are:\n:UN:" s.file

29259

might write to standard output:

29260

User Names for s.file are:

29261

xyz

29262

131

29263

abc

29264

2. The following example:

29265 prs -d "Delta for pgm :M:: :I: - :D: By :P:" -r s.file
29266 might write to standard output:
29267 Delta for pgm main.c: 3.7 - 77/12/01 By cas
29268 3. As a special case:
29269 prs s.file
29270 might write to standard output:
29271 s.file:
29272 <blank line>
29273 D 1.1 77/12/01 00:00:00 cas 1 000000/00000/00000
29274 MRS:
29275 bl78-12345
29276 bl79-54321
29277 COMMENTS:
29278 this is the comment line for s.file initial delta
29279 <blank line>
29280 for each delta table entry of the **D** type. The only option allowed to be used with this
29281 special case is the **-a** option.

29282 RATIONALE

29283 None.

29284 FUTURE DIRECTIONS

29285 None.

29286 SEE ALSO

29287 *admin, delta, get, what*

29288 CHANGE HISTORY

29289 First released in Issue 2.

29290 Issue 5

29291 The phrase “in which keyword substitution is followed by a <newline>” is deleted from the end
29292 of the second paragraph of **Data Keywords** (on page 748).

29293 The interpretation of the YY component of the **-c cutoff** argument is noted.

29294 Issue 6

29295 The normative text is reworded to emphasize the term “shall” for implementation requirements.

29296 The Open Group Base Resolution bwg2001-007 is applied, updating the table in STDOUT with a
29297 note that line statistics are capped at 99 999 for the **:Li:**, **:Ld:**, **:Lu:**, and **:DL:** keywords.

29298 The Open Group Interpretation PIN4C.00009 is applied.

29299 NAME

29300 ps — report process status

29301 SYNOPSIS

29302 UP XSI ps [-aA][-defl][-G *grouplist*] [-o *format*]... [-p *proclist*] [-t *termlist*]29303 [-U *userlist*] [-g *grouplist*] [-n *namelist*] [-u *userlist*]

29304

29305 DESCRIPTION

29306 The *ps* utility shall write information about processes, subject to having the appropriate
29307 privileges to obtain information about those processes.

29308 By default, *ps* shall select all processes with the same effective user ID as the current user and the
29309 same controlling terminal as the invoker.

29310 OPTIONS

29311 The *ps* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section 12.2,
29312 Utility Syntax Guidelines.

29313 The following options shall be supported:

29314 **-a** Write information for all processes associated with terminals. Implementations
29315 may omit session leaders from this list.

29316 **-A** Write information for all processes.

29317 XSI **-d** Write information for all processes, except session leaders.

29318 XSI **-e** Write information for all processes. (Equivalent to **-A**.)

29319 XSI **-f** Generate a **full** listing. (See the STDOUT section for the contents of a **full** listing.)

29320 XSI **-g *grouplist*** Write information for processes whose session leaders are given in *grouplist*. The
29321 application shall ensure that the *grouplist* is a single argument in the form of a
29322 <blank> or comma-separated list.

29323 **-G *grouplist*** Write information for processes whose real group ID numbers are given in *grouplist*. The application shall ensure that the *grouplist* is a single argument in the
29324 form of a <blank> or comma-separated list.
29325

29326 XSI **-l** Generate a **long** listing. (See STDOUT for the contents of a **long** listing.)

29327 XSI **-n *namelist*** Specify the name of an alternative system *namelist* file in place of the default. The
29328 name of the default file and the format of a *namelist* file are unspecified.

29329 **-o *format*** Write information according to the format specification given in *format*. This is
29330 fully described in the STDOUT section. Multiple **-o** options can be specified; the
29331 format specification shall be interpreted as the <space>-separated concatenation of
29332 all the *format* option-arguments.

29333 **-p *proclist*** Write information for processes whose process ID numbers are given in *proclist*.
29334 The application shall ensure that the *proclist* is a single argument in the form of a
29335 <blank> or comma-separated list.

29336 **-t *termlist*** Write information for processes associated with terminals given in *termlist*. The
29337 application shall ensure that the *termlist* is a single argument in the form of a
29338 <blank> or comma-separated list. Terminal identifiers shall be given in an
29339 XSI implementation-defined format. On XSI-conformant systems, they shall be given
29340 in one of two forms: the device's filename (for example, **tty04**) or, if the device's
29341 filename starts with **tty**, just the identifier following the characters **tty** (for

29342		example, "0 4").
29343 XSI	-u userlist	Write information for processes whose user ID numbers or login names are given in <i>userlist</i> . The application shall ensure that the <i>userlist</i> is a single argument in the form of a <blank> or comma-separated list. In the listing, the numerical user ID shall be written unless the -f option is used, in which case the login name shall be written.
29344		
29345		
29346		
29347		
29348	-U userlist	Write information for processes whose real user ID numbers or login names are given in <i>userlist</i> . The application shall ensure that the <i>userlist</i> is a single argument in the form of a <blank> or comma-separated list.
29349		
29350		
29351		With the exception of -o format , all of the options shown are used to select processes. If any are specified, the default list shall be ignored and <i>ps</i> shall select the processes represented by the inclusive OR of all the selection-criteria options.
29352		
29353		
29354	OPERANDS	
29355		None.
29356	STDIN	
29357		Not used.
29358	INPUT FILES	
29359		None.
29360	ENVIRONMENT VARIABLES	
29361		The following environment variables shall affect the execution of <i>ps</i> :
29362	COLUMNS	Override the system-selected horizontal display line size, used to determine the number of text columns to display. See the Base Definitions volume of IEEE Std 1003.1-2001, Chapter 8, Environment Variables for valid values and results when it is unset or null.
29363		
29364		
29365		
29366	LANG	Provide a default value for the internationalization variables that are unset or null. (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2, Internationalization Variables for the precedence of internationalization variables used to determine the values of locale categories.)
29367		
29368		
29369		
29370	LC_ALL	If set to a non-empty string value, override the values of all the other internationalization variables.
29371		
29372	LC_CTYPE	Determine the locale for the interpretation of sequences of bytes of text data as characters (for example, single-byte as opposed to multi-byte characters in arguments).
29373		
29374		
29375	LC_MESSAGES	
29376		Determine the locale that should be used to affect the format and contents of diagnostic messages written to standard error and informative messages written to standard output.
29377		
29378		
29379	LC_TIME	Determine the format and contents of the date and time strings displayed.
29380 XSI	NLSPATH	Determine the location of message catalogs for the processing of LC_MESSAGES .
29381	TZ	Determine the timezone used to calculate date and time strings displayed. If TZ is unset or null, an unspecified default timezone shall be used.
29382		

29383 ASYNCHRONOUS EVENTS

29384 Default.

29385 STDOUT

29386 When the **-o** option is not specified, the standard output format is unspecified.

29387 XSI On XSI-conformant systems, the output format shall be as follows. The column headings and descriptions of the columns in a *ps* listing are given below. The precise meanings of these fields are implementation-defined. The letters '**f**' and '**l**' (below) indicate the option (**full** or **long**) that shall cause the corresponding heading to appear; **all** means that the heading always appears. Note that these two options determine only what information is provided for a process; they do not determine which processes are listed.

29393	F	(l)	Flags (octal and additive) associated with the process.
29394	S	(l)	The state of the process.
29395	UID	(f,l)	The user ID number of the process owner; the login name is printed under the -f option.
29396	PID	(all)	The process ID of the process; it is possible to kill a process if this datum is known.
29397	PPID	(f,l)	The process ID of the parent process.
29400	C	(f,l)	Processor utilization for scheduling.
29401	PRI	(l)	The priority of the process; higher numbers mean lower priority.
29402	NI	(l)	Nice value; used in priority computation.
29403	ADDR	(l)	The address of the process.
29404	SZ	(l)	The size in blocks of the core image of the process.
29405	WCHAN	(l)	The event for which the process is waiting or sleeping; if blank, the process is running.
29407	STIME	(f)	Starting time of the process.
29408	TTY	(all)	The controlling terminal for the process.
29409	TIME	(all)	The cumulative execution time for the process.
29410	CMD	(all)	The command name; the full command name and its arguments are written under the -f option.
29411			

29412 A process that has exited and has a parent, but has not yet been waited for by the parent, shall be marked **defunct**.

29414 Under the option **-f**, *ps* tries to determine the command name and arguments given when the process was created by examining memory or the swap area. Failing this, the command name, as it would appear without the option **-f**, is written in square brackets.

29417 The **-o** option allows the output format to be specified under user control.

29418 The application shall ensure that the format specification is a list of names presented as a single argument, <blank> or comma-separated. Each variable has a default header. The default header can be overridden by appending an equals sign and the new text of the header. The rest of the characters in the argument shall be used as the header text. The fields specified shall be written in the order specified on the command line, and should be arranged in columns in the output. The field widths shall be selected by the system to be at least as wide as the header text (default or overridden value). If the header text is null, such as **-o user=**, the field width shall be at least as wide as the default header text. If all header text fields are null, no header line shall be written.

29427 The following names are recognized in the POSIX locale:

29428	ruser	The real user ID of the process. This shall be the textual user ID, if it can be obtained and the field width permits, or a decimal representation otherwise.
29429		
29430	user	The effective user ID of the process. This shall be the textual user ID, if it can be obtained and the field width permits, or a decimal representation otherwise.
29431		
29432	rgroup	The real group ID of the process. This shall be the textual group ID, if it can be obtained and the field width permits, or a decimal representation otherwise.
29433		
29434	group	The effective group ID of the process. This shall be the textual group ID, if it can be obtained and the field width permits, or a decimal representation otherwise.
29435		
29436	pid	The decimal value of the process ID.
29437	ppid	The decimal value of the parent process ID.
29438	pgid	The decimal value of the process group ID.
29439	pcpu	The ratio of CPU time used recently to CPU time available in the same period, expressed as a percentage. The meaning of “recently” in this context is unspecified. The CPU time available is determined in an unspecified manner.
29440		
29441		
29442	vsz	The size of the process in (virtual) memory in 1 024 byte units as a decimal integer.
29443	nice	The decimal value of the nice value of the process; see <i>nice</i> .
29444	etime	In the POSIX locale, the elapsed time since the process was started, in the form: [[dd-]hh:]mm:ss
29445		
29446		where <i>dd</i> shall represent the number of days, <i>hh</i> the number of hours, <i>mm</i> the number of minutes, and <i>ss</i> the number of seconds. The <i>dd</i> field shall be a decimal integer. The <i>hh</i> , <i>mm</i> , and <i>ss</i> fields shall be two-digit decimal integers padded on the left with zeros.
29447		
29448		
29449	time	In the POSIX locale, the cumulative CPU time of the process in the form: [dd-]hh:mm:ss
29450		
29451		The <i>dd</i> , <i>hh</i> , <i>mm</i> , and <i>ss</i> fields shall be as described in the etime specifier.
29452	tty	The name of the controlling terminal of the process (if any) in the same format used by the <i>who</i> utility.
29453		
29454	comm	The name of the command being executed (<i>argv[0]</i> value) as a string.
29455	args	The command with all its arguments as a string. The implementation may truncate this value to the field width; it is implementation-defined whether any further truncation occurs. It is unspecified whether the string represented is a version of the argument list as it was passed to the command when it started, or is a version of the arguments as they may have been modified by the application. Applications cannot depend on being able to modify their argument list and having that modification be reflected in the output of <i>ps</i> .
29456		
29457		
29458		
29459		
29460		
29461		
29462		Any field need not be meaningful in all implementations. In such a case a hyphen ('-') should be output in place of the field value.
29463		
29464		Only comm and args shall be allowed to contain <blank>s; all others shall not. Any implementation-defined variables shall be specified in the system documentation along with the default header and indicating whether the field may contain <blank>s.
29465		
29466		
29467		The following table specifies the default header to be used in the POSIX locale corresponding to each format specifier.
29468		

29469

Table 4-17 Variable Names and Default Headers in *ps*

Format Specifier	Default Header	Format Specifier	Default Header
args	COMMAND	ppid	PPID
comm	COMMAND	rgroup	RGROUP
etime	ELAPSED	ruser	RUSER
group	GROUP	time	TIME
nice	NI	tty	TT
pcpu	%CPU	user	USER
pgid	PGID	vsz	VSZ
pid	PID		

29479 STDERR

29480 The standard error shall be used only for diagnostic messages.

29481 OUTPUT FILES

29482 None.

29483 EXTENDED DESCRIPTION

29484 None.

29485 EXIT STATUS

29486 The following exit values shall be returned:

29487 0 Successful completion.

29488 >0 An error occurred.

29489 CONSEQUENCES OF ERRORS

29490 Default.

29491 APPLICATION USAGE

29492 Things can change while *ps* is running; the snapshot it gives is only true for an instant, and might not be accurate by the time it is displayed.

29494 The **args** format specifier is allowed to produce a truncated version of the command arguments.
29495 In some implementations, this information is no longer available when the *ps* utility is executed.

29496 If the field width is too narrow to display a textual ID, the system may use a numeric version.
29497 Normally, the system would be expected to choose large enough field widths, but if a large
29498 number of fields were selected to write, it might squeeze fields to their minimum sizes to fit on
29499 one line. One way to ensure adequate width for the textual IDs is to override the default header
29500 for a field to make it larger than most or all user or group names.

29501 There is no special quoting mechanism for header text. The header text is the rest of the
29502 argument. If multiple header changes are needed, multiple **-o** options can be used, such as:

29503 `ps -o "user=User Name" -o pid=Process\ ID`

29504 On some implementations, especially multi-level secure systems, *ps* may be severely restricted
29505 and produce information only about child processes owned by the user.

29506 EXAMPLES

29507 The command:

29508 `ps -o user,pid,ppid=MOM -o args`

29509 writes at least the following in the POSIX locale:

29510 USER PID MOM COMMAND
29511 helene 34 12 ps -o uid,pid,ppid=MOM -o args

29512 The contents of the **COMMAND** field need not be the same in all implementations, due to
29513 possible truncation.

29514 **RATIONALE**

29515 There is very little commonality between BSD and System V implementations of *ps*. Many
29516 options conflict or have subtly different usages. The standard developers attempted to select a
29517 set of options for the base standard that were useful on a wide range of systems and selected
29518 options that either can be implemented on both BSD and System V-based systems without
29519 breaking the current implementations or where the options are sufficiently similar that any
29520 changes would not be unduly problematic for users or implementors.

29521 It is recognized that on some implementations, especially multi-level secure systems, *ps* may be
29522 nearly useless. The default output has therefore been chosen such that it does not break
29523 historical implementations and also is likely to provide at least some useful information on most
29524 systems.

29525 The major change is the addition of the format specification capability. The motivation for this
29526 invention is to provide a mechanism for users to access a wider range of system information, if
29527 the system permits it, in a portable manner. The fields chosen to appear in this volume of
29528 IEEE Std 1003.1-2001 were arrived at after considering what concepts were likely to be both
29529 reasonably useful to the “average” user and had a reasonable chance of being implemented on a
29530 wide range of systems. Again it is recognized that not all systems are able to provide all the
29531 information and, conversely, some may wish to provide more. It is hoped that the approach
29532 adopted will be sufficiently flexible and extensible to accommodate most systems.
29533 Implementations may be expected to introduce new format specifiers.

29534 The default output should consist of a short listing containing the process ID, terminal name,
29535 cumulative execution time, and command name of each process.

29536 The preference of the standard developers would have been to make the format specification an
29537 operand of the *ps* command. Unfortunately, BSD usage precluded this.

29538 At one time a format was included to display the environment array of the process. This was
29539 deleted because there is no portable way to display it.

29540 The **-A** option is equivalent to the BSD **-g** and the SVID **-e**. Because the two systems differed, a
29541 mnemonic compromise was selected.

29542 The **-a** option is described with some optional behavior because the SVID omits session leaders,
29543 but BSD does not.

29544 In an early proposal, format specifiers appeared for priority and start time. The former was not
29545 defined adequately in this volume of IEEE Std 1003.1-2001 and was removed in deference to the
29546 defined nice value; the latter because elapsed time was considered to be more useful.

29547 In a new BSD version of *ps*, a **-O** option can be used to write all of the default information,
29548 followed by additional format specifiers. This was not adopted because the default output is
29549 implementation-defined. Nevertheless, this is a useful option that should be reserved for that
29550 purpose. In the **-o** option for the POSIX Shell and Utilities *ps*, the format is the concatenation of
29551 each **-o**. Therefore, the user can have an alias or function that defines the beginning of their
29552 desired format and add more fields to the end of the output in certain cases where that would be
29553 useful.

29554 The format of the terminal name is unspecified, but the descriptions of *ps*, *talk*, *who*, and *write*
29555 require that they all use the same format.

29556 The **pcpu** field indicates that the CPU time available is determined in an unspecified manner.
29557 This is because it is difficult to express an algorithm that is useful across all possible machine

29558 architectures. Historical counterparts to this value have attempted to show percentage of use in
29559 the recent past, such as the preceding minute. Frequently, these values for all processes did not
29560 add up to 100%. Implementations are encouraged to provide data in this field to users that will
29561 help them identify processes currently affecting the performance of the system.

29562 **FUTURE DIRECTIONS**

29563 None.

29564 **SEE ALSO**

29565 *kill, nice, renice*

29566 **CHANGE HISTORY**

29567 First released in Issue 2.

29568 **Issue 6**

29569 This utility is marked as part of the User Portability Utilities option.

29570 The normative text is reworded to avoid use of the term “must” for application requirements.

29571 The *TZ* entry is added to the ENVIRONMENT VARIABLES section.

29572 NAME

29573 *pwd* — return working directory name

29574 SYNOPSIS

29575 *pwd* [−L | −P]

29576 DESCRIPTION

29577 The *pwd* utility shall write to standard output an absolute pathname of the current working directory, which does not contain the filenames dot or dot-dot.

29579 OPTIONS

29580 The *pwd* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section 12.2, Utility Syntax Guidelines.

29582 The following options shall be supported by the implementation:

29583 **−L** If the *PWD* environment variable contains an absolute pathname of the current directory that does not contain the filenames dot or dot-dot, *pwd* shall write this pathname to standard output. Otherwise, the **−L** option shall behave as the **−P** option.

29587 **−P** The absolute pathname written shall not contain filenames that, in the context of the pathname, refer to files of type symbolic link.

29589 If both **−L** and **−P** are specified, the last one shall apply. If neither **−L** nor **−P** is specified, the *pwd* utility shall behave as if **−L** had been specified.

29591 OPERANDS

29592 None.

29593 STDIN

29594 Not used.

29595 INPUT FILES

29596 None.

29597 ENVIRONMENT VARIABLES

29598 The following environment variables shall affect the execution of *pwd*:

29599 **LANG** Provide a default value for the internationalization variables that are unset or null. (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2, Internationalization Variables for the precedence of internationalization variables used to determine the values of locale categories.)

29603 **LC_ALL** If set to a non-empty string value, override the values of all the other internationalization variables.

29605 *LC_MESSAGES*

29606 Determine the locale that should be used to affect the format and contents of diagnostic messages written to standard error.

29608 **XSI *NLSPATH*** Determine the location of message catalogs for the processing of *LC_MESSAGES*.

29609 **PWD** If the **−P** option is in effect, this variable shall be set to an absolute pathname of the current working directory that does not contain any components that specify symbolic links, does not contain any components that are dot, and does not contain any components that are dot-dot. If an application sets or unsets the value of *PWD*, the behavior of *pwd* is unspecified.

29614 ASYNCHRONOUS EVENTS

29615 Default.

29616 STDOUT

29617 The *pwd* utility output is an absolute pathname of the current working directory:

29618 "%s\n", <directory pathname>

29619 STDERR

29620 The standard error shall be used only for diagnostic messages.

29621 OUTPUT FILES

29622 None.

29623 EXTENDED DESCRIPTION

29624 None.

29625 EXIT STATUS

29626 The following exit values shall be returned:

29627 0 Successful completion.

29628 >0 An error occurred.

29629 CONSEQUENCES OF ERRORS

29630 If an error is detected, output shall not be written to standard output, a diagnostic message shall
29631 be written to standard error, and the exit status is not zero.

29632 APPLICATION USAGE

29633 None.

29634 EXAMPLES

29635 None.

29636 RATIONALE

29637 Some implementations have historically provided *pwd* as a shell special built-in command.

29638 In most utilities, if an error occurs, partial output may be written to standard output. This does
29639 not happen in historical implementations of *pwd*. Because *pwd* is frequently used in historical
29640 shell scripts without checking the exit status, it is important that the historical behavior is
29641 required here; therefore, the CONSEQUENCES OF ERRORS section specifically disallows any
29642 partial output being written to standard output.

29643 FUTURE DIRECTIONS

29644 None.

29645 SEE ALSO

29646 *cd*, the System Interfaces volume of IEEE Std 1003.1-2001, *getcwd()*

29647 CHANGE HISTORY

29648 First released in Issue 2.

29649 Issue 6

29650 The **-P** and **-L** options are added to describe actions relating to symbolic links as specified in the
29651 IEEE P1003.2b draft standard.

29652 NAME

29653 qalter — alter batch job

29654 SYNOPSIS

```
29655 BE    qalter [-a date_time][-A account_string][-c interval][-e path_name]
29656      [-h hold_list][-j join_list][-k keep_list][-l resource_list]
29657      [-m mail_options][-M mail_list][-N name][-o path_name]
29658      [-p priority][-r y|n][-S path_name_list][-u user_list]
29659      job_identifier ...
29660
```

29661 DESCRIPTION

29662 The attributes of a batch job are altered by a request to the batch server that manages the batch
 29663 job. The *qalter* utility is a user-accessible batch client that requests the alteration of the attributes
 29664 of one or more batch jobs.

29665 The *qalter* utility shall alter the attributes of those batch jobs, and only those batch jobs, for which
 29666 a batch *job_identifier* is presented to the utility.

29667 The *qalter* utility shall alter the attributes of batch jobs in the order in which the batch
 29668 *job_identifiers* are presented to the utility.

29669 If the *qalter* utility fails to process a batch *job_identifier* successfully, the utility shall proceed to
 29670 process the remaining batch *job_identifiers*, if any.

29671 For each batch *job_identifier* for which the *qalter* utility succeeds, each attribute of the identified
 29672 batch job shall be altered as indicated by all the options presented to the utility.

29673 For each identified batch job for which the *qalter* utility fails, the utility shall not alter any
 29674 attribute of the batch job.

29675 For each batch job that the *qalter* utility processes, the utility shall not modify any attribute other
 29676 than those required by the options and option-arguments presented to the utility.

29677 The *qalter* utility shall alter batch jobs by sending a *Modify Job Request* to the batch server that
 29678 manages each batch job. At the time the *qalter* utility exits, it shall have modified the batch job
 29679 corresponding to each successfully processed batch *job_identifier*. An attempt to alter the
 29680 attributes of a batch job in the RUNNING state is implementation-defined.

29681 OPTIONS

29682 The *qalter* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section
 29683 12.2, Utility Syntax Guidelines.

29684 The following options shall be supported by the implementation:

29685 **-a date_time** Redefine the time at which the batch job becomes eligible for execution.

29686 The *date_time* argument shall be in the same form and represent the same time as
 29687 for the *touch* utility. The time so represented shall be set into the *Execution_Time*
 29688 attribute of the batch job. If the time specified is earlier than the current time, the
 29689 **-a** option shall have no effect.

29690 **-A account_string**

29691 Redefine the account to which the resource consumption of the batch job should be
 29692 charged.

29693 The syntax of the *account_string* option-argument is unspecified.

29694 The *qalter* utility shall set the *Account_Name* attribute of the batch job to the value
 29695 of the *account_string* option-argument.

- 29696 **-c interval** Redefine whether the batch job should be checkpointed, and if so, how often.
- 29697 The *qalter* utility shall accept a value for the *interval* option-argument that is one of
29698 the following:
- 29699 n No checkpointing is to be performed on the batch job
29700 (NO_CHECKPOINT).
- 29701 s Checkpointing is to be performed only when the batch server is shut
29702 down (CHECKPOINT_AT_SHUTDOWN).
- 29703 c Automatic periodic checkpointing is to be performed at the
29704 *Minimum_Cpu_Interval* attribute of the batch queue, in units of CPU
29705 minutes (CHECKPOINT_AT_MIN_CPU_INTERVAL).
- 29706 c=minutes Automatic periodic checkpointing is to be performed every *minutes*
29707 of CPU time, or every *Minimum_Cpu_Interval* minutes, whichever is
29708 greater. The *minutes* argument shall conform to the syntax for
29709 unsigned integers and shall be greater than zero.
- 29710 An implementation may define other checkpoint intervals. The conformance
29711 document for an implementation shall describe any alternative checkpoint
29712 intervals, how they are specified, their internal behavior, and how they affect the
29713 behavior of the utility.
- 29714 The *qalter* utility shall set the *Checkpoint* attribute of the batch job to the value of the
29715 *interval* option-argument.
- 29716 **-e path_name** Redefine the path to be used for the standard error stream of the batch job.
- 29717 The *qalter* utility shall accept a *path_name* option-argument that conforms to the
29718 syntax of the *path_name* element defined in the System Interfaces volume of
29719 IEEE Std 1003.1-2001, which can be preceded by a host name element of the form
29720 *hostname*:.
- 29721 If the *path_name* option-argument constitutes an absolute pathname, the *qalter*
29722 utility shall set the *Error_Path* attribute of the batch job to the value of the
29723 *path_name* option-argument, including the host name element, if present.
- 29724 If the *path_name* option-argument constitutes a relative pathname and no host
29725 name element is specified, the *qalter* utility shall set the *Error_Path* attribute of the
29726 batch job to the value of the absolute pathname derived by expanding the
29727 *path_name* option-argument relative to the current directory of the process that
29728 executes the *qalter* utility.
- 29729 If the *path_name* option-argument constitutes a relative pathname and a host name
29730 element is specified, the *qalter* utility shall set the *Error_Path* attribute of the batch
29731 job to the value of the option-argument without expansion.
- 29732 If the *path_name* option-argument does not include a host name element, the *qalter*
29733 utility shall prefix the pathname in the *Error_Path* attribute with *hostname*:,
29734 where *hostname* is the name of the host upon which the *qalter* utility is being executed.
- 29735 **-h hold_list** Redefine the types of holds, if any, on the batch job. The *qalter -h* option shall
29736 accept a value for the *hold_list* option-argument that is a string of alphanumeric
29737 characters in the portable character set.
- 29738 The *qalter* utility shall accept a value for the *hold_list* option-argument that is a
29739 string of one or more of the characters 'u', 's', or 'o', or the single character

29741 'n'. For each unique character in the *hold_list* option-argument, the *qalter* utility
29742 shall add a value to the *Hold_Types* attribute of the batch job as follows, each
29743 representing a different hold type:

- 29744 u USER
29745 s SYSTEM
29746 o OPERATOR

29747 If any of these characters are duplicated in the *hold_list* option-argument, the
29748 duplicates shall be ignored. An existing *Hold_Types* attribute can be cleared by the
29749 hold type:

- 29750 n NO_HOLD

29751 The *qalter* utility shall consider it an error if any hold type other than 'n' is
29752 combined with hold type 'n'. Strictly conforming applications shall not repeat
29753 any of the characters 'u', 's', 'o', or 'n' within the *hold_list* option-argument.
29754 The *qalter* utility shall permit the repetition of characters, but shall not assign
29755 additional meaning to the repeated characters. An implementation may define
29756 other hold types. The conformance document for an implementation shall describe
29757 any additional hold types, how they are specified, their internal behavior, and how
29758 they affect the behavior of the utility.

29759 **-j** *join_list* Redefine which streams of the batch job are to be merged. The *qalter -j* option shall
29760 accept a value for the *join_list* option-argument that is a string of alphanumeric
29761 characters in the portable character set.

29762 The *qalter* utility shall accept a *join_list* option-argument that consists of one or
29763 more of the characters 'e' and 'o', or the single character 'n'.

29764 All of the other batch job output streams specified shall be merged into the output
29765 stream represented by the character listed first in the *join_list* option-argument.

29766 For each unique character in the *join_list* option-argument, the *qalter* utility shall
29767 add a value to the *Join_Path* attribute of the batch job as follows, each representing
29768 a different batch job stream to join:

- 29769 e The standard error of the batch job (JOIN_STD_ERROR).
29770 o The standard output of the batch job (JOIN_STD_OUTPUT).

29771 An existing *Join_Path* attribute can be cleared by the join type:

- 29772 n NO_JOIN

29773 If 'n' is specified, then no files are joined. The *qalter* utility shall consider it an
29774 error if any join type other than 'n' is combined with join type 'n'.

29775 Strictly conforming applications shall not repeat any of the characters 'e', 'o', or
29776 'n' within the *join_list* option-argument. The *qalter* utility shall permit the
29777 repetition of characters, but shall not assign additional meaning to the repeated
29778 characters.

29779 An implementation may define other join types. The conformance document for an
29780 implementation shall describe any additional batch job streams, how they are
29781 specified, their internal behavior, and how they affect the behavior of the utility.

29782 **-k** *keep_list* Redefine which output of the batch job to retain on the execution host.

29783 The *qalter -k* option shall accept a value for the *keep_list* option-argument that is a
29784 string of alphanumeric characters in the portable character set.

29785 The *qalter* utility shall accept a *keep_list* option-argument that consists of one or
29786 more of the characters 'e' and 'o', or the single character 'n'.

29787 For each unique character in the *keep_list* option-argument, the *qalter* utility shall
29788 add a value to the *Keep_Files* attribute of the batch job as follows, each representing
29789 a different batch job stream to keep:

- 29790 e The standard error of the batch job (KEEP_STD_ERROR).
29791 o The standard output of the batch job (KEEP_STD_OUTPUT).

29792 If both 'e' and 'o' are specified, then both files are retained. An existing
29793 *Keep_Files* attribute can be cleared by the keep type:

29794 n NO_KEEP

29795 If 'n' is specified, then no files are retained. The *qalter* utility shall consider it an
29796 error if any keep type other than 'n' is combined with keep type 'n'.

29797 Strictly conforming applications shall not repeat any of the characters 'e', 'o', or
29798 'n' within the *keep_list* option-argument. The *qalter* utility shall permit the
29799 repetition of characters, but shall not assign additional meaning to the repeated
29800 characters. An implementation may define other keep types. The conformance
29801 document for an implementation shall describe any additional keep types, how
29802 they are specified, their internal behavior, and how they affect the behavior of the
29803 utility.

29804 **-I resource_list**

29805 Redefine the resources that are allowed or required by the batch job.

29806 The *qalter* utility shall accept a *resource_list* option-argument that conforms to the
29807 following syntax:

29808 `resource=value[, ,resource=value, , ,]`

29809 The *qalter* utility shall set one entry in the value of the *Resource_List* attribute of the
29810 batch job for each resource listed in the *resource_list* option-argument.

29811 Because the list of supported resource names might vary by batch server, the *qalter*
29812 utility shall rely on the batch server to validate the resource names and associated
29813 values. See Section 3.3.3 (on page 123) for a means of removing *keyword=value* (and
29814 *value@keyword*) pairs and other general rules for list-oriented batch job attributes.

29815 **-m mail_options**

29816 Redefine the points in the execution of the batch job at which the batch server is to
29817 send mail about a change in the state of the batch job.

29818 The *qalter -m* option shall accept a value for the *mail_options* option-argument that
29819 is a string of alphanumeric characters in the portable character set.

29820 The *qalter* utility shall accept a value for the *mail_options* option-argument that is a
29821 string of one or more of the characters 'e', 'b', and 'a', or the single character
29822 'n'. For each unique character in the *mail_options* option-argument, the *qalter*
29823 utility shall add a value to the *Mail_Users* attribute of the batch job as follows, each
29824 representing a different time during the life of a batch job at which to send mail:

- 29825 e MAIL_AT_EXIT

29826 b MAIL_AT_BEGINNING
29827 a MAIL_AT_ABORT
29828 If any of these characters are duplicated in the *mail_options* option-argument, the
29829 duplicates shall be ignored.
29830 An existing *Mail_Points* attribute can be cleared by the mail type:
29831 n NO_MAIL
29832 If 'n' is specified, then mail is not sent. The *qalter* utility shall consider it an error
29833 if any mail type other than 'n' is combined with mail type 'n'. Strictly
29834 conforming applications shall not repeat any of the characters 'e', 'b', 'a', or
29835 'n' within the *mail_options* option-argument. The *qalter* utility shall permit the
29836 repetition of characters but shall not assign additional meaning to the repeated
29837 characters.
29838 An implementation may define other mail types. The conformance document for
29839 an implementation shall describe any additional mail types, how they are
29840 specified, their internal behavior, and how they affect the behavior of the utility.
29841 -M *mail_list* Redefine the list of users to which the batch server that executes the batch job is to
29842 send mail, if the batch server sends mail about the batch job.
29843 The syntax of the *mail_list* option-argument is unspecified. If the implementation
29844 of the *qalter* utility uses a name service to locate users, the utility shall accept the
29845 syntax used by the name service.
29846 If the implementation of the *qalter* utility does not use a name service to locate
29847 users, the implementation shall accept the following syntax for user names:
29848 mail_address[, mail_address, . . .]
29849 The interpretation of *mail_address* is implementation-defined.
29850 The *qalter* utility shall set the *Mail_Users* attribute of the batch job to the value of
29851 the *mail_list* option-argument.
29852 -N *name* Redefine the name of the batch job.
29853 The *qalter* -N option shall accept a value for the *name* option-argument that is a
29854 string of up to 15 alphanumeric characters in the portable character set where the
29855 first character is alphabetic.
29856 The syntax of the *name* option-argument is unspecified.
29857 The *qalter* utility shall set the *Job_Name* attribute of the batch job to the value of the
29858 *name* option-argument.
29859 -o *path_name* Redefine the path for the standard output of the batch job.
29860 The *qalter* utility shall accept a *path_name* option-argument that conforms to the
29861 syntax of the *path_name* element defined in the System Interfaces volume of
29862 IEEE Std 1003.1-2001, which can be preceded by a host name element of the form
29863 *hostname*:.
29864 If the *path_name* option-argument constitutes an absolute pathname, the *qalter*
29865 utility shall set the *Output_Path* attribute of the batch job to the value of the
29866 *path_name* option-argument.
29867

29868 If the *path_name* option-argument constitutes a relative pathname and no host
29869 name element is specified, the *qalter* utility shall set the *Output_Path* attribute of the
29870 batch job to the absolute pathname derived by expanding the *path_name* option-
29871 argument relative to the current directory of the process that executes the *qalter*
29872 utility.

29873 If the *path_name* option-argument constitutes a relative pathname and a host name
29874 element is specified, the *qalter* utility shall set the *Output_Path* attribute of the batch
29875 job to the value of the *path_name* option-argument without any expansion of the
29876 pathname.

29877 If the *path_name* option-argument does not include a host name element, the *qalter*
29878 utility shall prefix the pathname in the *Output_Path* attribute with *hostname*:
29879 where *hostname* is the name of the host upon which the *qalter* utility is being executed.

29880 **-p priority** Redefine the priority of the batch job.
29881 The *qalter* utility shall accept a value for the priority option-argument that
29882 conforms to the syntax for signed decimal integers, and which is not less than
29883 -1 024 and not greater than 1 023.

29884 The *qalter* utility shall set the *Priority* attribute of the batch job to the value of the
29885 *priority* option-argument.

29886 **-r y | n** Redefine whether the batch job is rerunnable.
29887 If the value of the option-argument is 'y', the *qalter* utility shall set the *Rerunable*
29888 attribute of the batch job to TRUE.
29889 If the value of the option-argument is 'n', the *qalter* utility shall set the *Rerunable*
29890 attribute of the batch job to FALSE.
29891 The *qalter* utility shall consider it an error if any character other than 'y' or 'n' is
29892 specified in the option-argument.

29893 **-S path_name_list** Redefine the shell that interprets the script at the destination system.
29894 The *qalter* utility shall accept a *path_name_list* option-argument that conforms to
29895 the following syntax:
29896 *pathname[@host] [, pathname[@host], ...]*
29897 The *qalter* utility shall accept only one pathname that is missing a corresponding
29898 host name. The *qalter* utility shall allow only one pathname per named host.
29899 The *qalter* utility shall add a value to the *Shell_Path_List* attribute of the batch job
29900 for each entry in the *path_name_list* option-argument. See Section 3.3.3 (on page
29901 123) for a means of removing *keyword=value* (and *value@keyword*) pairs and other
29902 general rules for list-oriented batch job attributes.
29903

29904 **-u user_list** Redefine the user name under which the batch job is to run at the destination
29905 system.
29906 The *qalter* utility shall accept a *user_list* option-argument that conforms to the
29907 following syntax:
29908 *username[@host] [, , username[@host], ...]*
29909 The *qalter* utility shall accept only one user name that is missing a corresponding
29910 host name. The *qalter* utility shall accept only one user name per named host.

29911 The *qalter* utility shall add a value to the *User_List* attribute of the batch job for each
29912 entry in the *user_list* option-argument. See Section 3.3.3 (on page 123) for a means
29913 of removing *keyword=value* (and *value@keyword*) pairs and other general rules for
29914 list-oriented batch job attributes.

29915 OPERANDS

29916 The *qalter* utility shall accept one or more operands that conform to the syntax for a batch
29917 *job_identifier* (see Section 3.3.1 (on page 122)).

29918 STDIN

29919 Not used.

29920 INPUT FILES

29921 None.

29922 ENVIRONMENT VARIABLES

29923 The following environment variables shall affect the execution of *qalter*:

29924 *LANG* Provide a default value for the internationalization variables that are unset or null.
29925 (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2,
29926 Internationalization Variables for the precedence of internationalization variables
29927 used to determine the values of locale categories.)

29928 *LC_ALL* If set to a non-empty string value, override the values of all the other
29929 internationalization variables.

29930 *LC_CTYPE* Determine the locale for the interpretation of sequences of bytes of text data as
29931 characters (for example, single-byte as opposed to multi-byte characters in
29932 arguments).

29933 *LC_MESSAGES*

29934 Determine the locale that should be used to affect the format and contents of
29935 diagnostic messages written to standard error.

29936 *LOGNAME* Determine the login name of the user.

29937 *TZ* Determine the timezone used to interpret the *date-time* option-argument. If *TZ* is
29938 unset or null, an unspecified default timezone shall be used.

29939 ASYNCHRONOUS EVENTS

29940 Default.

29941 STDOUT

29942 None.

29943 STDERR

29944 The standard error shall be used only for diagnostic messages.

29945 OUTPUT FILES

29946 None.

29947 EXTENDED DESCRIPTION

29948 None.

29949 EXIT STATUS

29950 The following exit values shall be returned:

29951 0 Successful completion.

29952 >0 An error occurred.

29953 **CONSEQUENCES OF ERRORS**

29954 In addition to the default behavior, the *qalter* utility shall not be required to write a diagnostic
29955 message to standard error when the error reply received from a batch server indicates that the
29956 batch *job_identifier* does not exist on the server. Whether or not the *qalter* utility attempts to
29957 locate the batch job on other batch servers is implementation-defined.

29958 **APPLICATION USAGE**

29959 None.

29960 **EXAMPLES**

29961 None.

29962 **RATIONALE**

29963 The *qalter* utility allows users to change the attributes of a batch job.

29964 As a means of altering a queued job, the *qalter* utility is superior to deleting and requeuing the
29965 batch job insofar as an altered job retains its place in the queue with some traditional selection
29966 algorithms. In addition, the *qalter* utility is both shorter and simpler than a sequence of *qdel* and
29967 *qsub* utilities.

29968 The result of an attempt on the part of a user to alter a batch job in a RUNNING state is
29969 implementation-defined because a batch job in the RUNNING state will already have opened its
29970 output files and otherwise performed any actions indicated by the options in effect at the time
29971 the batch job began execution.

29972 The options processed by the *qalter* utility are identical to those of the *qsub* utility, with a few
29973 exceptions: **-V**, **-v**, and **-q**. The **-V** and **-v** are inappropriate for the *qalter* utility, since they
29974 capture potentially transient environment information from the submitting process. The **-q**
29975 option would specify a new queue, which would largely negate the previously stated advantage
29976 of using *qalter*; furthermore, the *qmove* utility provides a superior means of moving jobs.

29977 Each of the following paragraphs provides the rationale for a *qalter* option.

29978 Additional rationale concerning these options can be found in the rationale for the *qsub* utility.

29979 The **-a** option allows users to alter the date and time at which a batch job becomes eligible to
29980 run.

29981 The **-A** option allows users to change the account that will be charged for the resources
29982 consumed by the batch job. Support for the **-A** option is mandatory for conforming
29983 implementations of *qalter*, even though support of accounting is optional for servers. Whether or
29984 not to support accounting is left to the implementor of the server, but mandatory support of the
29985 **-A** option assures users of a consistent interface and allows them to control accounting on
29986 servers that support accounting.

29987 The **-c** option allows users to alter the checkpointing interval of a batch job. A checkpointing
29988 system, which is not defined by IEEE Std 1003.1-2001, allows recovery of a batch job at the most
29989 recent checkpoint in the event of a crash. Checkpointing is typically used for jobs that consume
29990 expensive computing time or must meet a critical schedule. Users should be allowed to make
29991 the tradeoff between the overhead of checkpointing and the risk to the timely completion of the
29992 batch job; therefore, this volume of IEEE Std 1003.1-2001 provides the checkpointing interval
29993 option. Support for checkpointing is optional for servers.

29994 The **-e** option allows users to alter the name and location of the standard error stream written by
29995 a batch job. However, the path of the standard error stream is meaningless if the value of the
29996 *Join_Path* attribute of the batch job is TRUE.

29997 The **-h** option allows users to set the hold type in the *Hold_Types* attribute of a batch job. The
29998 *qhold* and *qrsl* utilities add or remove hold types to the *Hold_Types* attribute, respectively. The **-h**

29999 option has been modified to allow for implementation-defined hold types.

30000 The **-j** option allows users to alter the decision to join (merge) the standard error stream of the
30001 batch job with the standard output stream of the batch job.

30002 The **-l** option allows users to change the resource limits imposed on a batch job.

30003 The **-m** option allows users to modify the list of points in the life of a batch job at which the
30004 designated users will receive mail notification.

30005 The **-M** option allows users to alter the list of users who will receive notification about events in
30006 the life of a batch job.

30007 The **-N** option allows users to change the name of a batch job.

30008 The **-o** option allows users to alter the name and path to which the standard output stream of
30009 the batch job will be written.

30010 The **-P** option allows users to modify the priority of a batch job. Support for priority is optional
30011 for batch servers.

30012 The **-r** option allows users to alter the rerunability status of a batch job.

30013 The **-S** option allows users to change the name and location of the shell image that will be
30014 invoked to interpret the script of the batch job. This option has been modified to allow a list of
30015 shell name and locations associated with different hosts.

30016 The **-u** option allows users to change the user identifier under which the batch job will execute.

30017 The *job_identifier* operand syntax is provided so that the user can differentiate between the
30018 originating and destination (or executing) batch server. These may or may not be the same. The
30019 *.server_name* portion identifies the originating batch server, while the *@server* portion identifies
30020 the destination batch server.

30021 Historically, the *qalter* utility has been a component of the Network Queuing System (NQS), the
30022 existing practice from which this utility has been derived.

30023 **FUTURE DIRECTIONS**

30024 None.

30025 **SEE ALSO**

30026 Chapter 3 (on page 101), *qdel*, *qhold*, *qmove*, *qrsls*, *qsub*, *touch*

30027 **CHANGE HISTORY**

30028 Derived from IEEE Std 1003.2d-1994.

30029 **Issue 6**

30030 The *TZ* entry is added to the ENVIRONMENT VARIABLES section.

30031 IEEE PASC Interpretation 1003.2 #182 is applied, clarifying the description of the **-a** option.

30032 **NAME**

30033 qdel — delete batch jobs

30034 **SYNOPSIS**

30035 BE qdel *job_identifier* ...

30036

30037 **DESCRIPTION**

30038 A batch job is deleted by sending a request to the batch server that manages the batch job. A
30039 batch job that has been deleted is no longer subject to management by batch services.

30040 The *qdel* utility is a user-accessible client of batch services that requests the deletion of one or
30041 more batch jobs.

30042 The *qdel* utility shall request a batch server to delete those batch jobs for which a batch
30043 *job_identifier* is presented to the utility.

30044 The *qdel* utility shall delete batch jobs in the order in which their batch *job_identifiers* are
30045 presented to the utility.

30046 If the *qdel* utility fails to process any batch *job_identifier* successfully, the utility shall proceed to
30047 process the remaining batch *job_identifiers*, if any.

30048 The *qdel* utility shall delete each batch job by sending a *Delete Job Request* to the batch server that
30049 manages the batch job.

30050 The *qdel* utility shall not exit until the batch job corresponding to each successfully processed
30051 batch *job_identifier* has been deleted.

30052 **OPTIONS**

30053 None.

30054 **OPERANDS**

30055 The *qdel* utility shall accept one or more operands that conform to the syntax for a batch
30056 *job_identifier* (see Section 3.3.1 (on page 122)).

30057 **STDIN**

30058 Not used.

30059 **INPUT FILES**

30060 None.

30061 **ENVIRONMENT VARIABLES**

30062 The following environment variables shall affect the execution of *qdel*:

30063 *LANG* Provide a default value for the internationalization variables that are unset or null.
30064 (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2,
30065 Internationalization Variables for the precedence of internationalization variables
30066 used to determine the values of locale categories.)

30067 *LC_ALL* If set to a non-empty string value, override the values of all the other
30068 internationalization variables.

30069 *LC_CTYPE* Determine the locale for the interpretation of sequences of bytes of text data as
30070 characters (for example, single-byte as opposed to multi-byte characters in
30071 arguments).

30072 *LC_MESSAGES*

30073 Determine the locale that should be used to affect the format and contents of
30074 diagnostic messages written to standard error.

- 30075 **LOGNAME** Determine the login name of the user.
- 30076 **ASYNCHRONOUS EVENTS**
- 30077 Default.
- 30078 **STDOUT**
- 30079 An implementation of the *qdel* utility may write informative messages to standard output.
- 30080 **STDERR**
- 30081 The standard error shall be used only for diagnostic messages.
- 30082 **OUTPUT FILES**
- 30083 None.
- 30084 **EXTENDED DESCRIPTION**
- 30085 None.
- 30086 **EXIT STATUS**
- 30087 The following exit values shall be returned:
- 30088 0 Successful completion.
- 30089 >0 An error occurred.
- 30090 **CONSEQUENCES OF ERRORS**
- 30091 In addition to the default behavior, the *qdel* utility shall not be required to write a diagnostic message to standard error when the error reply received from a batch server indicates that the batch *job_identifier* does not exist on the server. Whether or not the *qdel* utility waits to output the diagnostic message while attempting to locate the job on other servers is implementation-defined.
- 30096 **APPLICATION USAGE**
- 30097 None.
- 30098 **EXAMPLES**
- 30099 None.
- 30100 **RATIONALE**
- 30101 The *qdel* utility allows users and administrators to delete jobs.
- 30102 The *qdel* utility provides functionality that is not otherwise available. For example, the *kill* utility of the operating system does not suffice. First, to use the *kill* utility, the user might have to log in on a remote node, because the *kill* utility does not operate across the network. Second, unlike *qdel*, *kill* cannot remove jobs from queues. Lastly, the arguments of the *qdel* utility are job identifiers rather than process identifiers, and so this utility can be passed the output of the *qselect* utility, thus providing users with a means of deleting a list of jobs.
- 30108 Because a set of jobs can be selected using the *qselect* utility, the *qdel* utility has not been complicated with options that provide for selection of jobs. Instead, the batch jobs to be deleted are identified individually by their job identifiers.
- 30111 Historically, the *qdel* utility has been a component of NQS, the existing practice on which it is based. However, the *qdel* utility defined in this volume of IEEE Std 1003.1-2001 does not provide an option for specifying a signal number to send to the batch job prior to the killing of the process; that capability has been subsumed by the *qsig* utility.
- 30115 A discussion was held about the delays of networking and the possibility that the batch server may never respond, due to a down router, down batch server, or other network mishap. The DESCRIPTION records this under the words “fails to process any job identifier”. In the broad sense, the network problem is also an error, which causes the failure to process the batch job

30119 identifier.

30120 FUTURE DIRECTIONS

30121 None.

30122 SEE ALSO

30123 Chapter 3 (on page 101), *kill*, *qselect*, *qsig*

30124 CHANGE HISTORY

30125 Derived from IEEE Std 1003.2d-1994.

30126 Issue 6

30127 The *LC_TIME* and *TZ* entries are removed from the ENVIRONMENT VARIABLES section.

30128 NAME

30129 qhold — hold batch jobs

30130 SYNOPSIS

30131 BE qhold [-h *hold_list*] *job_identifier* ...

30132

30133 DESCRIPTION

30134 A hold is placed on a batch job by a request to the batch server that manages the batch job. A
30135 batch job that has one or more holds is not eligible for execution. The *qhold* utility is a user-
30136 accessible client of batch services that requests one or more types of hold to be placed on one or
30137 more batch jobs.

30138 The *qhold* utility shall place holds on those batch jobs for which a batch *job_identifier* is presented
30139 to the utility.

30140 The *qhold* utility shall place holds on batch jobs in the order in which their batch *job_identifiers*
30141 are presented to the utility. If the *qhold* utility fails to process any batch *job_identifier* successfully,
30142 the utility shall proceed to process the remaining batch *job_identifiers*, if any.

30143 The *qhold* utility shall place holds on each batch job by sending a *Hold Job Request* to the batch
30144 server that manages the batch job.

30145 The *qhold* utility shall not exit until holds have been placed on the batch job corresponding to
30146 each successfully processed batch *job_identifier*.

30147 OPTIONS

30148 The *qhold* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section
30149 12.2, Utility Syntax Guidelines.

30150 The following option shall be supported by the implementation:

30151 **-h *hold_list*** Define the types of holds to be placed on the batch job.

30152 The *qhold* **-h** option shall accept a value for the *hold_list* option-argument that is a
30153 string of alphanumeric characters in the portable character set (see the Base
30154 Definitions volume of IEEE Std 1003.1-2001, Section 6.1, Portable Character Set).

30155 The *qhold* utility shall accept a value for the *hold_list* option-argument that is a
30156 string of one or more of the characters 'u', 's', or 'o', or the single character
30157 'n'.

30158 For each unique character in the *hold_list* option-argument, the *qhold* utility shall
30159 add a value to the *Hold_Types* attribute of the batch job as follows, each
30160 representing a different hold type:

30161 u USER

30162 s SYSTEM

30163 o OPERATOR

30164 If any of these characters are duplicated in the *hold_list* option-argument, the
30165 duplicates shall be ignored.

30166 An existing *Hold_Types* attribute can be cleared by the following hold type:

30167 n NO_HOLD

30168 The *qhold* utility shall consider it an error if any hold type other than 'n' is
30169 combined with hold type 'n'.

30170 Strictly conforming applications shall not repeat any of the characters 'u', 's',
30171 'o', or 'n' within the *hold_list* option-argument. The *qhold* utility shall permit the
30172 repetition of characters, but shall not assign additional meaning to the repeated
30173 characters.

30174 An implementation may define other hold types. The conformance document for
30175 an implementation shall describe any additional hold types, how they are
30176 specified, their internal behavior, and how they affect the behavior of the utility.

30177 If the **-h** option is not presented to the *qhold* utility, the implementation shall set
30178 the *Hold_Types* attribute to USER.

30179 OPERANDS

30180 The *qhold* utility shall accept one or more operands that conform to the syntax for a batch
30181 *job_identifier* (see Section 3.3.1 (on page 122)).

30182 STDIN

30183 Not used.

30184 INPUT FILES

30185 None.

30186 ENVIRONMENT VARIABLES

30187 The following environment variables shall affect the execution of *qhold*:

30188 *LANG* Provide a default value for the internationalization variables that are unset or null.
30189 (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2,
30190 Internationalization Variables for the precedence of internationalization variables
30191 used to determine the values of locale categories.)

30192 *LC_ALL* If set to a non-empty string value, override the values of all the other
30193 internationalization variables.

30194 *LC_CTYPE* Determine the locale for the interpretation of sequences of bytes of text data as
30195 characters (for example, single-byte as opposed to multi-byte characters in
30196 arguments).

30197 *LC_MESSAGES*

30198 Determine the locale that should be used to affect the format and contents of
30199 diagnostic messages written to standard error.

30200 *LOGNAME* Determine the login name of the user.

30201 ASYNCHRONOUS EVENTS

30202 Default.

30203 STDOUT

30204 None.

30205 STDERR

30206 The standard error shall be used only for diagnostic messages.

30207 OUTPUT FILES

30208 None.

30209 EXTENDED DESCRIPTION

30210 None.

30211 EXIT STATUS

30212 The following exit values shall be returned:
30213 0 Successful completion.
30214 >0 An error occurred.

30215 CONSEQUENCES OF ERRORS

30216 In addition to the default behavior, the *qhold* utility shall not be required to write a diagnostic
30217 message to standard error when the error reply received from a batch server indicates that the
30218 batch *job_identifier* does not exist on the server. Whether or not the *qhold* utility waits to output
30219 the diagnostic message while attempting to locate the job on other servers is implementation-
30220 defined.

30221 APPLICATION USAGE

30222 None.

30223 EXAMPLES

30224 None.

30225 RATIONALE

30226 The *qhold* utility allows users to place a hold on one or more jobs. A hold makes a batch job
30227 ineligible for execution.

30228 The *qhold* utility has options that allow the user to specify the type of hold. Should the user wish
30229 to place a hold on a set of jobs that meet a selection criteria, such a list of jobs can be acquired
30230 using the *qselect* utility.

30231 The **-h** option allows the user to specify the type of hold that is to be placed on the job. This
30232 option allows for USER, SYSTEM, OPERATOR, and implementation-defined hold types. The
30233 USER and OPERATOR holds are distinct. The batch server that manages the batch job will verify
30234 that the user is authorized to set the specified hold for the batch job.

30235 Mail is not required on hold because the administrator has the tools and libraries to build this
30236 option if he or she wishes.

30237 Historically, the *qhold* utility has been a part of some existing batch systems, although it has not
30238 traditionally been a part of the NQS.

30239 FUTURE DIRECTIONS

30240 None.

30241 SEE ALSO

30242 Chapter 3 (on page 101), *qselect*

30243 CHANGE HISTORY

30244 Derived from IEEE Std 1003.2d-1994.

30245 Issue 6

30246 The *LC_TIME* and *TZ* entries are removed from the ENVIRONMENT VARIABLES section.

30247 NAME

30248 qmove — move batch jobs

30249 SYNOPSIS

30250 BE qmove destination job_identifier ...

30251

30252 DESCRIPTION

30253 To move a batch job is to remove the batch job from the batch queue in which it resides and
30254 instantiate the batch job in another batch queue. A batch job is moved by a request to the batch
30255 server that manages the batch job. The *qmove* utility is a user-accessible batch client that requests
30256 the movement of one or more batch jobs.

30257 The *qmove* utility shall move those batch jobs, and only those batch jobs, for which a batch
30258 *job_identifier* is presented to the utility.

30259 The *qmove* utility shall move batch jobs in the order in which the corresponding batch
30260 *job_identifiers* are presented to the utility.

30261 If the *qmove* utility fails to process a batch *job_identifier* successfully, the utility shall proceed to
30262 process the remaining batch *job_identifiers*, if any.

30263 The *qmove* utility shall move batch jobs by sending a *Move Job Request* to the batch server that
30264 manages each batch job. The *qmove* utility shall not exit before the batch jobs corresponding to all
30265 successfully processed batch *job_identifiers* have been moved.

30266 OPTIONS

30267 None.

30268 OPERANDS

30269 The *qmove* utility shall accept one operand that conforms to the syntax for a destination (see
30270 Section 3.3.2 (on page 123)).

30271 The *qmove* utility shall accept one or more operands that conform to the syntax for a batch
30272 *job_identifier* (see Section 3.3.1 (on page 122)).

30273 STDIN

30274 Not used.

30275 INPUT FILES

30276 None.

30277 ENVIRONMENT VARIABLES

30278 The following environment variables shall affect the execution of *qmove*:

30279 **LANG** Provide a default value for the internationalization variables that are unset or null.
30280 (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2,
30281 Internationalization Variables for the precedence of internationalization variables
30282 used to determine the values of locale categories.)

30283 **LC_ALL** If set to a non-empty string value, override the values of all the other
30284 internationalization variables.

30285 **LC_CTYPE** Determine the locale for the interpretation of sequences of bytes of text data as
30286 characters (for example, single-byte as opposed to multi-byte characters in
30287 arguments).

30288 LC_MESSAGES

30289 Determine the locale that should be used to affect the format and contents of
30290 diagnostic messages written to standard error.

30291 **LOGNAME** Determine the login name of the user.

30292 **ASYNCHRONOUS EVENTS**

30293 Default.

30294 **STDOUT**

30295 None.

30296 **STDERR**

30297 The standard error shall be used only for diagnostic messages.

30298 **OUTPUT FILES**

30299 None.

30300 **EXTENDED DESCRIPTION**

30301 None.

30302 **EXIT STATUS**

30303 The following exit values shall be returned:

30304 0 Successful completion.

30305 >0 An error occurred.

30306 **CONSEQUENCES OF ERRORS**

30307 In addition to the default behavior, the *qmove* utility shall not be required to write a diagnostic message to standard error when the error reply received from a batch server indicates that the batch *job_identifier* does not exist on the server. Whether or not the *qmove* utility waits to output the diagnostic message while attempting to locate the job on other servers is implementation-defined.

30312 **APPLICATION USAGE**

30313 None.

30314 **EXAMPLES**

30315 None.

30316 **RATIONALE**

30317 The *qmove* utility allows users to move jobs between queues.

30318 The alternative to using the *qmove* utility—deleting the batch job and requeueing it—entails considerably more typing.

30320 Since the means of selecting jobs based on attributes has been encapsulated in the *qselect* utility, the only option of the *qmove* utility concerns authorization. The **-u** option provides the user with the convenience of changing the user identifier under which the batch job will execute. Minimalism and consistency have taken precedence over convenience; the **-u** option has been deleted because the equivalent capability exists with the **-u** option of the *qalter* utility.

30325 **FUTURE DIRECTIONS**

30326 None.

30327 **SEE ALSO**

30328 Chapter 3 (on page 101), *qalter*, *qselect*

30329 **CHANGE HISTORY**

30330 Derived from IEEE Std 1003.2d-1994.

30331 **Issue 6**30332 The *LC_TIME* and *TZ* entries are removed from the ENVIRONMENT VARIABLES section.

30333 NAME

30334 qmsg — send message to batch jobs

30335 SYNOPSIS

30336 BE qmsg [-E][-O] *message_string job_identifier ...*

30337

30338 DESCRIPTION

30339 To send a message to a batch job is to request that a server write a message string into one or
30340 more output files of the batch job. A message is sent to a batch job by a request to the batch
30341 server that manages the batch job. The *qmsg* utility is a user-accessible batch client that requests
30342 the sending of messages to one or more batch jobs.

30343 The *qmsg* utility shall write messages into the files of batch jobs by sending a *Job Message Request*
30344 to the batch server that manages the batch job. The *qmsg* utility shall not directly write the
30345 message into the files of the batch job.

30346 The *qmsg* utility shall send a *Job Message Request* for those batch jobs, and only those batch jobs,
30347 for which a batch *job_identifier* is presented to the utility.

30348 The *qmsg* utility shall send *Job Message Requests* for batch jobs in the order in which their batch
30349 *job_identifiers* are presented to the utility.

30350 If the *qmsg* utility fails to process any batch *job_identifier* successfully, the utility shall proceed to
30351 process the remaining batch *job_identifiers*, if any.

30352 The *qmsg* utility shall not exit before a *Job Message Request* has been sent to the server that
30353 manages the batch job that corresponds to each successfully processed batch *job_identifier*.

30354 OPTIONS

30355 The *qmsg* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section
30356 12.2, Utility Syntax Guidelines.

30357 The following options shall be supported by the implementation:

30358 -E Specify that the message is written to the standard error of each batch job.

30359 The *qmsg* utility shall write the message into the standard error of the batch job.

30360 -O Specify that the message is written to the standard output of each batch job.

30361 The *qmsg* utility shall write the message into the standard output of the batch job.

30362 If neither the -O nor the -E option is presented to the *qmsg* utility, the utility shall write the
30363 message into an implementation-defined file. The conformance document for the
30364 implementation shall describe the name and location of the implementation-defined file. If both
30365 the -O and the -E options are presented to the *qmsg* utility, then the utility shall write the
30366 messages to both standard output and standard error.

30367 OPERANDS

30368 The *qmsg* utility shall accept a minimum of two operands, *message_string* and one or more batch
30369 *job_identifiers*.

30370 The *message_string* operand shall be the string to be written to one or more output files of the
30371 batch job followed by a <newline>. If the string contains <blank>s, then the application shall
30372 ensure that the string is quoted. The *message_string* shall be encoded in the portable character set
30373 (see the Base Definitions volume of IEEE Std 1003.1-2001, Section 6.1, Portable Character Set).

30374 All remaining operands are batch *job_identifiers* that conform to the syntax for a batch
30375 *job_identifier* (see Section 3.3.1 (on page 122)).

30376 STDIN

30377 Not used.

30378 INPUT FILES

30379 None.

30380 ENVIRONMENT VARIABLES

30381 The following environment variables shall affect the execution of *qmsg*:

30382 *LANG* Provide a default value for the internationalization variables that are unset or null.
(See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2, Internationalization Variables for the precedence of internationalization variables used to determine the values of locale categories.)

30386 *LC_ALL* If set to a non-empty string value, override the values of all the other internationalization variables.

30388 *LC_CTYPE* Determine the locale for the interpretation of sequences of bytes of text data as characters (for example, single-byte as opposed to multi-byte characters in arguments).

30391 *LC_MESSAGES*

30392 Determine the locale that should be used to affect the format and contents of diagnostic messages written to standard error.

30394 *LOGNAME* Determine the login name of the user.

30395 ASYNCHRONOUS EVENTS

30396 Default.

30397 STDOUT

30398 None.

30399 STDERR

30400 The standard error shall be used only for diagnostic messages.

30401 OUTPUT FILES

30402 None.

30403 EXTENDED DESCRIPTION

30404 None.

30405 EXIT STATUS

30406 The following exit values shall be returned:

30407 0 Successful completion.

30408 >0 An error occurred.

30409 CONSEQUENCES OF ERRORS

30410 In addition to the default behavior, the *qmsg* utility shall not be required to write a diagnostic message to standard error when the error reply received from a batch server indicates that the batch *job_identifier* does not exist on the server. Whether or not the *qmsg* utility waits to output the diagnostic message while attempting to locate the job on other servers is implementation-defined.

30415 APPLICATION USAGE

30416 None.

30417 EXAMPLES

30418 None.

30419 RATIONALE

30420 The *qmsg* utility allows users to write messages into the output files of running jobs. Users, 30421 including operators and administrators, have a number of occasions when they want to place 30422 messages in the output files of a batch job. For example, if a disk that is being used by a batch job 30423 is showing errors, the operator might note this in the standard error stream of the batch job.

30424 The options of the *qmsg* utility provide users with the means of placing the message in the 30425 output stream of their choice. The default output stream for the message—if the user does not 30426 designate an output stream—is implementation-defined, since many implementations will 30427 provide, as an extension to this volume of IEEE Std 1003.1-2001, a log file that shows the history 30428 of utility execution.

30429 If users wish to send a message to a set of jobs that meet a selection criteria, the *qselect* utility can 30430 be used to acquire the appropriate list of job identifiers.

30431 The **–E** option allows users to place the message in the standard error stream of the batch job.

30432 The **–O** option allows users to place the message in the standard output stream of the batch job.

30433 Historically, the *qmsg* utility is an existing practice in the offerings of one or more implementors 30434 of an NQS-derived batch system. The utility has been found to be useful enough that it deserves 30435 to be included in this volume of IEEE Std 1003.1-2001.

30436 FUTURE DIRECTIONS

30437 None.

30438 SEE ALSO

30439 Chapter 3 (on page 101), *qselect*

30440 CHANGE HISTORY

30441 Derived from IEEE Std 1003.2d-1994.

30442 Issue 6

30443 The *LC_TIME* and *TZ* entries are removed from the ENVIRONMENT VARIABLES section.

30444 NAME

30445 qrerun — rerun batch jobs

30446 SYNOPSIS

30447 BE qrerun *job_identifier* ...

30448

30449 DESCRIPTION

30450 To rerun a batch job is to terminate the session leader of the batch job, delete any associated
30451 checkpoint files, and return the batch job to the batch queued state. A batch job is rerun by a
30452 request to the batch server that manages the batch job. The *qrerun* utility is a user-accessible
30453 batch client that requests the rerunning of one or more batch jobs.

30454 The *qrerun* utility shall rerun those batch jobs for which a batch *job_identifier* is presented to the
30455 utility.

30456 The *qrerun* utility shall rerun batch jobs in the order in which their batch *job_identifiers* are
30457 presented to the utility.

30458 If the *qrerun* utility fails to process any batch *job_identifier* successfully, the utility shall proceed
30459 to process the remaining batch *job_identifiers*, if any.

30460 The *qrerun* utility shall rerun batch jobs by sending a *Rerun Job Request* to the batch server that
30461 manages each batch job.

30462 For each successfully processed batch *job_identifier*, the *qrerun* utility shall have rerun the
30463 corresponding batch job at the time the utility exits.

30464 OPTIONS

30465 None.

30466 OPERANDS

30467 The *qrerun* utility shall accept one or more operands that conform to the syntax for a batch
30468 *job_identifier* (see Section 3.3.1 (on page 122)).

30469 STDIN

30470 Not used.

30471 INPUT FILES

30472 None.

30473 ENVIRONMENT VARIABLES

30474 The following environment variables shall affect the execution of *qrerun*:

30475 *LANG* Provide a default value for the internationalization variables that are unset or null.
30476 (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2,
30477 Internationalization Variables for the precedence of internationalization variables
30478 used to determine the values of locale categories.)

30479 *LC_ALL* If set to a non-empty string value, override the values of all the other
30480 internationalization variables.

30481 *LC_CTYPE* Determine the locale for the interpretation of sequences of bytes of text data as
30482 characters (for example, single-byte as opposed to multi-byte characters in
30483 arguments).

30484 *LC_MESSAGES*

30485 Determine the locale that should be used to affect the format and contents of
30486 diagnostic messages written to standard error.

30487 **LOGNAME** Determine the login name of the user.

30488 ASYNCHRONOUS EVENTS

30489 Default.

30490 STDOUT

30491 None.

30492 STDERR

30493 The standard error shall be used only for diagnostic messages.

30494 OUTPUT FILES

30495 None.

30496 EXTENDED DESCRIPTION

30497 None.

30498 EXIT STATUS

30499 The following exit values shall be returned:

30500 0 Successful completion.

30501 >0 An error occurred.

30502 CONSEQUENCES OF ERRORS

30503 In addition to the default behavior, the *qrerun* utility shall not be required to write a diagnostic message to standard error when the error reply received from a batch server indicates that the batch *job_identifier* does not exist on the server. Whether or not the *qrerun* utility waits to output the diagnostic message while attempting to locate the job on other servers is implementation-defined.

30508 APPLICATION USAGE

30509 None.

30510 EXAMPLES

30511 None.

30512 RATIONALE

30513 The *qrerun* utility allows users to cause jobs in the running state to exit and rerun.

30514 The *qrerun* utility is a new utility, *vis-a-vis* existing practice, that has been defined in this volume of IEEE Std 1003.1-2001 to correct user-perceived deficiencies in the existing practice.

30516 FUTURE DIRECTIONS

30517 None.

30518 SEE ALSO

30519 Chapter 3 (on page 101)

30520 CHANGE HISTORY

30521 Derived from IEEE Std 1003.2d-1994.

30522 Issue 6

30523 The *LC_TIME* and *TZ* entries are removed from the ENVIRONMENT VARIABLES section.

30524 NAME

30525 qrsls — release batch jobs

30526 SYNOPSIS

30527 BE qrsls [-h hold_list] job_identifier ...

30528

30529 DESCRIPTION

30530 A batch job might have one or more holds, which prevent the batch job from executing. A batch
30531 job from which all the holds have been removed becomes eligible for execution and is said to
30532 have been released. A batch job hold is removed by sending a request to the batch server that
30533 manages the batch job. The *qrsls* utility is a user-accessible client of batch services that requests
30534 holds be removed from one or more batch jobs.

30535 The *qrsls* utility shall remove one or more holds from those batch jobs for which a batch
30536 *job_identifier* is presented to the utility.

30537 The *qrsls* utility shall remove holds from batch jobs in the order in which their batch *job_identifiers*
30538 are presented to the utility.

30539 If the *qrsls* utility fails to process a batch *job_identifier* successfully, the utility shall proceed to
30540 process the remaining batch *job_identifiers*, if any.

30541 The *qrsls* utility shall remove holds on each batch job by sending a *Release Job Request* to the batch
30542 server that manages the batch job.

30543 The *qrsls* utility shall not exit until the holds have been removed from the batch job
30544 corresponding to each successfully processed batch *job_identifier*.

30545 OPTIONS

30546 The *qrsls* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section
30547 12.2, Utility Syntax Guidelines.

30548 The following option shall be supported by the implementation:

30549 **-h hold_list** Define the types of holds to be removed from the batch job.

30550 The *qrsls -h* option shall accept a value for the *hold_list* option-argument that is a
30551 string of alphanumeric characters in the portable character set (see the Base
30552 Definitions volume of IEEE Std 1003.1-2001, Section 6.1, Portable Character Set).

30553 The *qrsls* utility shall accept a value for the *hold_list* option-argument that is a string
30554 of one or more of the characters 'u', 's', or 'o', or the single character 'n'.

30555 For each unique character in the *hold_list* option-argument, the *qrsls* utility shall add
30556 a value to the *Hold_Types* attribute of the batch job as follows, each representing a
30557 different hold type:

30558 u USER

30559 s SYSTEM

30560 o OPERATOR

30561 If any of these characters are duplicated in the *hold_list* option-argument, the
30562 duplicates shall be ignored.

30563 An existing *Hold_Types* attribute can be cleared by the following hold type:

30564 n NO_HOLD

30565 The *qrsls* utility shall consider it an error if any hold type other than 'n' is
30566 combined with hold type 'n'.

30567 Strictly conforming applications shall not repeat any of the characters 'u', 's',
30568 'o', or 'n' within the *hold_list* option-argument. The *qrsls* utility shall permit the
30569 repetition of characters, but shall not assign additional meaning to the repeated
30570 characters.

30571 An implementation may define other hold types. The conformance document for
30572 an implementation shall describe any additional hold types, how they are
30573 specified, their internal behavior, and how they affect the behavior of the utility.

30574 If the **-h** option is not presented to the *qrsls* utility, the implementation shall remove
30575 the USER hold in the *Hold_Types* attribute.

30576 OPERANDS

30577 The *qrsls* utility shall accept one or more operands that conform to the syntax for a batch
30578 *job_identifier* (see Section 3.3.1 (on page 122)).

30579 STDIN

30580 Not used.

30581 INPUT FILES

30582 None.

30583 ENVIRONMENT VARIABLES

30584 The following environment variables shall affect the execution of *qrsls*:

30585 *LANG* Provide a default value for the internationalization variables that are unset or null.
30586 (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2,
30587 Internationalization Variables for the precedence of internationalization variables
30588 used to determine the values of locale categories.)

30589 *LC_ALL* If set to a non-empty string value, override the values of all the other
30590 internationalization variables.

30591 *LC_CTYPE* Determine the locale for the interpretation of sequences of bytes of text data as
30592 characters (for example, single-byte as opposed to multi-byte characters in
30593 arguments).

30594 *LC_MESSAGES*

30595 Determine the locale that should be used to affect the format and contents of
30596 diagnostic messages written to standard error.

30597 *LOGNAME* Determine the login name of the user.

30598 ASYNCHRONOUS EVENTS

30599 Default.

30600 STDOUT

30601 None.

30602 STDERR

30603 The standard error shall be used only for diagnostic messages.

30604 OUTPUT FILES

30605 None.

30606 EXTENDED DESCRIPTION

30607 None.

30608 EXIT STATUS

30609 The following exit values shall be returned:

30610 0 Successful completion.

30611 >0 An error occurred.

30612 CONSEQUENCES OF ERRORS

30613 In addition to the default behavior, the *qrfs* utility shall not be required to write a diagnostic message to standard error when the error reply received from a batch server indicates that the batch *job_identifier* does not exist on the server. Whether or not the *qrfs* utility waits to output the diagnostic message while attempting to locate the job on other servers is implementation-defined.

30618 APPLICATION USAGE

30619 None.

30620 EXAMPLES

30621 None.

30622 RATIONALE

30623 The *qrfs* utility allows users, operators, and administrators to remove holds from jobs.

30624 The *qrfs* utility does not support any job selection options or wildcard arguments. Users may acquire a list of jobs selected by attributes using the *qselect* utility. For example, a user could select all of their held jobs.

30627 The **-h** option allows the user to specify the type of hold that is to be removed. This option allows for USER, SYSTEM, OPERATOR, and implementation-defined hold types. The batch server that manages the batch job will verify whether the user is authorized to remove the specified hold for the batch job. If more than one type of hold has been placed on the batch job, a user may wish to remove only some of them.

30632 Mail is not required on release because the administrator has the tools and libraries to build this option if required.

30634 The *qrfs* utility is a new utility *vis-a-vis* existing practice; it has been defined in this volume of IEEE Std 1003.1-2001 as the natural complement to the *qhold* utility.

30636 FUTURE DIRECTIONS

30637 None.

30638 SEE ALSO

30639 Chapter 3 (on page 101), *qhold*, *qselect*

30640 CHANGE HISTORY

30641 Derived from IEEE Std 1003.2d-1994.

30642 Issue 6

30643 The *LC_TIME* and *TZ* entries are removed from the ENVIRONMENT VARIABLES section.

30644 NAME

30645 qselect — select batch jobs

30646 SYNOPSIS

30647 BE qselect [-a [op]date_time][-A account_string][-c [op]interval]
 30648 [-h hold_list][-l resource_list][-N name][-p [op]priority]
 30649 [-q destination][-r y|n][-s states][-u user_list]

30650

30651 DESCRIPTION

30652 To select a set of batch jobs is to return the batch *job_identifiers* for each batch job that meets a list
 30653 of selection criteria. A set of batch jobs is selected by a request to a batch server. The *qselect*
 30654 utility is a user-accessible batch client that requests the selection of batch jobs.

30655 Upon successful completion, the *qselect* utility shall have returned a list of zero or more batch
 30656 *job_identifiers* that meet the criteria specified by the options and option-arguments presented to
 30657 the utility.

30658 The *qselect* utility shall select batch jobs by sending a *Select Jobs Request* to a batch server. The
 30659 *qselect* utility shall not exit until the server replies to each request generated.

30660 For each option presented to the *qselect* utility, the utility shall restrict the set of selected batch
 30661 jobs as described in the OPTIONS section.

30662 The *qselect* utility shall not restrict selection of batch jobs except by authorization and as required
 30663 by the options presented to the utility.

30664 When an option is specified with a mandatory or optional *op* component to the option-
 30665 argument, then *op* shall specify a relation between the value of a certain batch job attribute and
 30666 the *value* component of the option-argument. If an *op* is allowable on an option, then the
 30667 description of the option letter indicates the *op* as either mandatory or optional. Acceptable
 30668 strings for the *op* component, and the relation the string indicates, are shown in the following
 30669 list:

- 30670 .eq. The value represented by the attribute of the batch job is equal to the value represented
 30671 by the option-argument.
- 30672 .ge. The value represented by the attribute of the batch job is greater than or equal to the
 30673 value represented by the option-argument.
- 30674 .gt. The value represented by the attribute of the batch job is greater than the value
 30675 represented by the option-argument.
- 30676 .lt. The value represented by the attribute of the batch job is less than the value
 30677 represented by the option-argument.
- 30678 .le. The value represented by the attribute of the batch job is less than or equal to the value
 30679 represented by the option-argument.
- 30680 .ne. The value represented by the attribute of the batch job is not equal to the value
 30681 represented by the option-argument.

30682 OPTIONS

30683 The *qselect* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section
 30684 12.2, Utility Syntax Guidelines.

30685 The following options shall be supported by the implementation:

30686 -a [op]date_time
 30687 Restrict selection to a specific time, or a range of times.

30688 The *qselect* utility shall select only batch jobs for which the value of the
30689 *Execution_Time* attribute is related to the Epoch equivalent of the local time
30690 expressed by the value of the *date_time* component of the option-argument in the
30691 manner indicated by the value of the *op* component of the option-argument.

30692 The *qselect* utility shall accept a *date_time* component of the option-argument that
30693 conforms to the syntax of the *time* operand of the *touch* utility.

30694 If the *op* component of the option-argument is not presented to the *qselect* utility,
30695 the utility shall select batch jobs for which the *Execution_Time* attribute is equal to
30696 the *date_time* component of the option-argument.

30697 When comparing times, the *qselect* utility shall use the following definitions for the
30698 *op* component of the option-argument:

- 30699 .eq. The time represented by value of the *Execution_Time* attribute of the batch
30700 job is equal to the time represented by the *date_time* component of the
30701 option-argument.
- 30702 .ge. The time represented by value of the *Execution_Time* attribute of the batch
30703 job is after or equal to the time represented by the *date_time* component of
30704 the option-argument.
- 30705 .gt. The time represented by value of the *Execution_Time* attribute of the batch
30706 job is after the time represented by the *date_time* component of the
30707 option-argument.
- 30708 .lt. The time represented by value of the *Execution_Time* attribute of the batch
30709 job is before the time represented by the *date_time* component of the
30710 option-argument.
- 30711 .le. The time represented by value of the *Execution_Time* attribute of the batch
30712 job is before or equal to the time represented by the *date_time* component of
30713 the option-argument.
- 30714 .ne. The time represented by value of the *Execution_Time* attribute of the batch
30715 job is not equal to the time represented by the *date_time* component of the
30716 option-argument.

30717 The *qselect* utility shall accept the defined character strings for the *op* component of
30718 the option-argument.

30719 **-A account_string**

30720 Restrict selection to the batch jobs charging a specified account.

30721 The *qselect* utility shall select only batch jobs for which the value of the
30722 *Account_Name* attribute of the batch job matches the value of the *account_string*
30723 option-argument.

30724 The syntax of the *account_string* option-argument is unspecified.

30725 **-c [op]interval**

30726 Restrict selection to batch jobs within a range of checkpoint intervals.

30727 The *qselect* utility shall select only batch jobs for which the value of the *CHECKPOINT*
30728 attribute relates to the value of the *interval* component of the option-argument in
30729 the manner indicated by the value of the *op* component of the option-argument.

30730 If the *op* component of the option-argument is omitted, the *qselect* utility shall
30731 select batch jobs for which the value of the *CHECKPOINT* attribute is equal to the value

- 30732 of the *interval* component of the option-argument.
- 30733 When comparing checkpoint intervals, the *qselect* utility shall use the following
30734 definitions for the *op* component of the option-argument:
- 30735 .eq. The value of the *Clockpoint* attribute of the batch job equals the value of
30736 the *interval* component of the option-argument.
 - 30737 .ge. The value of the *Clockpoint* attribute of the batch job is greater than or
30738 equal to the value of the *interval* component option-argument.
 - 30739 .gt. The value of the *Clockpoint* attribute of the batch job is greater than the
30740 value of the *interval* component option-argument.
 - 30741 .lt. The value of the *Clockpoint* attribute of the batch job is less than the value
30742 of the *interval* component option-argument.
 - 30743 .le. The value of the *Clockpoint* attribute of the batch job is less than or equal
30744 to the value of the *interval* component option-argument.
 - 30745 .ne. The value of the *Clockpoint* attribute of the batch job does not equal the
30746 value of the *interval* component option-argument.
- 30747 The *qselect* utility shall accept the defined character strings for the *op* component of
30748 the option-argument.
- 30749 The ordering relationship for the values of the interval option-argument is defined
30750 to be:
- 30751 'n' .gt. 's' .gt. 'c=minutes' .ge. 'c'
- 30752 When comparing *Clockpoint* attributes with an interval having the value of the
30753 single character 'u', only equality or inequality are valid comparisons.
- 30754 **-h hold_list** Restrict selection to batch jobs that have a specific type of hold.
- 30755 The *qselect* utility shall select only batch jobs for which the value of the *Hold_Types*
30756 attribute matches the value of the *hold_list* option-argument.
- 30757 The *qselect* -h option shall accept a value for the *hold_list* option-argument that is a
30758 string of alphanumeric characters in the portable character set (see the Base
30759 Definitions volume of IEEE Std 1003.1-2001, Section 6.1, Portable Character Set).
- 30760 The *qselect* utility shall accept a value for the *hold_list* option-argument that is a
30761 string of one or more of the characters 'u', 's', or 'o', or the single character
30762 'n'.
- 30763 Each unique character in the *hold_list* option-argument of the *qselect* utility is
30764 defined as follows, each representing a different hold type:
- 30765 u USER
- 30766 s SYSTEM
- 30767 o OPERATOR
- 30768 If any of these characters are duplicated in the *hold_list* option-argument, the
30769 duplicates shall be ignored.
- 30770 The *qselect* utility shall consider it an error if any hold type other than 'n' is
30771 combined with hold type 'n'.

30772 Strictly conforming applications shall not repeat any of the characters 'u', 's',
30773 'o', or 'n' within the *hold_list* option-argument. The *qselect* utility shall permit
30774 the repetition of characters, but shall not assign additional meaning to the repeated
30775 characters.

30776 An implementation may define other hold types. The conformance document for
30777 an implementation shall describe any additional hold types, how they are
30778 specified, their internal behavior, and how they affect the behavior of the utility.

30779 **-l resource_list**

30780 Restrict selection to batch jobs with specified resource limits and attributes.

30781 The *qselect* utility shall accept a *resource_list* option-argument with the following
30782 syntax:

30783 *resource_name op value [,,resource_name op value,, ...]*

30784 When comparing resource values, the *qselect* utility shall use the following
30785 definitions for the *op* component of the option-argument:

- 30786 .eq. The value of the resource of the same name in the *Resource_List* attribute
30787 of the batch job equals the value of the *value* component of the option-
30788 argument.
- 30789 .ge. The value of the resource of the same name in the *Resource_List* attribute
30790 of the batch job is greater than or equal to the value of the *value*
30791 component of the option-argument.
- 30792 .gt. The value of the resource of the same name in the *Resource_List* attribute
30793 of the batch job is greater than the value of the *value* component of the
30794 option-argument.
- 30795 .lt. The value of the resource of the same name in the *Resource_List* attribute
30796 of the batch job is less than the value of the *value* component of the
30797 option-argument.
- 30798 .ne. The value of the resource of the same name in the *Resource_List* attribute
30799 of the batch job does not equal the value of the *value* component of the
30800 option-argument.
- 30801 .le. The value of the resource of the same name in the *Resource_List* attribute
30802 of the batch job is less than or equal to the value of the *value* component
30803 of the option-argument.

30804 When comparing the limit of a *Resource_List* attribute with the *value* component of
30805 the option-argument, if the limit, the value, or both are non-numeric, only equality
30806 or inequality are valid comparisons.

30807 The *qselect* utility shall select only batch jobs for which the values of the
30808 *resource_names* listed in the *resource_list* option-argument match the corresponding
30809 limits of the *Resource_List* attribute of the batch job.

30810 Limits of *resource_names* present in the *Resource_List* attribute of the batch job that
30811 have no corresponding values in the *resource_list* option-argument shall not be
30812 considered when selecting batch jobs.

30813 **-N name** Restrict selection to batch jobs with a specified name.

30814 The *qselect* utility shall select only batch jobs for which the value of the *Job_Name*
30815 attribute matches the value of the *name* option-argument. The string specified in

30816 the *name* option-argument shall be passed, uninterpreted, to the server. This allows
30817 an implementation to match “wildcard” patterns against batch job names.

30818 An implementation shall describe in the conformance document the format it
30819 supports for matching against the *Job_Name* attribute.

30820 **-p [op]priority**

30821 Restrict selection to batch jobs of the specified priority or range of priorities.

30822 The *qselect* utility shall select only batch jobs for which the value of the *Priority*
30823 attribute of the batch job relates to the value of the *priority* component of the
30824 option-argument in the manner indicated by the value of the *op* component of the
30825 option-argument.

30826 If the *op* component of the option-argument is omitted, the *qselect* utility shall
30827 select batch jobs for which the value of the *Priority* attribute of the batch job is
30828 equal to the value of the *priority* component of the option-argument.

30829 When comparing priority values, the *qselect* utility shall use the following
30830 definitions for the *op* component of the option-argument:

- 30831 .eq. The value of the *Priority* attribute of the batch job equals the value of the
30832 *priority* component of the option-argument.
- 30833 .ge. The value of the *Priority* attribute of the batch job is greater than or equal
30834 to the value of the *priority* component option-argument.
- 30835 .gt. The value of the *Priority* attribute of the batch job is greater than the value
30836 of the *priority* component option-argument.
- 30837 .lt. The value of the *Priority* attribute of the batch job is less than the value of
30838 the *priority* component option-argument.
- 30839 .le. The value of the *Priority* attribute of the batch job is less than or equal to
30840 the value of the *priority* component option-argument.
- 30841 .ne. The value of the *Priority* attribute of the batch job does not equal the value
30842 of the *priority* component option-argument.

30843 **-q destination**

30844 Restrict selection to the specified batch queue or server, or both.

30845 The *qselect* utility shall select only batch jobs that are located at the destination
30846 indicated by the value of the *destination* option-argument.

30847 The destination defines a batch queue, a server, or a batch queue at a server.

30848 The *qselect* utility shall accept an option-argument for the **-q** option that conforms
30849 to the syntax for a destination. If the **-q** option is not presented to the *qselect* utility,
30850 the utility shall select batch jobs from all batch queues at the default batch server.

30851 If the option-argument describes only a batch queue, the *qselect* utility shall select
30852 only batch jobs from the batch queue of the specified name at the default batch
30853 server. The means by which *qselect* determines the default server is
30854 implementation-defined.

30855 If the option-argument describes only a batch server, the *qselect* utility shall select
30856 batch jobs from all the batch queues at that batch server.

30857 If the option-argument describes both a batch queue and a batch server, the *qselect*
30858 utility shall select only batch jobs from the specified batch queue at the specified

30859		server.
30860	-r y n	Restrict selection to batch jobs with the specified rerunability status. The <i>qselect</i> utility shall select only batch jobs for which the value of the <i>Rerunable</i> attribute of the batch job matches the value of the option-argument.
30863		The <i>qselect</i> utility shall accept a value for the option-argument that consists of either the single character 'y' or the single character 'n'. The character 'y' represents the value TRUE, and the character 'n' represents the value FALSE.
30866	-s states	Restrict selection to batch jobs in the specified states. The <i>qselect</i> utility shall accept an option-argument that consists of any combination of the characters 'e', 'q', 'r', 'w', 'h', and 't'. Conforming applications shall not repeat any character in the option-argument. The <i>qselect</i> utility shall permit the repetition of characters in the option-argument, but shall not assign additional meaning to repeated characters. The <i>qselect</i> utility shall interpret the characters in the <i>states</i> option-argument as follows: <ul style="list-style-type: none">e Represents the EXITING state.q Represents the QUEUED state.r Represents the RUNNING state.t Represents the TRANSITING state.h Represents the HELD state.w Represents the WAITING state. For each character in the <i>states</i> option-argument, the <i>qselect</i> utility shall select batch jobs in the corresponding state.
30882	-u user_list	Restrict selection to batch jobs owned by the specified user names. The <i>qselect</i> utility shall select only the batch jobs of those users specified in the <i>user_list</i> option-argument. The <i>qselect</i> utility shall accept a <i>user_list</i> option-argument that conforms to the following syntax: <i>username</i> [@ <i>host</i>] [, <i>username</i> [@ <i>host</i>] , ...] The <i>qselect</i> utility shall accept only one user name that is missing a corresponding host name. The <i>qselect</i> utility shall accept only one user name per named host.
30890	OPERANDS	
30891		None.
30892	STDIN	
30893		Not used.
30894	INPUT FILES	
30895		None.

30896 ENVIRONMENT VARIABLES

- 30897 The following environment variables shall affect the execution of *qselect*:
- 30898 **LANG** Provide a default value for the internationalization variables that are unset or null.
30899 (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2,
30900 Internationalization Variables for the precedence of internationalization variables
30901 used to determine the values of locale categories.)
- 30902 **LC_ALL** If set to a non-empty string value, override the values of all the other
30903 internationalization variables.
- 30904 **LC_CTYPE** Determine the locale for the interpretation of sequences of bytes of text data as
30905 characters (for example, single-byte as opposed to multi-byte characters in
30906 arguments).
- 30907 **LC_MESSAGES**
30908 Determine the locale that should be used to affect the format and contents of
30909 diagnostic messages written to standard error.
- 30910 **LOGNAME** Determine the login name of the user.
- 30911 **TZ** Determine the timezone used to interpret the *date-time* option-argument. If *TZ* is
30912 unset or null, an unspecified default timezone shall be used.

30913 ASYNCHRONOUS EVENTS

- 30914 Default.

30915 STDOUT

- 30916 The *qselect* utility shall write zero or more batch *job_identifiers* to standard output.
- 30917 The *qselect* utility shall separate the batch *job_identifiers* written to standard output by white
30918 space.
- 30919 The *qselect* utility shall write batch *job_identifiers* in the following format:
- 30920 *sequence_number.server_name@server*

30921 STDERR

- 30922 The standard error shall be used only for diagnostic messages.

30923 OUTPUT FILES

- 30924 None.

30925 EXTENDED DESCRIPTION

- 30926 None.

30927 EXIT STATUS

- 30928 The following exit values shall be returned:
- 30929 0 Successful completion.
- 30930 >0 An error occurred.

30931 CONSEQUENCES OF ERRORS

- 30932 Default.

30933 APPLICATION USAGE

30934 None.

30935 EXAMPLES

30936 The following example shows how a user might use the *qselect* utility in conjunction with the
30937 *qdel* utility to delete all of his or her jobs in the queued state without affecting any jobs that are
30938 already running:

30939 `qdel $(qselect -s q)`

30940 or:

30941 `qselect -s q || xargs qdel`

30942 RATIONALE

30943 The *qselect* utility allows users to acquire a list of job identifiers that match user-specified
30944 selection criteria. The list of identifiers returned by the *qselect* utility conforms to the syntax of
30945 the batch job identifier list processed by a utility such as *qmove*, *qdel*, and *qrsl*. The *qselect* utility is
30946 thus a powerful tool for causing another batch system utility to act upon a set of jobs that match
30947 a list of selection criteria.

30948 The options of the *qselect* utility let the user apply a number of useful filters for selecting jobs.
30949 Each option further restricts the selection of jobs. Many of the selection options allow the
30950 specification of a relational operator. The FORTRAN-like syntax of the operator—that is,
30951 ".lt."—was chosen rather than the C-like "<=" meta-characters.

30952 The **-a** option allows users to restrict the selected jobs to those that have been submitted (or
30953 altered) to wait until a particular time. The time period is determined by the argument of this
30954 option, which includes both a time and an operator—it is thus possible to select jobs waiting
30955 until a specific time, jobs waiting until after a certain time, or those waiting for a time before the
30956 specified time.

30957 The **-A** option allows users to restrict the selected jobs to those that have been submitted (or
30958 altered) to charge a particular account.

30959 The **-c** option allows users to restrict the selected jobs to those whose checkpointing interval
30960 falls within the specified range.

30961 The **-l** option allows users to select those jobs whose resource limits fall within the range
30962 indicated by the value of the option. For example, a user could select those jobs for which the
30963 CPU time limit is greater than two hours.

30964 The **-N** option allows users to select jobs by job name. For instance, all the parts of a task that
30965 have been divided in parallel jobs might be given the same name, and thus manipulated as a
30966 group by means of this option.

30967 The **-q** option allows users to select jobs in a specified queue.

30968 The **-r** option allows users to select only those jobs with a specified rerun criteria. For instance, a
30969 user might select only those jobs that can be rerun for use with the *qrerun* utility.

30970 The **-s** option allows users to select only those jobs that are in a certain state.

30971 The **-u** option allows users to select jobs that have been submitted to execute under a particular
30972 account.

30973 The selection criteria provided by the options of the *qselect* utility allow users to select jobs based
30974 on all the appropriate attributes that can be assigned to jobs by the *qsub* utility.

30975 Historically, the *qselect* utility has not been a part of existing practice; it is an improvement that
30976 has been introduced in this volume of IEEE Std 1003.1-2001.

30977 FUTURE DIRECTIONS

30978 None.

30979 SEE ALSO

30980 *qdel, qrerun, qrls, qselect, qsub, touch*, Chapter 3 (on page 101)

30981 CHANGE HISTORY

30982 Derived from IEEE Std 1003.2d-1994.

30983 **NAME**

30984 *qsig* — signal batch jobs

30985 **SYNOPSIS**

30986 BE *qsig [-s signal] job_identifier ...*

30987

30988 **DESCRIPTION**

30989 To signal a batch job is to send a signal to the session leader of the batch job. A batch job is
30990 signaled by sending a request to the batch server that manages the batch job. The *qsig* utility is a
30991 user-accessible batch client that requests the signaling of a batch job.

30992 The *qsig* utility shall signal those batch jobs for which a batch *job_identifier* is presented to the
30993 utility. The *qsig* utility shall not signal any batch jobs whose batch *job_identifiers* are not
30994 presented to the utility.

30995 The *qsig* utility shall signal batch jobs in the order in which the corresponding batch
30996 *job_identifiers* are presented to the utility. If the *qsig* utility fails to process a batch *job_identifier*
30997 successfully, the utility shall proceed to process the remaining batch *job_identifiers*, if any.

30998 The *qsig* utility shall signal batch jobs by sending a *Signal Job Request* to the batch server that
30999 manages the batch job.

31000 For each successfully processed batch *job_identifier*, the *qsig* utility shall have received a
31001 completion reply to each *Signal Job Request* sent to a batch server at the time the utility exits.

31002 **OPTIONS**

31003 The *qsig* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section
31004 12.2, Utility Syntax Guidelines.

31005 The following option shall be supported by the implementation:

31006 **-s signal** Define the signal to be sent to the batch job.

31007 The *qsig* utility shall accept a *signal* option-argument that is either a symbolic
31008 signal name or an unsigned integer signal number (see the POSIX.1-1990 standard,
31009 Section 3.3.1.1). The *qsig* utility shall accept signal names for which the SIG prefix
31010 has been omitted.

31011 If the *signal* option-argument is a signal name, the *qsig* utility shall send that name.

31012 If the *signal* option-argument is a number, the *qsig* utility shall send the signal
31013 value represented by the number.

31014 If the **-s** option is not presented to the *qsig* utility, the utility shall send the signal
31015 SIGTERM to each signaled batch job.

31016 **OPERANDS**

31017 The *qsig* utility shall accept one or more operands that conform to the syntax for a batch
31018 *job_identifier* (see Section 3.3.1 (on page 122)).

31019 **STDIN**

31020 Not used.

31021 **INPUT FILES**

31022 None.

31023 ENVIRONMENT VARIABLES

31024 The following environment variables shall affect the execution of *qsig*:

31025 **LANG** Provide a default value for the internationalization variables that are unset or null.
31026 (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2,
31027 Internationalization Variables for the precedence of internationalization variables
31028 used to determine the values of locale categories.)

31029 **LC_ALL** If set to a non-empty string value, override the values of all the other
31030 internationalization variables.

31031 **LC_CTYPE** Determine the locale for the interpretation of sequences of bytes of text data as
31032 characters (for example, single-byte as opposed to multi-byte characters in
31033 arguments).

31034 **LC_MESSAGES**

31035 Determine the locale that should be used to affect the format and contents of
31036 diagnostic messages written to standard error.

31037 **LOGNAME** Determine the login name of the user.

31038 ASYNCHRONOUS EVENTS

31039 Default.

31040 STDOUT

31041 An implementation of the *qsig* utility may write informative messages to standard output.

31042 STDERR

31043 The standard error shall be used only for diagnostic messages.

31044 OUTPUT FILES

31045 None.

31046 EXTENDED DESCRIPTION

31047 None.

31048 EXIT STATUS

31049 The following exit values shall be returned:

31050 0 Successful completion.

31051 >0 An error occurred.

31052 CONSEQUENCES OF ERRORS

31053 In addition to the default behavior, the *qsig* utility shall not be required to write a diagnostic
31054 message to standard error when the error reply received from a batch server indicates that the
31055 batch *job_identifier* does not exist on the server. Whether or not the *qsig* utility waits to output the
31056 diagnostic message while attempting to locate the batch job on other servers is implementation-
31057 defined.

31058 APPLICATION USAGE

31059 None.

31060 EXAMPLES

31061 None.

31062 RATIONALE

31063 The *qsig* utility allows users to signal batch jobs.

31064 A user may be unable to signal a batch job with the *kill* utility of the operating system for a
31065 number of reasons. First, the process ID of the batch job may be unknown to the user. Second,

31066 the processes of the batch job may be on a remote node. However, by virtue of communication
31067 between batch nodes, the *qsig* utility can arrange for the signaling of a process.

31068 Because a batch job that is not running cannot be signaled, and because the signal may not
31069 terminate the batch job, the *qsig* utility is not a substitute for the *qdel* utility.

31070 The options of the *qsig* utility allow the user to specify the signal that is to be sent to the batch
31071 job.

31072 The **-s** option allows users to specify a signal by name or by number, and thus override the
31073 default signal. The POSIX.1-1990 standard defines signals by both name and number.

31074 The *qsig* utility is a new utility, *vis-a-vis* existing practice; it has been defined in this volume of
31075 IEEE Std 1003.1-2001 in response to user-perceived shortcomings in existing practice.

31076 **FUTURE DIRECTIONS**

31077 None.

31078 **SEE ALSO**

31079 Chapter 3 (on page 101), *kill*, *qdel*

31080 **CHANGE HISTORY**

31081 Derived from IEEE Std 1003.2d-1994.

31082 **Issue 6**

31083 The *LC_TIME* and *TZ* entries are removed from the ENVIRONMENT VARIABLES section.

31084 NAME

31085 qstat — show status of batch jobs

31086 SYNOPSIS

31087 BE qstat [-f] *job_identifier* ...

31088 qstat -Q [-f] *destination* ...

31089 qstat -B [-f] *server_name* ...

31090

31091 DESCRIPTION

31092 The status of a batch job, batch queue, or batch server is obtained by a request to the server. The
31093 *qstat* utility is a user-accessible batch client that requests the status of one or more batch jobs,
31094 batch queues, or servers, and writes the status information to standard output.

31095 For each successfully processed batch *job_identifier*, the *qstat* utility shall display information
31096 about the corresponding batch job.

31097 For each successfully processed destination, the *qstat* utility shall display information about the
31098 corresponding batch queue.

31099 For each successfully processed server name, the *qstat* utility shall display information about the
31100 corresponding server.

31101 The *qstat* utility shall acquire batch job status information by sending a *Job Status Request* to a
31102 batch server. The *qstat* utility shall acquire batch queue status information by sending a *Queue*
31103 *Status Request* to a batch server. The *qstat* utility shall acquire server status information by
31104 sending a *Server Status Request* to a batch server.

31105 OPTIONS

31106 The *qstat* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section
31107 12.2, Utility Syntax Guidelines.

31108 The following options shall be supported by the implementation:

31109 **-f** Specify that a full display is produced.

31110 The minimum contents of a full display are specified in the STDOUT section.

31111 Additional contents and format of a full display are implementation-defined.

31112 **-Q** Specify that the operand is a destination.

31113 The *qstat* utility shall display information about each batch queue at each
31114 destination identified as an operand.

31115 **-B** Specify that the operand is a server name.

31116 The *qstat* utility shall display information about each server identified as an
31117 operand.

31118 OPERANDS

31119 If the **-Q** option is presented to the *qstat* utility, the utility shall accept one or more operands that
31120 conform to the syntax for a destination (see Section 3.3.2 (on page 123)).

31121 If the **-B** option is presented to the *qstat* utility, the utility shall accept one or more *server_name*
31122 operands.

31123 If neither the **-B** nor the **-Q** option is presented to the *qstat* utility, the utility shall accept one or
31124 more operands that conform to the syntax for a batch *job_identifier* (see Section 3.3.1 (on page
31125 122)).

31126 STDIN

31127 Not used.

31128 INPUT FILES

31129 None.

31130 ENVIRONMENT VARIABLES

31131 The following environment variables shall affect the execution of *qstat*:

31132 *HOME* Determine the pathname of the user's home directory.

31133 *LANG* Provide a default value for the internationalization variables that are unset or null.
(See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2,
Internationalization Variables for the precedence of internationalization variables
used to determine the values of locale categories.)

31137 *LC_ALL* If set to a non-empty string value, override the values of all the other
31138 internationalization variables.

LC_COLLATE

31140 Determine the locale for the behavior of ranges, equivalence classes, and multi-
31141 character collating elements within regular expressions.

31142 *LC_CTYPE* Determine the locale for the interpretation of sequences of bytes of text data as
31143 characters (for example, single-byte as opposed to multi-byte characters in
31144 arguments).

LC_MESSAGES

31146 Determine the locale that should be used to affect the format and contents of
31147 diagnostic messages written to standard error.

LC_NUMERIC

31149 Determine the locale for selecting the radix character used when writing floating-
31150 point formatted output.

31151 ASYNCHRONOUS EVENTS

31152 Default.

31153 STDOUT

31154 If an operand presented to the *qstat* utility is a batch *job_identifier* and the **-f** option is not
31155 specified, the *qstat* utility shall display the following items on a single line, in the stated order,
31156 with white space between each item, for each successfully processed operand:

- 31157 • The batch *job_identifier*
- 31158 • The batch job name
- 31159 • The *Job_Owner* attribute
- 31160 • The CPU time used by the batch job
- 31161 • The batch job state
- 31162 • The batch job location

31163 If an operand presented to the *qstat* utility is a batch *job_identifier* and the **-f** option is specified,
31164 the *qstat* utility shall display the following items for each success fully processed operand:

- 31165 • The batch *job_identifier*
- 31166 • The batch job name

- 31167 • The *Job_Owner* attribute
- 31168 • The execution user ID
- 31169 • The CPU time used by the batch job
- 31170 • The batch job state
- 31171 • The batch job location
- 31172 • Additional implementation-defined information, if any, about the batch job or batch queue

31173 If an operand presented to the *qstat* utility is a destination, the **-Q** option is specified, and the **-f** option is not specified, the *qstat* utility shall display the following items on a single line, in the stated order, with white space between each item, for each successfully processed operand:

- 31176 • The batch queue name
- 31177 • The maximum number of batch jobs that shall be run in the batch queue concurrently
- 31178 • The total number of batch jobs in the batch queue
- 31179 • The status of the batch queue
- 31180 • For each state, the number of batch jobs in that state in the batch queue and the name of the state
- 31182 • The type of batch queue (execution or routing)

31183 If the operands presented to the *qstat* utility are destinations, the **-Q** option is specified, and the **-f** option is specified, the *qstat* utility shall display the following items for each successfully processed operand:

- 31186 • The batch queue name
- 31187 • The maximum number of batch jobs that shall be run in the batch queue concurrently
- 31188 • The total number of batch jobs in the batch queue
- 31189 • The status of the batch queue
- 31190 • For each state, the number of batch jobs in that state in the batch queue and the name of the state
- 31192 • The type of batch queue (execution or routing)
- 31193 • Additional implementation-defined information, if any, about the batch queue

31194 If the operands presented to the *qstat* utility are batch server names, the **-B** option is specified, and the **-f** option is not specified, the *qstat* utility shall display the following items on a single line, in the stated order, with white space between each item, for each successfully processed operand:

- 31198 • The batch server name
- 31199 • The maximum number of batch jobs that shall be run in the batch queue concurrently
- 31200 • The total number of batch jobs managed by the batch server
- 31201 • The status of the batch server
- 31202 • For each state, the number of batch jobs in that state and the name of the state

31203 If the operands presented to the *qstat* utility are server names, the **-B** option is specified, and the **-f** option is specified, the *qstat* utility shall display the following items for each successfully processed operand:

- 31206 • The server name
- 31207 • The maximum number of batch jobs that shall be run in the batch queue concurrently
- 31208 • The total number of batch jobs managed by the server
- 31209 • The status of the server
- 31210 • For each state, the number of batch jobs in that state and the name of the state
- 31211 • Additional implementation-defined information, if any, about the server

31212 STDERR

31213 The standard error shall be used only for diagnostic messages.

31214 OUTPUT FILES

31215 None.

31216 EXTENDED DESCRIPTION

31217 None.

31218 EXIT STATUS

31219 The following exit values shall be returned:

31220 0 Successful completion.

31221 >0 An error occurred.

31222 CONSEQUENCES OF ERRORS

31223 In addition to the default behavior, the *qstat* utility shall not be required to write a diagnostic message to standard error when the error reply received from a batch server indicates that the batch *job_identifier* does not exist on the server. Whether or not the *qstat* utility waits to output the diagnostic message while attempting to locate the batch job on other servers is implementation-defined.

31228 APPLICATION USAGE

31229 None.

31230 EXAMPLES

31231 None.

31232 RATIONALE

31233 The *qstat* utility allows users to display the status of jobs and list the batch jobs in queues.

31234 The operands of the *qstat* utility may be either job identifiers, queues (specified as destination identifiers), or batch server names. The **-Q** and **-B** options, or absence thereof, indicate the nature of the operands.

31237 The other options of the *qstat* utility allow the user to control the amount of information displayed and the format in which it is displayed. Should a user wish to display the status of a set of jobs that match a selection criteria, the *qselect* utility may be used to acquire such a list.

31240 The **-f** option allows users to request a “full” display in an implementation-defined format.

31241 Historically, the *qstat* utility has been a part of the NQS and its derivatives, the existing practice on which it is based.

31243 FUTURE DIRECTIONS

31244 None.

31245 SEE ALSO

31246 Chapter 3 (on page 101), *qselect*

31247 CHANGE HISTORY

31248 Derived from IEEE Std 1003.2d-1994.

31249 Issue 6

31250 IEEE PASC Interpretation 1003.2 #191 is applied, removing the following ENVIRONMENT
31251 VARIABLES listed as affecting *qstat*: *COLUMNS*, *LINES*, *LOGNAME*, *TERM*, and *TZ*.

31252 The *LC_TIME* entry is also removed from the ENVIRONMENT VARIABLES section.

31253 NAME

31254 qsub — submit a script

31255 SYNOPSIS

```
31256 BE   qsub [-a date_time][-A account_string][-c interval]
31257     [-C directive_prefix][-e path_name][-h][-j join_list][-k keep_list]
31258     [-m mail_options][-M mail_list][-N name]
31259     [-o path_name][-p priority][-q destination][-r y|n]
31260     [-S path_name_list][-u user_list][-v variable_list][-V]
31261     [-z][script]
```

31262

31263 DESCRIPTION

31264 To submit a script is to create a batch job that executes the script. A script is submitted by a
31265 request to a batch server. The *qsub* utility is a user-accessible batch client that submits a script.

31266 Upon successful completion, the *qsub* utility shall have created a batch job that will execute the
31267 submitted script.

31268 The *qsub* utility shall submit a script by sending a *Queue Job Request* to a batch server.

31269 The *qsub* utility shall place the value of the following environment variables in the *Variable_List*
31270 attribute of the batch job: *HOME*, *LANG*, *LOGNAME*, *PATH*, *MAIL*, *SHELL*, and *TZ*. The name
31271 of the environment variable shall be the current name prefixed with the string *PBS_O_*.

31272 **Note:** If the current value of the *HOME* variable in the environment space of the *qsub* utility is
31273 /aa/bb/cc, then *qsub* shall place *PBS_O_HOME=/aa/bb/cc* in the *Variable_List* attribute of the
31274 batch job.

31275 In addition to the variables described above, the *qsub* utility shall add the following variables
31276 with the indicated values to the variable list:

31277 *PBS_O_WORKDIR* The absolute path of the current working directory of the *qsub* utility
31278 process.

31279 *PBS_O_HOST* The name of the host on which the *qsub* utility is running.

31280 OPTIONS

31281 The *qsub* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section
31282 12.2, Utility Syntax Guidelines.

31283 The following options shall be supported by the implementation:

31284 **-a** *date_time* Define the time at which a batch job becomes eligible for execution.

31285 The *qsub* utility shall accept an option-argument that conforms to the syntax of the
31286 *time* operand of the *touch* utility.

31287

Table 4-18 Environment Variable Values (Utilities)

31288

Variable Name	Value at qsub Time
<i>PBS_O_HOME</i>	<i>HOME</i>
<i>PBS_O_HOST</i>	Client host name
<i>PBS_O_LANG</i>	<i>LANG</i>
<i>PBS_O_LOGNAME</i>	<i>LOGNAME</i>
<i>PBS_O_PATH</i>	<i>PATH</i>
<i>PBS_O_MAIL</i>	<i>MAIL</i>
<i>PBS_O_SHELL</i>	<i>SHELL</i>
<i>PBS_O_TZ</i>	<i>TZ</i>
<i>PBS_O_WORKDIR</i>	Current working directory

31289

31290

31291

31292

31293

31294

31295

31296

31297

Note: The server that initiates execution of the batch job will add other variables to the batch job's environment; see Section 3.2.2.1 (on page 106).

The *qsub* utility shall set the *Execution_Time* attribute of the batch job to the number of seconds since the Epoch that is equivalent to the local time expressed by the value of the *date_time* option-argument. The Epoch is defined in the Base Definitions volume of IEEE Std 1003.1-2001, Section 3.149, Epoch.

If the **-a** option is not presented to the *qsub* utility, the utility shall set the *Execution_Time* attribute of the batch job to a time (number of seconds since the Epoch) that is earlier than the time at which the utility exits.

-A account_string
Define the account to which the resource consumption of the batch job should be charged.

The syntax of the *account_string* option-argument is unspecified.

The *qsub* utility shall set the *Account_Name* attribute of the batch job to the value of the *account_string* option-argument.

If the **-A** option is not presented to the *qsub* utility, the utility shall omit the *Account_Name* attribute from the attributes of the batch job.

-c interval
Define whether the batch job should be checkpointed, and if so, how often.

The *qsub* utility shall accept a value for the *interval* option-argument that is one of the following:

n No checkpointing shall be performed on the batch job (NO_CHECKPOINT).

s Checkpointing shall be performed only when the batch server is shut down (CHECKPOINT_AT_SHUTDOWN).

c Automatic periodic checkpointing shall be performed at the *Minimum_Cpu_Interval* attribute of the batch queue, in units of CPU minutes (CHECKPOINT_AT_MIN_CPU_INTERVAL).

c=minutes Automatic periodic checkpointing shall be performed every *minutes* of CPU time, or every *Minimum_Cpu_Interval* minutes, whichever is greater. The *minutes* argument shall conform to the syntax for unsigned integers and shall be greater than zero.

The *qsub* utility shall set the *Checkpoint* attribute of the batch job to the value of the *interval* option-argument.

31331	If the -c option is not presented to the <i>qsub</i> utility, the utility shall set the <i>Checkpoint</i> attribute of the batch job to the single character 'u' (CHECKPOINT_UNSPECIFIED).
31334	-C directive_prefix
31335	Define the prefix that declares a directive to the <i>qsub</i> utility within the script.
31336	The <i>directive_prefix</i> is not a batch job attribute; it affects the behavior of the <i>qsub</i> utility.
31338	If the -C option is presented to the <i>qsub</i> utility, and the value of the <i>directive_prefix</i> option-argument is the null string, the utility shall not scan the script file for directives. If the -C option is not presented to the <i>qsub</i> utility, then the value of the <i>PBS_DPREFIX</i> environment variable is used. If the environment variable is not defined, then #PBS encoded in the portable character set is the default.
31343	-e path_name
31344	Define the path to be used for the standard error stream of the batch job.
31345	The <i>qsub</i> utility shall accept a <i>path_name</i> option-argument which can be preceded by a host name element of the form <i>hostname</i> :
31347	If the <i>path_name</i> option-argument constitutes an absolute pathname, the <i>qsub</i> utility shall set the <i>Error_Path</i> attribute of the batch job to the value of the <i>path_name</i> option-argument.
31350	If the <i>path_name</i> option-argument constitutes a relative pathname and no host name element is specified, the <i>qsub</i> utility shall set the <i>Error_Path</i> attribute of the batch job to the value of the absolute pathname derived by expanding the <i>path_name</i> option-argument relative to the current directory of the process executing <i>qsub</i> .
31355	If the <i>path_name</i> option-argument constitutes a relative pathname and a host name element is specified, the <i>qsub</i> utility shall set the <i>Error_Path</i> attribute of the batch job to the value of the <i>path_name</i> option-argument without expansion. The host name element shall be included.
31359	If the <i>path_name</i> option-argument does not include a host name element, the <i>qsub</i> utility shall prefix the pathname with <i>hostname</i> : <i>, where hostname is the name of the host upon which the qsub utility is being executed.</i>
31362	If the -e option is not presented to the <i>qsub</i> utility, the utility shall set the <i>Error_Path</i> attribute of the batch job to the host name and path of the current directory of the submitting process and the default filename.
31365	The default filename for standard error has the following format:
31366	<i>job_name.esequence_number</i>
31367	-h
	Specify that a USER hold is applied to the batch job.
31368	The <i>qsub</i> utility shall set the value of the <i>Hold_Types</i> attribute of the batch job to the value USER.
31370	If the -h option is not presented to the <i>qsub</i> utility, the utility shall set the <i>Hold_Types</i> attribute of the batch job to the value NO_HOLD.
31372	-j join_list
31373	Define which streams of the batch job are to be merged. The <i>qsub -j</i> option shall accept a value for the <i>join_list</i> option-argument that is a string of alphanumeric characters in the portable character set (see the Base Definitions volume of

31375 IEEE Std 1003.1-2001, Section 6.1, Portable Character Set).

31376 The *qsub* utility shall accept a *join_list* option-argument that consists of one or
31377 more of the characters 'e' and 'o', or the single character 'n'.

31378 All of the other batch job output streams specified will be merged into the output
31379 stream represented by the character listed first in the *join_list* option-argument.

31380 For each unique character in the *join_list* option-argument, the *qsub* utility shall
31381 add a value to the *Join_Path* attribute of the batch job as follows, each representing
31382 a different batch job stream to join:

31383 e The standard error of the batch job (JOIN_STD_ERROR).

31384 o The standard output of the batch job (JOIN_STD_OUTPUT).

31385 An existing *Join_Path* attribute can be cleared by the following join type:

31386 n NO_JOIN

31387 If 'n' is specified, then no files are joined. The *qsub* utility shall consider it an error
31388 if any join type other than 'n' is combined with join type 'n'.

31389 Strictly conforming applications shall not repeat any of the characters 'e', 'o', or
31390 'n' within the *join_list* option-argument. The *qsub* utility shall permit the
31391 repetition of characters, but shall not assign additional meaning to the repeated
31392 characters.

31393 An implementation may define other join types. The conformance document for an
31394 implementation shall describe any additional batch job streams, how they are
31395 specified, their internal behavior, and how they affect the behavior of the utility.

31396 If the **-j** option is not presented to the *qsub* utility, the utility shall set the value of
31397 the *Join_Path* attribute of the batch job to NO_JOIN.

31398 **-k** *keep_list* Define which output of the batch job to retain on the execution host.

31399 The *qsub -k* option shall accept a value for the *keep_list* option-argument that is a
31400 string of alphanumeric characters in the portable character set (see the Base
31401 Definitions volume of IEEE Std 1003.1-2001, Section 6.1, Portable Character Set).

31402 The *qsub* utility shall accept a *keep_list* option-argument that consists of one or
31403 more of the characters 'e' and 'o', or the single character 'n'.

31404 For each unique character in the *keep_list* option-argument, the *qsub* utility shall
31405 add a value to the *Keep_Files* attribute of the batch job as follows, each representing
31406 a different batch job stream to keep:

31407 e The standard error of the batch job (KEEP_STD_ERROR).

31408 o The standard output of the batch job (KEEP_STD_OUTPUT).

31409 If both 'e' and 'o' are specified, then both files are retained. An existing
31410 *Keep_Files* attribute can be cleared by the following keep type:

31411 n NO_KEEP

31412 If 'n' is specified, then no files are retained. The *qsub* utility shall consider it an
31413 error if any keep type other than 'n' is combined with keep type 'n'.

31414 Strictly conforming applications shall not repeat any of the characters 'e', 'o', or
31415 'n' within the *keep_list* option-argument. The *qsub* utility shall permit the
31416 repetition of characters, but shall not assign additional meaning to the repeated

- 31417 characters.
- 31418 An implementation may define other keep types. The conformance document for
31419 an implementation shall describe any additional keep types, how they are
31420 specified, their internal behavior, and how they affect the behavior of the utility. If
31421 the **-k** option is not presented to the *qsub* utility, the utility shall set the *Keep_Files*
31422 attribute of the batch job to the value NO_KEEP.
- 31423 **-m mail_options**
- 31424 Define the points in the execution of the batch job at which the batch server that
31425 manages the batch job shall send mail about a change in the state of the batch job.
- 31426 The *qsub -m* option shall accept a value for the *mail_options* option-argument that
31427 is a string of alphanumeric characters in the portable character set (see the Base
31428 Definitions volume of IEEE Std 1003.1-2001, Section 6.1, Portable Character Set).
- 31429 The *qsub* utility shall accept a value for the *mail_options* option-argument that is a
31430 string of one or more of the characters 'e', 'b', and 'a', or the single character
31431 'n'.
- 31432 For each unique character in the *mail_options* option-argument, the *qsub* utility shall
31433 add a value to the *Mail_Users* attribute of the batch job as follows, each
31434 representing a different time during the life of a batch job at which to send mail:
- 31435 e MAIL_AT_EXIT
- 31436 b MAIL_AT_BEGINNING
- 31437 a MAIL_AT_ABORT
- 31438 If any of these characters are duplicated in the *mail_options* option-argument, the
31439 duplicates shall be ignored.
- 31440 An existing *Mail_Points* attribute can be cleared by the following mail type:
- 31441 n NO_MAIL
- 31442 If 'n' is specified, then mail is not sent. The *qsub* utility shall consider it an error if
31443 any mail type other than 'n' is combined with mail type 'n'.
- 31444 Strictly conforming applications shall not repeat any of the characters 'e', 'b',
31445 'a', or 'n' within the *mail_options* option-argument.
- 31446 The *qsub* utility shall permit the repetition of characters, but shall not assign
31447 additional meaning to the repeated characters. An implementation may define
31448 other mail types. The conformance document for an implementation shall describe
31449 any additional mail types, how they are specified, their internal behavior, and how
31450 they affect the behavior of the utility.
- 31451 If the **-m** option is not presented to the *qsub* utility, the utility shall set the
31452 *Mail_Points* attribute to the value MAIL_AT_ABORT.
- 31453 **-M mail_list** Define the list of users to which a batch server that executes the batch job shall
31454 send mail, if the server sends mail about the batch job.
- 31455 The syntax of the *mail_list* option-argument is unspecified.
- 31456 If the implementation of the *qsub* utility uses a name service to locate users, the
31457 utility should accept the syntax used by the name service.
- 31458 If the implementation of the *qsub* utility does not use a name service to locate
31459 users, the implementation should accept the following syntax for user names:

31460 *mail_address[, , mail_address , , . . .]*

31461 The interpretation of *mail_address* is implementation-defined.

31462 The *qsub* utility shall set the *Mail_Users* attribute of the batch job to the value of the *mail_list* option-argument.

31464 If the **-M** option is not presented to the *qsub* utility, the utility shall place only the user name and host name for the current process in the *Mail_Users* attribute of the batch job.

31467 **-N name**
 Define the name of the batch job.

31468 The *qsub* **-N** option shall accept a value for the *name* option-argument that is a string of up to 15 alphanumeric characters in the portable character set (see the Base Definitions volume of IEEE Std 1003.1-2001, Section 6.1, Portable Character Set) where the first character is alphabetic.

31472 The *qsub* utility shall set the value of the *Job_Name* attribute of the batch job to the value of the *name* option-argument.

31474 If the **-N** option is not presented to the *qsub* utility, the utility shall set the *Job_Name* attribute of the batch job to the name of the *script* argument from which the directory specification if any, has been removed.

31477 If the **-N** option is not presented to the *qsub* utility, and the script is read from standard input, the utility shall set the *Job_Name* attribute of the batch job to the value STDIN.

31480 **-o path_name**
 Define the path for the standard output of the batch job.

31482 The *qsub* utility shall accept a *path_name* option-argument that conforms to the syntax of the *path_name* element defined in the System Interfaces volume of IEEE Std 1003.1-2001, which can be preceded by a host name element of the form *hostname*:

31486 If the *path_name* option-argument constitutes an absolute pathname, the *qsub* utility shall set the *Output_Path* attribute of the batch job to the value of the *path_name* option-argument without expansion.

31489 If the *path_name* option-argument constitutes a relative pathname and no host name element is specified, the *qsub* utility shall set the *Output_Path* attribute of the batch job to the pathname derived by expanding the value of the *path_name* option-argument relative to the current directory of the process executing the *qsub*.

31493 If the *path_name* option-argument constitutes a relative pathname and a host name element is specified, the *qsub* utility shall set the *Output_Path* attribute of the batch job to the value of the *path_name* option-argument without expansion.

31496 If the *path_name* option-argument does not specify a host name element, the *qsub* utility shall prefix the pathname with *hostname*:*,* where *hostname* is the name of the host upon which the *qsub* utility is executing.

31500 If the **-o** option is not presented to the *qsub* utility, the utility shall set the *Output_Path* attribute of the batch job to the host name and path of the current directory of the submitting process and the default filename.

31502 The default filename for standard output has the following format:

31503		<i>job_name.osequence_number</i>
31504	-p priority	Define the priority the batch job should have relative to other batch jobs owned by the batch server.
31505		The <i>qsub</i> utility shall set the <i>Priority</i> attribute of the batch job to the value of the <i>priority</i> option-argument.
31506		If the -p option is not presented to the <i>qsub</i> utility, the value of the <i>Priority</i> attribute is implementation-defined.
31507		The <i>qsub</i> utility shall accept a value for the <i>priority</i> option-argument that conforms to the syntax for signed decimal integers, and which is not less than -1 024 and not greater than 1 023.
31508		
31509		
31510		
31511		
31512		
31513	-q destination	Define the destination of the batch job.
31514		The destination is not a batch job attribute; it determines the batch server, and possibly the batch queue, to which the <i>qsub</i> utility batch queues the batch job.
31515		
31516		
31517		The <i>qsub</i> utility shall submit the script to the batch server named by the <i>destination</i> option-argument or the server that owns the batch queue named in the <i>destination</i> option-argument.
31518		
31519		
31520		The <i>qsub</i> utility shall accept an option-argument for the -q option that conforms to the syntax for a destination (see Section 3.3.2 (on page 123)).
31521		
31522		If the -q option is not presented to the <i>qsub</i> utility, the <i>qsub</i> utility shall submit the batch job to the default destination. The mechanism for determining the default destination is implementation-defined.
31523		
31524		
31525	-r y n	Define whether the batch job is rerunnable.
31526		If the value of the option-argument is <i>y</i> , the <i>qsub</i> utility shall set the <i>Rerunable</i> attribute of the batch job to TRUE.
31527		
31528		If the value of the option-argument is <i>n</i> , the <i>qsub</i> utility shall set the <i>Rerunable</i> attribute of the batch job to FALSE.
31529		
31530		If the -r option is not presented to the <i>qsub</i> utility, the utility shall set the <i>Rerunable</i> attribute of the batch job to TRUE.
31531		
31532	-S path_name_list	Define the pathname to the shell under which the batch job is to execute.
31533		
31534		The <i>qsub</i> utility shall accept a <i>path_name_list</i> option-argument that conforms to the following syntax:
31535		
31536		<i>pathname[@host][, , pathname[@host], , . . .]</i>
31537		The <i>qsub</i> utility shall allow only one pathname for a given host name. The <i>qsub</i> utility shall allow only one pathname that is missing a corresponding host name.
31538		
31539		The <i>qsub</i> utility shall add a value to the <i>Shell_Path_List</i> attribute of the batch job for each entry in the <i>path_name_list</i> option-argument.
31540		
31541		If the -S option is not presented to the <i>qsub</i> utility, the utility shall set the <i>Shell_Path_List</i> attribute of the batch job to the null string.
31542		
31543		The conformance document for an implementation shall describe the mechanism used to set the default shell and determine the current value of the default shell.
31544		

- 31545 An implementation shall provide a means for the installation to set the default
31546 shell to the login shell of the user under which the batch job is to execute. See
31547 Section 3.3.3 (on page 123) for a means of removing *keyword=value* (and
31548 *value@keyword*) pairs and other general rules for list-oriented batch job attributes.
- 31549 **-u user_list** Define the user name under which the batch job is to execute.
31550 The *qsub* utility shall accept a *user_list* option-argument that conforms to the
31551 following syntax:
31552 *username[@host] [, , username[@host], , . . .]*
31553 The *qsub* utility shall accept only one user name that is missing a corresponding
31554 host name. The *qsub* utility shall accept only one user name per named host.
31555 The *qsub* utility shall add a value to the *User_List* attribute of the batch job for each
31556 entry in the *user_list* option-argument.
31557 If the **-u** option is not presented to the *qsub* utility, the utility shall set the *User_List*
31558 attribute of the batch job to the user name from which the utility is executing. See
31559 Section 3.3.3 (on page 123) for a means of removing *keyword=value* (and
31560 *value@keyword*) pairs and other general rules for list-oriented batch job attributes.
- 31561 **-v variable_list**
31562 Add to the list of variables that are exported to the session leader of the batch job.
31563 A *variable_list* is a set of strings of either the form <*variable*> or <*variable*=*value*>,
31564 delimited by commas.
31565 If the **-v** option is presented to the *qsub* utility, the utility shall also add, to the
31566 environment *Variable_List* attribute of the batch job, every variable named in the
31567 environment *variable_list* option-argument and, optionally, values of specified
31568 variables.
31569 If a value is not provided on the command line, the *qsub* utility shall set the value
31570 of each variable in the environment *Variable_List* attribute of the batch job to the
31571 value of the corresponding environment variable for the process in which the
31572 utility is executing; see Table 4-18 (on page 805).
31573 A conforming application shall not repeat a variable in the environment
31574 *variable_list* option-argument.
31575 The *qsub* utility shall not repeat a variable in the environment *Variable_List*
31576 attribute of the batch job. See Section 3.3.3 (on page 123) for a means of removing
31577 *keyword=value* (and *value@keyword*) pairs and other general rules for list-oriented
31578 batch job attributes.
31579 **-V** Specify that all of the environment variables of the process are exported to the
31580 context of the batch job.
31581 The *qsub* utility shall place every environment variable in the process in which the
31582 utility is executing in the list and shall set the value of each variable in the attribute
31583 to the value of that variable in the process.
31584 **-z** Specify that the utility does not write the batch *job_identifier* of the created batch
31585 job to standard output.
31586 If the **-z** option is presented to the *qsub* utility, the utility shall not write the batch
31587 *job_identifier* of the created batch job to standard output.

31588 If the **-z** option is not presented to the *qsub* utility, the utility shall write the
31589 identifier of the created batch job to standard output.

31590 OPERANDS

31591 The *qsub* utility shall accept a *script* operand that indicates the path to the script of the batch job.

31592 If the *script* operand is not presented to the *qsub* utility, or if the operand is the single-character
31593 string '**-**', the utility shall read the script from standard input.

31594 If the script represents a partial path, the *qsub* utility shall expand the path relative to the current
31595 directory of the process executing the utility.

31596 STDIN

31597 The *qsub* utility reads the script of the batch job from standard input if the script operand is
31598 omitted or is the single character '**-**'.

31599 INPUT FILES

31600 In addition to binding the file indicated by the *script* operand to the batch job, the *qsub* utility
31601 reads the script file and acts on directives in the script.

31602 ENVIRONMENT VARIABLES

31603 The following environment variables shall affect the execution of *qsub*:

31604 *LANG* Provide a default value for the internationalization variables that are unset or null.
31605 (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2,
31606 Internationalization Variables for the precedence of internationalization variables
31607 used to determine the values of locale categories.)

31608 *LC_ALL* If set to a non-empty string value, override the values of all the other
31609 internationalization variables.

31610 *LC_CTYPE* Determine the locale for the interpretation of sequences of bytes of text data as
31611 characters (for example, single-byte as opposed to multi-byte characters in
31612 arguments).

31613 *LC_MESSAGES*

31614 Determine the locale that should be used to affect the format and contents of
31615 diagnostic messages written to standard error.

31616 *LOGNAME* Determine the login name of the user.

31617 *PBS_DPREFIX*

31618 Determine the default prefix for directives within the script.

31619 *SHELL* Determine the pathname of the preferred command language interpreter of the
31620 user.

31621 *TZ* Determine the timezone used to interpret the *date-time* option-argument. If *TZ* is
31622 unset or null, an unspecified default timezone shall be used.

31623 ASYNCHRONOUS EVENTS

31624 Once created, a batch job exists until it exits, aborts, or is deleted.

31625 After a batch job is created by the *qsub* utility, batch servers might route, execute, modify, or
31626 delete the batch job.

31627 STDOUT

31628 The *qsub* utility writes the batch *job_identifier* assigned to the batch job to standard output, unless
31629 the **-z** option is specified.

31630 STDERR

31631 The standard error shall be used only for diagnostic messages.

31632 OUTPUT FILES

31633 None.

31634 EXTENDED DESCRIPTION**31635 Script Preservation**

31636 The *qsub* utility shall make the script available to the server executing the batch job in such a way
31637 that the server executes the script as it exists at the time of submission.

31638 The *qsub* utility can send a copy of the script to the server with the *Queue Job Request* or store a
31639 temporary copy of the script in a location specified to the server.

31640 Option Specification

31641 A script can contain directives to the *qsub* utility.

31642 The *qsub* utility shall scan the lines of the script for directives, skipping blank lines, until the first
31643 line that begins with a string other than the directive string; if directives occur on subsequent
31644 lines, the utility shall ignore those directives.

31645 Lines are separated by a <newline>. If the first line of the script begins with "#!" or a colon
31646 (' : '), then it is skipped. The *qsub* utility shall process a line in the script as a directive if and
31647 only if the string of characters from the first non-white-space character on the line until the first
31648 <space> or <tab> on the line match the directive prefix. If a line in the script contains a directive
31649 and the final characters of the line are backslash ('\') and <newline>, then the next line shall be
31650 interpreted as a continuation of that directive.

31651 The *qsub* utility shall process the options and option-arguments contained on the directive prefix
31652 line using the same syntax as if the options were input on the *qsub* utility.

31653 The *qsub* utility shall continue to process a directive prefix line until after a <newline> is
31654 encountered. An implementation may ignore lines which, according to the syntax of the shell
31655 that will interpret the script, are comments. An implementation shall describe in the
31656 conformance document the format of any shell comments that it will recognize.

31657 If an option is present in both a directive and the arguments to the *qsub* utility, the utility shall
31658 ignore the option and the corresponding option-argument, if any, in the directive.

31659 If an option that is present in the directive is not present in the arguments to the *qsub* utility, the
31660 utility shall process the option and the option-argument, if any.

31661 In order of preference, the *qsub* utility shall select the directive prefix from one of the following
31662 sources:

- If the **-C** option is presented to the utility, the value of the *directive_prefix* option-argument
- If the environment variable *PBS_DPREFIX* is defined, the value of that variable
- The four-character string "#PBS" encoded in the portable character set

31666 If the **-C** option is present in the script file it shall be ignored.

31667 EXIT STATUS

31668 The following exit values shall be returned:

31669 0 Successful completion.

31670 >0 An error occurred.

31671 CONSEQUENCES OF ERRORS

31672 Default.

31673 APPLICATION USAGE

31674 None.

31675 EXAMPLES

31676 None.

31677 RATIONALE

31678 The *qsub* utility allows users to create a batch job that will process the script specified as the
31679 operand of the utility.

31680 The options of the *qsub* utility allow users to control many aspects of the queuing and execution
31681 of a batch job.

31682 The **-a** option allows users to designate the time after which the batch job will become eligible to
31683 run. By specifying an execution time, users can take advantage of resources at off-peak hours,
31684 synchronize jobs with chronologically predictable events, and perhaps take advantage of off-
31685 peak pricing of computing time. For these reasons and others, a timing option is existing practice
31686 on the part of almost every batch system, including NQS.

31687 The **-A** option allows users to specify the account that will be charged for the batch job. Support
31688 for account is not mandatory for conforming batch servers.

31689 The **-C** option allows users to prescribe the prefix for directives within the script file. The default
31690 prefix "#PBS" may be inappropriate if the script will be interpreted with an alternate shell, as
31691 specified by the **-S** option.

31692 The **-c** option allows users to establish the checkpointing interval for their jobs. A checkpointing
31693 system, which is not defined by this volume of IEEE Std 1003.1-2001, allows recovery of a batch
31694 job at the most recent checkpoint in the event of a crash. Checkpointing is typically used for jobs
31695 that consume expensive computing time or must meet a critical schedule. Users should be
31696 allowed to make the tradeoff between the overhead of checkpointing and the risk to the timely
31697 completion of the batch job; therefore, this volume of IEEE Std 1003.1-2001 provides the
31698 checkpointing interval option. Support for checkpointing is optional for batch servers.

31699 The **-e** option allows users to redirect the standard error streams of their jobs to a non-default
31700 path. For example, if the submitted script generally produces a great deal of useless error output,
31701 a user might redirect the standard error output to the null device. Or, if the file system holding
31702 the default location (the home directory of the user) has too little free space, the user might
31703 redirect the standard error stream to a file in another file system.

31704 The **-h** option allows users to create a batch job that is held until explicitly released. The ability
31705 to create a held job is useful when some external event must complete before the batch job can
31706 execute. For example, the user might submit a held job and release it when the system load has
31707 dropped.

31708 The **-j** option allows users to merge the standard error of a batch job into its standard output
31709 stream, which has the advantage of showing the sequential relationship between output and
31710 error messages.

31711 The **-m** option allows users to designate those points in the execution of a batch job at which
31712 mail will be sent to the submitting user, or to the account(s) indicated by the **-M** option. By
31713 requesting mail notification at points of interest in the life of a job, the submitting user, or other
31714 designated users, can track the progress of a batch job.

31715 The **-N** option allows users to associate a name with the batch job. The job name in no way
31716 affects the processing of the batch job, but rather serves as a mnemonic handle for users. For
31717 example, the batch job name can help the user distinguish between multiple jobs listed by the
31718 *qstat* utility.

31719 The **-o** option allows users to redirect the standard output stream. A user might, for example,
31720 wish to redirect to the null device the standard output stream of a job that produces copious yet
31721 superfluous output.

31722 The **-P** option allows users to designate the relative priority of a batch job for selection from a
31723 queue.

31724 The **-q** option allows users to specify an initial queue for the batch job. If the user specifies a
31725 routing queue, the batch server routes the batch job to another queue for execution or further
31726 routing. If the user specifies a non-routing queue, the batch server of the queue eventually
31727 executes the batch job.

31728 The **-r** option allows users to control whether the submitted job will be rerun if the controlling
31729 batch node fails during execution of the batch job. The **-r** option likewise allows users to
31730 indicate whether or not the batch job is eligible to be rerun by the *qrerun* utility. Some jobs cannot
31731 be correctly rerun because of changes they make in the state of databases or other aspects of
31732 their environment. This volume of IEEE Std 1003.1-2001 specifies that the default, if the **-r**
31733 option is not presented to the utility, will be that the batch job cannot be rerun, since the result of
31734 rerunning a non-rerunnable job might be catastrophic.

31735 The **-S** option allows users to specify the program (usually a shell) that will be invoked to
31736 process the script of the batch job. This option has been modified to allow a list of shell names
31737 and locations associated with different hosts.

31738 The **-u** option is useful when the submitting user is authorized to use more than one account on
31739 a given host, in which case the **-u** option allows the user to select from among those accounts.
31740 The option-argument is a list of user-host pairs, so that the submitting user can provide different
31741 user identifiers for different nodes in the event the batch job is routed. The **-u** option provides a
31742 lot of flexibility to accommodate sites with complex account structures. Users that have the
31743 same user identifier on all the hosts they are authorized to use will not need to use the **-u** option.

31744 The **-V** option allows users to export all their current environment variables, as of the time the
31745 batch job is submitted, to the context of the processes of the batch job.

31746 The **-v** option allows users to export specific environment variables from their current process
31747 to the processes of the batch job.

31748 The **-z** option allows users to suppress the writing of the batch job identifier to standard output.
31749 The **-z** option is an existing NQS practice that has been standardized.

31750 Historically, the *qsub* utility has served the batch job-submission function in the NQS system, the
31751 existing practice on which it is based. Some changes and additions have been made to the *qsub*
31752 utility in this volume of IEEE Std 1003.1-2001, *vis-a-vis* NQS, as a result of the growing pool of
31753 experience with distributed batch systems.

31754 The set of features of the *qsub* utility as defined in this volume of IEEE Std 1003.1-2001 appears to
31755 incorporate all the common existing practice on potentially conforming platforms.

31756 **FUTURE DIRECTIONS**

31757 None.

31758 **SEE ALSO**31759 Chapter 3 (on page 101), *qrerun*, *qstat*, *touch*31760 **CHANGE HISTORY**

31761 Derived from IEEE Std 1003.2d-1994.

31762 **Issue 6**31763 The **-l** option has been removed as there is no portable description of the resources that are
31764 allowed or required by the batch job.

31765 NAME

31766 *read* — read a line from standard input

31767 SYNOPSIS

31768 *read [-r] var...*

31769 DESCRIPTION

31770 The *read* utility shall read a single line from standard input.

31771 By default, unless the *-r* option is specified, backslash ('\\') shall act as an escape character, as
31772 described in Section 2.2.1 (on page 30). If standard input is a terminal device and the invoking
31773 shell is interactive, *read* shall prompt for a continuation line when:

- 31774 • The shell reads an input line ending with a backslash, unless the *-r* option is specified.
- 31775 • A here-document is not terminated after a <newline> is entered.

31776 The line shall be split into fields as in the shell (see Section 2.6.5 (on page 42)); the first field shall
31777 be assigned to the first variable *var*, the second field to the second variable *var*, and so on. If
31778 there are fewer *var* operands specified than there are fields, the leftover fields and their
31779 intervening separators shall be assigned to the last *var*. If there are fewer fields than *vars*, the
31780 remaining *vars* shall be set to empty strings.

31781 The setting of variables specified by the *var* operands shall affect the current shell execution
31782 environment; see Section 2.12 (on page 61). If it is called in a subshell or separate utility
31783 execution environment, such as one of the following:

31784 (read foo)
31785 nohup read ...
31786 find . -exec read ... \\;

31787 it shall not affect the shell variables in the caller's environment.

31788 OPTIONS

31789 The *read* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section
31790 12.2, Utility Syntax Guidelines.

31791 The following option is supported:

31792 **-r** Do not treat a backslash character in any special way. Consider each backslash to
31793 be part of the input line.

31794 OPERANDS

31795 The following operand shall be supported:

31796 *var* The name of an existing or nonexisting shell variable.

31797 STDIN

31798 The standard input shall be a text file.

31799 INPUT FILES

31800 None.

31801 ENVIRONMENT VARIABLES

31802 The following environment variables shall affect the execution of *read*:

31803 *IFS* Determine the internal field separators used to delimit fields; see Section 2.5.3 (on
31804 page 34).

31805 *LANG* Provide a default value for the internationalization variables that are unset or null.
31806 (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2,
31807 Internationalization Variables for the precedence of internationalization variables

31808	used to determine the values of locale categories.)	
31809	<i>LC_ALL</i>	If set to a non-empty string value, override the values of all the other internationalization variables.
31810		
31811	<i>LC_CTYPE</i>	Determine the locale for the interpretation of sequences of bytes of text data as characters (for example, single-byte as opposed to multi-byte characters in arguments).
31812		
31813		
31814	<i>LC_MESSAGES</i>	
31815		Determine the locale that should be used to affect the format and contents of diagnostic messages written to standard error.
31816		
31817 XSI	<i>NLSPATH</i>	Determine the location of message catalogs for the processing of <i>LC_MESSAGES</i> .
31818	<i>PS2</i>	Provide the prompt string that an interactive shell shall write to standard error when a line ending with a backslash is read and the <i>-r</i> option was not specified, or if a here-document is not terminated after a <newline> is entered.
31819		
31820		
31821	ASYNCHRONOUS EVENTS	
31822		Default.
31823	STDOUT	
31824		Not used.
31825	STDERR	
31826		The standard error shall be used for diagnostic messages and prompts for continued input.
31827	OUTPUT FILES	
31828		None.
31829	EXTENDED DESCRIPTION	
31830		None.
31831	EXIT STATUS	
31832		The following exit values shall be returned:
31833		0 Successful completion.
31834		>0 End-of-file was detected or an error occurred.
31835	CONSEQUENCES OF ERRORS	
31836		Default.
31837	APPLICATION USAGE	
31838		The <i>-r</i> option is included to enable <i>read</i> to subsume the purpose of the <i>line</i> utility, which is not included in IEEE Std 1003.1-2001.
31839		
31840		The results are undefined if an end-of-file is detected following a backslash at the end of a line when <i>-r</i> is not specified.
31841		
31842	EXAMPLES	
31843		The following command:
31844		while read -r xx yy
31845		do
31846		printf "%s %s\n" "\$yy" "\$xx"
31847		done < <i>input_file</i>
31848		prints a file with the first field of each line moved to the end of the line.

31849 **RATIONALE**

31850 The *read* utility historically has been a shell built-in. It was separated off into its own utility to
31851 take advantage of the richer description of functionality introduced by this volume of
31852 IEEE Std 1003.1-2001.

31853 Since *read* affects the current shell execution environment, it is generally provided as a shell
31854 regular built-in. If it is called in a subshell or separate utility execution environment, such as one
31855 of the following:

31856 (read foo)
31857 nohup read ...
31858 find . -exec read ... \\;

31859 it does not affect the shell variables in the environment of the caller.

31860 **FUTURE DIRECTIONS**

31861 None.

31862 **SEE ALSO**

31863 Chapter 2 (on page 29)

31864 **CHANGE HISTORY**

31865 First released in Issue 2.

31866 NAME

31867 renice — set nice values of running processes

31868 SYNOPSIS

31869 UP `renice -n increment [-g | -p | -u] ID ...`

31870

31871 DESCRIPTION

31872 The *renice* utility shall request that the nice values (see the Base Definitions volume of
31873 IEEE Std 1003.1-2001, Section 3.239, Nice Value) of one or more running processes be changed.
31874 By default, the applicable processes are specified by their process IDs. When a process group is
31875 specified (see **-g**), the request shall apply to all processes in the process group.

31876 The nice value shall be bounded in an implementation-defined manner. If the requested
31877 *increment* would raise or lower the nice value of the executed utility beyond implementation-
31878 defined limits, then the limit whose value was exceeded shall be used.

31879 When a user is *reniced*, the request applies to all processes whose saved set-user-ID matches the
31880 user ID corresponding to the user.

31881 Regardless of which options are supplied or any other factor, *renice* shall not alter the nice values
31882 of any process unless the user requesting such a change has appropriate privileges to do so for
31883 the specified process. If the user lacks appropriate privileges to perform the requested action, the
31884 utility shall return an error status.

31885 The saved set-user-ID of the user's process shall be checked instead of its effective user ID when
31886 *renice* attempts to determine the user ID of the process in order to determine whether the user
31887 has appropriate privileges.

31888 OPTIONS

31889 The *renice* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section
31890 12.2, Utility Syntax Guidelines.

31891 The following options shall be supported:

31892 **-g** Interpret all operands as unsigned decimal integer process group IDs.

31893 **-n increment** Specify how the nice value of the specified process or processes is to be adjusted.
31894 The *increment* option-argument is a positive or negative decimal integer that shall
31895 be used to modify the nice value of the specified process or processes.

31896 Positive *increment* values shall cause a lower nice value. Negative *increment* values
31897 may require appropriate privileges and shall cause a higher nice value.

31898 **-p** Interpret all operands as unsigned decimal integer process IDs. The **-p** option is
31899 the default if no options are specified.

31900 **-u** Interpret all operands as users. If a user exists with a user name equal to the
31901 operand, then the user ID of that user is used in further processing. Otherwise, if
31902 the operand represents an unsigned decimal integer, it shall be used as the numeric
31903 user ID of the user.

31904 OPERANDS

31905 The following operands shall be supported:

31906 **ID** A process ID, process group ID, or user name/user ID, depending on the option
31907 selected.

31908 STDIN

31909 Not used.

31910 INPUT FILES

31911 None.

31912 ENVIRONMENT VARIABLES

31913 The following environment variables shall affect the execution of *renice*:

31914 *LANG* Provide a default value for the internationalization variables that are unset or null.
(See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2, Internationalization Variables for the precedence of internationalization variables used to determine the values of locale categories.)

31918 *LC_ALL* If set to a non-empty string value, override the values of all the other internationalization variables.

31920 *LC_CTYPE* Determine the locale for the interpretation of sequences of bytes of text data as characters (for example, single-byte as opposed to multi-byte characters in arguments).

31923 *LC_MESSAGES*

31924 Determine the locale that should be used to affect the format and contents of diagnostic messages written to standard error.

31926 XSI *NLSPATH* Determine the location of message catalogs for the processing of *LC_MESSAGES*.

31927 ASYNCHRONOUS EVENTS

31928 Default.

31929 STDOUT

31930 Not used.

31931 STDERR

31932 The standard error shall be used only for diagnostic messages.

31933 OUTPUT FILES

31934 None.

31935 EXTENDED DESCRIPTION

31936 None.

31937 EXIT STATUS

31938 The following exit values shall be returned:

31939 0 Successful completion.

31940 >0 An error occurred.

31941 CONSEQUENCES OF ERRORS

31942 Default.

31943 APPLICATION USAGE

31944 None.

31945 EXAMPLES

31946 1. Adjust the nice value so that process IDs 987 and 32 would have a lower nice value:

31947 `renice -n 5 -p 987 32`31948 2. Adjust the nice value so that group IDs 324 and 76 would have a higher nice value, if the
31949 user has the appropriate privileges to do so:31950 `renice -n -4 -g 324 76`31951 3. Adjust the nice value so that numeric user ID 8 and user **sas** would have a lower nice
31952 value:31953 `renice -n 4 -u 8 sas`31954 Useful nice value increments on historical systems include 19 or 20 (the affected processes run
31955 only when nothing else in the system attempts to run) and any negative number (to make
31956 processes run faster).

31957 RATIONALE

31958 The *gid*, *pid*, and *user* specifications do not fit either the definition of operand or option-
31959 argument. However, for clarity, they have been included in the OPTIONS section, rather than
31960 the OPERANDS section.31961 The definition of nice value is not intended to suggest that all processes in a system have
31962 priorities that are comparable. Scheduling policy extensions such as the realtime priorities in the
31963 System Interfaces volume of IEEE Std 1003.1-2001 make the notion of a single underlying
31964 priority for all scheduling policies problematic. Some implementations may implement the *nice*-
31965 related features to affect all processes on the system, others to affect just the general time-
31966 sharing activities implied by this volume of IEEE Std 1003.1-2001, and others may have no effect
31967 at all. Because of the use of “implementation-defined” in *nice* and *renice*, a wide range of
31968 implementation strategies are possible.31969 Originally, this utility was written in the historical manner, using the term “nice value”. This
31970 was always a point of concern with users because it was never intuitively obvious what this
31971 meant. With a newer version of *renice*, which used the term “system scheduling priority”, it was
31972 hoped that novice users could better understand what this utility was meant to do. Also, it
31973 would be easier to document what the utility was meant to do. Unfortunately, the addition of
31974 the POSIX realtime scheduling capabilities introduced the concepts of process and thread
31975 scheduling priorities that were totally unaffected by the *nice*/*renice* utilities or the
31976 *nice()*/*setpriority()* functions. Continuing to use the term “system scheduling priority” would
31977 have incorrectly suggested that these utilities and functions were indeed affecting these realtime
31978 priorities. It was decided to revert to the historical term “nice value” to reference this unrelated
31979 process attribute.31980 Although this utility has use by system administrators (and in fact appears in the system
31981 administration portion of the BSD documentation), the standard developers considered that it
31982 was very useful for individual end users to control their own processes.

31983 FUTURE DIRECTIONS

31984 None.

31985 **SEE ALSO**

31986 *nice*

31987 **CHANGE HISTORY**

31988 First released in Issue 4.

31989 **Issue 5**

31990 In the SYNOPSIS, an ellipsis is added to the **-u** option in all three obsolescent forms.

31991 **Issue 6**

31992 This utility is marked as part of the User Portability Utilities option.

31993 The APPLICATION USAGE section is added.

31994 The obsolescent forms of the SYNOPSIS are removed.

31995 Text previously conditional on **POSIX_SAVED_IDS** is mandatory in this issue. This is a FIPS requirement.
31996

31997 NAME

31998 rm — remove directory entries

31999 SYNOPSIS

32000 rm [-f i R r] file...

32001 DESCRIPTION

32002 The *rm* utility shall remove the directory entry specified by each *file* argument.32003 If either of the files dot or dot-dot are specified as the basename portion of an operand (that is, 32004 the final pathname component), *rm* shall write a diagnostic message to standard error and do 32005 nothing more with such operands.32006 For each *file* the following steps shall be taken:32007 1. If the *file* does not exist:

- 32008 a. If the
- f**
- option is not specified,
- rm*
- shall write a diagnostic message to standard error.
-
- 32009 b. Go on to any remaining
- files*
- .

32010 2. If *file* is of type directory, the following steps shall be taken:

- 32011 a. If neither the
- R**
- option nor the
- r**
- option is specified,
- rm*
- shall write a diagnostic 32012 message to standard error, do nothing more with
- file*
- , and go on to any remaining 32013 files.

- 32014 b. If the
- f**
- option is not specified, and either the permissions of
- file*
- do not permit 32015 writing and the standard input is a terminal or the
- i**
- option is specified,
- rm*
- shall 32016 write a prompt to standard error and read a line from the standard input. If the 32017 response is not affirmative,
- rm*
- shall do nothing more with the current file and go on 32018 to any remaining files.

- 32019 c. For each entry contained in
- file*
- , other than dot or dot-dot, the four steps listed here (1 32020 to 4) shall be taken with the entry as if it were a
- file*
- operand. The
- rm*
- utility shall not 32021 traverse directories by following symbolic links into other parts of the hierarchy, but 32022 shall remove the links themselves.

- 32023 d. If the
- i**
- option is specified,
- rm*
- shall write a prompt to standard error and read a line 32024 from the standard input. If the response is not affirmative,
- rm*
- shall do nothing more 32025 with the current file, and go on to any remaining files.

- 32026 3. If
- file*
- is not of type directory, the
- f**
- option is not specified, and either the permissions of 32027
- file*
- do not permit writing and the standard input is a terminal or the
- i**
- option is specified, 32028
- rm*
- shall write a prompt to the standard error and read a line from the standard input. If the 32029 response is not affirmative,
- rm*
- shall do nothing more with the current file and go on to any 32030 remaining files.

- 32031 4. If the current file is a directory,
- rm*
- shall perform actions equivalent to the
- rmdir()*
- function 32032 defined in the System Interfaces volume of IEEE Std 1003.1-2001 called with a pathname of 32033 the current file used as the
- path*
- argument. If the current file is not a directory,
- rm*
- shall 32034 perform actions equivalent to the
- unlink()*
- function defined in the System Interfaces 32035 volume of IEEE Std 1003.1-2001 called with a pathname of the current file used as the
- path*
- 32036 argument.

32037 If this fails for any reason, *rm* shall write a diagnostic message to standard error, do 32038 nothing more with the current file, and go on to any remaining files.32039 The *rm* utility shall be able to descend to arbitrary depths in a file hierarchy, and shall not fail 32040 due to path length limitations (unless an operand specified by the user exceeds system

32041 limitations).

32042 OPTIONS

32043 The *rm* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section 12.2,
32044 Utility Syntax Guidelines.

32045 The following options shall be supported:

- 32046 **-f** Do not prompt for confirmation. Do not write diagnostic messages or modify the
32047 exit status in the case of nonexistent operands. Any previous occurrences of the **-i**
32048 option shall be ignored.
- 32049 **-i** Prompt for confirmation as described previously. Any previous occurrences of the
32050 **-f** option shall be ignored.
- 32051 **-R** Remove file hierarchies. See the DESCRIPTION.
- 32052 **-r** Equivalent to **-R**.

32053 OPERANDS

32054 The following operand shall be supported:

- 32055 *file* A pathname of a directory entry to be removed.

32056 STDIN

32057 The standard input shall be used to read an input line in response to each prompt specified in
32058 the STDOUT section. Otherwise, the standard input shall not be used.

32059 INPUT FILES

32060 None.

32061 ENVIRONMENT VARIABLES

32062 The following environment variables shall affect the execution of *rm*:

- 32063 **LANG** Provide a default value for the internationalization variables that are unset or null.
(See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2,
32065 Internationalization Variables for the precedence of internationalization variables
32066 used to determine the values of locale categories.)
- 32067 **LC_ALL** If set to a non-empty string value, override the values of all the other
32068 internationalization variables.
- 32069 **LC_COLLATE** Determine the locale for the behavior of ranges, equivalence classes, and multi-
32070 character collating elements used in the extended regular expression defined for
32071 the **yesexpr** locale keyword in the **LC_MESSAGES** category.
- 32073 **LC_CTYPE** Determine the locale for the interpretation of sequences of bytes of text data as
32074 characters (for example, single-byte as opposed to multi-byte characters in
32075 arguments) and the behavior of character classes within regular expressions used
32076 in the extended regular expression defined for the **yesexpr** locale keyword in the
32077 **LC_MESSAGES** category.
- 32078 **LC_MESSAGES** Determine the locale for the processing of affirmative responses that should be
32079 used to affect the format and contents of diagnostic messages written to standard
32080 error.
- 32082 XSI **NLSPATH** Determine the location of message catalogs for the processing of **LC_MESSAGES**.

32083 ASYNCHRONOUS EVENTS

32084 Default.

32085 STDOUT

32086 Not used.

32087 STDERR

32088 Prompts shall be written to standard error under the conditions specified in the DESCRIPTION and OPTIONS sections. The prompts shall contain the *file* pathname, but their format is otherwise unspecified. The standard error also shall be used for diagnostic messages.

32091 OUTPUT FILES

32092 None.

32093 EXTENDED DESCRIPTION

32094 None.

32095 EXIT STATUS

32096 The following exit values shall be returned:

32097 0 All of the named directory entries for which *rm* performed actions equivalent to the *rmdir()* or *unlink()* functions were removed.

32099 >0 An error occurred.

32100 CONSEQUENCES OF ERRORS

32101 Default.

32102 APPLICATION USAGE

32103 The *rm* utility is forbidden to remove the names dot and dot-dot in order to avoid the consequences of inadvertently doing something like:

32105 *rm -r .**

32106 Some implementations do not permit the removal of the last link to an executable binary file that
32107 is being executed; see the [EBUSY] error in the *unlink()* function defined in the System Interfaces
32108 volume of IEEE Std 1003.1-2001. Thus, the *rm* utility can fail to remove such files.

32109 The *-i* option causes *rm* to prompt and read the standard input even if the standard input is not
32110 a terminal, but in the absence of *-i* the mode prompting is not done when the standard input is
32111 not a terminal.

32112 EXAMPLES

32113 1. The following command:

32114 *rm a.out core*

32115 removes the directory entries: **a.out** and **core**.

32116 2. The following command:

32117 *rm -Rf junk*

32118 removes the directory **junk** and all its contents, without prompting.

32119 RATIONALE

32120 For absolute clarity, paragraphs (2b) and (3) in the DESCRIPTION of *rm* describing the behavior
32121 when prompting for confirmation, should be interpreted in the following manner:

32122 *if ((NOT f_option) AND
(not_writable AND input_is_terminal) OR i_option))*

32124 The exact format of the interactive prompts is unspecified. Only the general nature of the
32125 contents of prompts are specified because implementations may desire more descriptive
32126 prompts than those used on historical implementations. Therefore, an application not using the
32127 **-f** option, or using the **-i** option, relies on the system to provide the most suitable dialog directly
32128 with the user, based on the behavior specified.

32129 The **-r** option is historical practice on all known systems. The synonym **-R** option is provided
32130 for consistency with the other utilities in this volume of IEEE Std 1003.1-2001 that provide
32131 options requesting recursive descent through the file hierarchy.

32132 The behavior of the **-f** option in historical versions of *rm* is inconsistent. In general, along with
32133 “forcing” the unlink without prompting for permission, it always causes diagnostic messages to
32134 be suppressed and the exit status to be unmodified for nonexistent operands and files that
32135 cannot be unlinked. In some versions, however, the **-f** option suppresses usage messages and
32136 system errors as well. Suppressing such messages is not a service to either shell scripts or users.

32137 It is less clear that error messages regarding files that cannot be unlinked (removed) should be
32138 suppressed. Although this is historical practice, this volume of IEEE Std 1003.1-2001 does not
32139 permit the **-f** option to suppress such messages.

32140 When given the **-r** and **-i** options, historical versions of *rm* prompt the user twice for each
32141 directory, once before removing its contents and once before actually attempting to delete the
32142 directory entry that names it. This allows the user to “prune” the file hierarchy walk. Historical
32143 versions of *rm* were inconsistent in that some did not do the former prompt for directories
32144 named on the command line and others had obscure prompting behavior when the **-i** option
32145 was specified and the permissions of the file did not permit writing. The POSIX Shell and
32146 Utilities *rm* differs little from historic practice, but does require that prompts be consistent.
32147 Historical versions of *rm* were also inconsistent in that prompts were done to both standard
32148 output and standard error. This volume of IEEE Std 1003.1-2001 requires that prompts be done
32149 to standard error, for consistency with *cp* and *mv*, and to allow historical extensions to *rm* that
32150 provide an option to list deleted files on standard output.

32151 The *rm* utility is required to descend to arbitrary depths so that any file hierarchy may be
32152 deleted. This means, for example, that the *rm* utility cannot run out of file descriptors during its
32153 descent (that is, if the number of file descriptors is limited, *rm* cannot be implemented in the
32154 historical fashion where one file descriptor is used per directory level). Also, *rm* is not permitted
32155 to fail because of path length restrictions, unless an operand specified by the user is longer than
32156 **{PATH_MAX}**.

32157 The *rm* utility removes symbolic links themselves, not the files they refer to, as a consequence of
32158 the dependence on the *unlink()* functionality, per the DESCRIPTION. When removing
32159 hierarchies with **-r** or **-R**, the prohibition on following symbolic links has to be made explicit.

32160 FUTURE DIRECTIONS

32161 None.

32162 SEE ALSO

32163 *rmdir*, the System Interfaces volume of IEEE Std 1003.1-2001, *remove()*, *rmdir()*, *unlink()*

32164 CHANGE HISTORY

32165 First released in Issue 2.

32166 Issue 5

32167 The FUTURE DIRECTIONS section is added.

32168 Issue 6

32169 Text is added to clarify actions relating to symbolic links as specified in the IEEE P1003.2b draft
32170 standard.

32171 NAME

32172 **rmdel** — remove a delta from an SCCS file (**DEVELOPMENT**)

32173 SYNOPSIS

32174 XSI **rmdel -r SID file...**

32175

32176 DESCRIPTION

32177 The *rmdel* utility shall remove the delta specified by the SID from each named SCCS file. The
32178 delta to be removed shall be the most recent delta in its branch in the delta chain of each named
32179 SCCS file. In addition, the application shall ensure that the SID specified is not that of a version
32180 being edited for the purpose of making a delta; that is, if a *p-file* (see *get*) exists for the named
32181 SCCS file, the SID specified shall not appear in any entry of the *p-file*.

32182 Removal of a delta shall be restricted to:

- 32183 1. The user who made the delta
- 32184 2. The owner of the SCCS file
- 32185 3. The owner of the directory containing the SCCS file

32186 OPTIONS

32187 The *rmdel* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section
32188 12.2, Utility Syntax Guidelines.

32189 The following option shall be supported:

32190 **-r SID** Specify the SCCS identification string (*SID*) of the delta to be deleted.

32191 OPERANDS

32192 The following operand shall be supported:

32193 *file* A pathname of an existing SCCS file or a directory. If *file* is a directory, the *rmdel*
32194 utility shall behave as though each file in the directory were specified as a named
32195 file, except that non-SCCS files (last component of the pathname does not begin
32196 with s.) and unreadable files shall be silently ignored.

32197 If exactly one *file* operand appears, and it is ‘-’, the standard input shall be read;
32198 each line of the standard input is taken to be the name of an SCCS file to be
32199 processed. Non-SCCS files and unreadable files shall be silently ignored.

32200 STDIN

32201 The standard input shall be a text file used only when the *file* operand is specified as ‘-’. Each
32202 line of the text file shall be interpreted as an SCCS pathname.

32203 INPUT FILES

32204 The SCCS files shall be files of unspecified format.

32205 ENVIRONMENT VARIABLES

32206 The following environment variables shall affect the execution of *rmdel*:

32207 *LANG* Provide a default value for the internationalization variables that are unset or null.
32208 (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2,
32209 Internationalization Variables for the precedence of internationalization variables
32210 used to determine the values of locale categories.)

32211 *LC_ALL* If set to a non-empty string value, override the values of all the other
32212 internationalization variables.

32213	<i>LC_CTYPE</i>	Determine the locale for the interpretation of sequences of bytes of text data as characters (for example, single-byte as opposed to multi-byte characters in arguments and input files).
32216	<i>LC_MESSAGES</i>	Determine the locale that should be used to affect the format and contents of diagnostic messages written to standard error.
32219	<i>NLSPATH</i>	Determine the location of message catalogs for the processing of <i>LC_MESSAGES</i> .
32220	ASYNCHRONOUS EVENTS	
32221		Default.
32222	STDOUT	
32223		Not used.
32224	STDERR	
32225		The standard error shall be used only for diagnostic messages.
32226	OUTPUT FILES	
32227		The SCCS files shall be files of unspecified format. During processing of a <i>file</i> , a temporary <i>x-file</i> , as described in <i>admin</i> , may be created and deleted; a locking <i>z-file</i> , as described in <i>get</i> , may be created and deleted.
32230	EXTENDED DESCRIPTION	
32231		None.
32232	EXIT STATUS	
32233		The following exit values shall be returned:
32234		0 Successful completion.
32235		>0 An error occurred.
32236	CONSEQUENCES OF ERRORS	
32237		Default.
32238	APPLICATION USAGE	
32239		None.
32240	EXAMPLES	
32241		None.
32242	RATIONALE	
32243		None.
32244	FUTURE DIRECTIONS	
32245		None.
32246	SEE ALSO	
32247		<i>admin, delta, get, prs</i>
32248	CHANGE HISTORY	
32249		First released in Issue 2.
32250	Issue 6	
32251		The normative text is reworded to avoid use of the term “must” for application requirements.

32252 NAME

32253 **rmdir** — remove directories

32254 SYNOPSIS

32255 **rmdir [-p] dir...**

32256 DESCRIPTION

32257 The *rmdir* utility shall remove the directory entry specified by each *dir* operand.

32258 For each *dir* operand, the *rmdir* utility shall perform actions equivalent to the *rmdir()* function called with the *dir* operand as its only argument.

32260 Directories shall be processed in the order specified. If a directory and a subdirectory of that
32261 directory are specified in a single invocation of the *rmdir* utility, the application shall specify the
32262 subdirectory before the parent directory so that the parent directory will be empty when the
32263 *rmdir* utility tries to remove it.

32264 OPTIONS

32265 The *rmdir* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section
32266 12.2, Utility Syntax Guidelines.

32267 The following option shall be supported:

32268 **-p** Remove all directories in a pathname. For each *dir* operand:

32269 1. The directory entry it names shall be removed.

32270 2. If the *dir* operand includes more than one pathname component, effects
32271 equivalent to the following command shall occur:

32272 **rmdir -p \$(dirname dir)**

32273 OPERANDS

32274 The following operand shall be supported:

32275 *dir* A pathname of an empty directory to be removed.

32276 STDIN

32277 Not used.

32278 INPUT FILES

32279 None.

32280 ENVIRONMENT VARIABLES

32281 The following environment variables shall affect the execution of *rmdir*:

32282 **LANG** Provide a default value for the internationalization variables that are unset or null.
32283 (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2,
32284 Internationalization Variables for the precedence of internationalization variables
32285 used to determine the values of locale categories.)

32286 **LC_ALL** If set to a non-empty string value, override the values of all the other
32287 internationalization variables.

32288 **LC_CTYPE** Determine the locale for the interpretation of sequences of bytes of text data as
32289 characters (for example, single-byte as opposed to multi-byte characters in
32290 arguments).

32291 **LC_MESSAGES**

32292 Determine the locale that should be used to affect the format and contents of
32293 diagnostic messages written to standard error.

32294 XSI **NLSPATH** Determine the location of message catalogs for the processing of *LC_MESSAGES*.

32295 **ASYNCHRONOUS EVENTS**

32296 Default.

32297 **STDOUT**

32298 Not used.

32299 **STDERR**

32300 The standard error shall be used only for diagnostic messages.

32301 **OUTPUT FILES**

32302 None.

32303 **EXTENDED DESCRIPTION**

32304 None.

32305 **EXIT STATUS**

32306 The following exit values shall be returned:

32307 0 Each directory entry specified by a *dir* operand was removed successfully.

32308 >0 An error occurred.

32309 **CONSEQUENCES OF ERRORS**

32310 Default.

32311 **APPLICATION USAGE**

32312 The definition of an empty directory is one that contains, at most, directory entries for dot and
32313 dot-dot.

32314 **EXAMPLES**

32315 If a directory **a** in the current directory is empty except it contains a directory **b** and **a/b** is empty
32316 except it contains a directory **c**:

32317 `rmdir -p a/b/c`

32318 removes all three directories.

32319 **RATIONALE**

32320 On historical System V systems, the **-p** option also caused a message to be written to the
32321 standard output. The message indicated whether the whole path was removed or whether part
32322 of the path remained for some reason. The **STDERR** section requires this diagnostic when the
32323 entire path specified by a *dir* operand is not removed, but does not allow the status message
32324 reporting success to be written as a diagnostic.

32325 The *rmdir* utility on System V also included a **-s** option that suppressed the informational
32326 message output by the **-p** option. This option has been omitted because the informational
32327 message is not specified by this volume of IEEE Std 1003.1-2001.

32328 **FUTURE DIRECTIONS**

32329 None.

32330 **SEE ALSO**

32331 *rm*, the System Interfaces volume of IEEE Std 1003.1-2001, *remove()*, *rmdir()*, *unlink()*

32332 **CHANGE HISTORY**

32333 First released in Issue 2.

32334 Issue 6

32335 The normative text is reworded to avoid use of the term “must” for application requirements.

32336 NAME

32337 sact — print current SCCS file-editing activity (**DEVELOPMENT**)

32338 SYNOPSIS

32339 XSI sact *file...*

32340

32341 DESCRIPTION

32342 The *sact* utility shall inform the user of any impending deltas to a named SCCS file by writing a
32343 list to standard output. This situation occurs when *get -e* has been executed previously without
32344 a subsequent execution of *delta*, *unget*, or *scs unedit*.

32345 OPTIONS

32346 None.

32347 OPERANDS

32348 The following operand shall be supported:

32349 *file* A pathname of an existing SCCS file or a directory. If *file* is a directory, the *sact*
32350 utility shall behave as though each file in the directory were specified as a named
32351 file, except that non-SCCS files (last component of the pathname does not begin
32352 with **s**.) and unreadable files shall be silently ignored.

32353 If exactly one *file* operand appears, and it is '**-**', the standard input shall be read;
32354 each line of the standard input shall be taken to be the name of an SCCS file to be
32355 processed. Non-SCCS files and unreadable files shall be silently ignored.

32356 STDIN

32357 The standard input shall be a text file used only when the *file* operand is specified as '**-**'. Each
32358 line of the text file shall be interpreted as an SCCS pathname.

32359 INPUT FILES

32360 Any SCCS files interrogated are files of an unspecified format.

32361 ENVIRONMENT VARIABLES

32362 The following environment variables shall affect the execution of *sact*:

32363 *LANG* Provide a default value for the internationalization variables that are unset or null.
32364 (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2,
32365 Internationalization Variables for the precedence of internationalization variables
32366 used to determine the values of locale categories.)

32367 *LC_ALL* If set to a non-empty string value, override the values of all the other
32368 internationalization variables.

32369 *LC_CTYPE* Determine the locale for the interpretation of sequences of bytes of text data as
32370 characters (for example, single-byte as opposed to multi-byte characters in
32371 arguments and input files).

32372 *LC_MESSAGES*

32373 Determine the locale that should be used to affect the format and contents of
32374 diagnostic messages written to standard error.

32375 *NLSPATH* Determine the location of message catalogs for the processing of *LC_MESSAGES*.

32376 ASYNCHRONOUS EVENTS

32377 Default.

32378 STDOUT

32379 The output for each named file shall consist of a line in the following format:

32380 "`%sΔ%Δ%Δ%Δ%Δ\n`", `<SID>`, `<new SID>`, `<login>`, `<date>`, `<time>`

32381 `<SID>` Specifies the SID of a delta that currently exists in the SCCS file to which changes
32382 are made to make the new delta.

32383 `<new SID>` Specifies the SID for the new delta to be created.

32384 `<login>` Contains the login name of the user who makes the delta (that is, who executed a
32385 *get* for editing).

32386 `<date>` Contains the date that *get -e* was executed, in the format used by the *prs :D:* data
32387 keyword.

32388 `<time>` Contains the time that *get -e* was executed, in the format used by the *prs :T:* data
32389 keyword.

32390 If there is more than one named file or if a directory or standard input is named, each pathname
32391 shall be written before each of the preceding lines:

32392 "`\n%s:\n`", `<pathname>`

32393 STDERR

32394 The standard error shall be used only for optional informative messages concerning SCCS files
32395 with no impending deltas, and for diagnostic messages.

32396 OUTPUT FILES

32397 None.

32398 EXTENDED DESCRIPTION

32399 None.

32400 EXIT STATUS

32401 The following exit values shall be returned:

32402 0 Successful completion.

32403 >0 An error occurred.

32404 CONSEQUENCES OF ERRORS

32405 Default.

32406 APPLICATION USAGE

32407 None.

32408 EXAMPLES

32409 None.

32410 RATIONALE

32411 None.

32412 FUTURE DIRECTIONS

32413 None.

32414 SEE ALSO

32415 *delta, get, sccs, unget*

32416 CHANGE HISTORY

32417 First released in Issue 2.

32418 NAME

32419 sccs — front end for the SCCS subsystem (**DEVELOPMENT**)

32420 SYNOPSIS

32421 XSI **sccs [-r][-d path][-p path] command [options...][operands...]**

32422

32423 DESCRIPTION

32424 The *sccs* utility is a front end to the SCCS programs. It also includes the capability to run set-user-id to another user to provide additional protection.32426 The *sccs* utility shall invoke the specified *command* with the specified *options* and *operands*. By
32427 default, each of the *operands* shall be modified by prefixing it with the string "SCCS/s.".32428 The *command* can be the name of one of the SCCS utilities in this volume of IEEE Std 1003.1-2001
32429 (*admin*, *delta*, *get*, *prs*, *rmdel*, *sact*, *unget*, *val*, or *what*) or one of the pseudo-utilities listed in the
32430 EXTENDED DESCRIPTION section.

32431 OPTIONS

32432 The *sccs* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section
32433 12.2, Utility Syntax Guidelines, except that *options* operands are actually options to be passed to
32434 the utility named by *command*. When the portion of the command:32435 *command* [*options* ...] [*operands* ...]32436 is considered, all of the pseudo-utilities used as *command* shall support the Utility Syntax
32437 Guidelines. Any of the other SCCS utilities that can be invoked in this manner support the
32438 Guidelines to the extent indicated by their individual OPTIONS sections.32439 The following options shall be supported preceding the *command* operand:32440 **-d path** A pathname of a directory to be used as a root directory for the SCCS files. The
32441 default shall be the current directory. The **-d** option shall take precedence over the
32442 *PROJECTDIR* variable. See **-p**.32443 **-p path** A pathname of a directory in which the SCCS files are located. The default shall be
32444 the **SCCS** directory.32445 The **-p** option differs from the **-d** option in that the **-d** option-argument shall be
32446 prefixed to the entire pathname and the **-p** option-argument shall be inserted
32447 before the final component of the pathname. For example:32448 *sccs -d /x -p y get a/b*

32449 converts to:

32450 *get /x/a/y/s.b*

32451 This allows the creation of aliases such as:

32452 *alias sysscscs="sccs -d /usr/src"*

32453 which is used as:

32454 *sysscscs get cmd/who.c*32455 **-r** Invoke *command* with the real user ID of the process, not any effective user ID that
32456 the *sccs* utility is set to. Certain commands (*admin*, *check*, *clean*, *diffs*, *info*, *rmdel*,
32457 and *tell*) cannot be run set-user-ID by all users, since this would allow anyone to
32458 change the authorizations. These commands are always run as the real user.

32459 **OPERANDS**

32460 The following operands shall be supported:

32461 *command* An SCCS utility name or the name of one of the pseudo-utilities listed in the
32462 EXTENDED DESCRIPTION section.

32463 *options* An option or option-argument to be passed to *command*.

32464 *operands* An operand to be passed to *command*.

32465 **STDIN**

32466 See the utility description for the specified *command*.

32467 **INPUT FILES**

32468 See the utility description for the specified *command*.

32469 **ENVIRONMENT VARIABLES**

32470 The following environment variables shall affect the execution of *sccs*:

32471 *LANG* Provide a default value for the internationalization variables that are unset or null.
32472 (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2,
32473 Internationalization Variables for the precedence of internationalization variables
32474 used to determine the values of locale categories.)

32475 *LC_ALL* If set to a non-empty string value, override the values of all the other
32476 internationalization variables.

32477 *LC_CTYPE* Determine the locale for the interpretation of sequences of bytes of text data as
32478 characters (for example, single-byte as opposed to multi-byte characters in
32479 arguments and input files).

32480 *LC_MESSAGES*

32481 Determine the locale that should be used to affect the format and contents of
32482 diagnostic messages written to standard error.

32483 *NLSPATH* Determine the location of message catalogs for the processing of *LC_MESSAGES*.

32484 *PROJECTDIR*

32485 Provide a default value for the **-d** *path* option. If the value of *PROJECTDIR* begins
32486 with a slash, it shall be considered an absolute pathname; otherwise, the value of
32487 *PROJECTDIR* is treated as a user name and that user's initial working directory
32488 shall be examined for a subdirectory **src** or **source**. If such a directory is found, it
32489 shall be used. Otherwise, the value shall be used as a relative pathname.

32490 Additional environment variable effects may be found in the utility description for the specified
32491 *command*.

32492 **ASYNCHRONOUS EVENTS**

32493 Default.

32494 **STDOUT**

32495 See the utility description for the specified *command*.

32496 **STDERR**

32497 See the utility description for the specified *command*.

32498 **OUTPUT FILES**

32499 See the utility description for the specified *command*.

32500 EXTENDED DESCRIPTION

- 32501 The following pseudo-utilities shall be supported as *command* operands. All options referred to
 32502 in the following list are values given in the *options* operands following *command*.
- 32503 **check** Equivalent to **info**, except that nothing shall be printed if nothing is being edited, and a
 32504 non-zero exit status shall be returned if anything is being edited. The intent is to have
 32505 this included in an “install” entry in a makefile to ensure that everything is included
 32506 into the SCCS file before a version is installed.
- 32507 **clean** Remove everything from the current directory that can be recreated from SCCS files,
 32508 but do not remove any files being edited. If the **-b** option is given, branches shall be
 32509 ignored in the determination of whether they are being edited; this is dangerous if
 32510 branches are kept in the same directory.
- 32511 **create** Create an SCCS file, taking the initial contents from the file of the same name. Any
 32512 options to *admin* are accepted. If the creation is successful, the original files shall be
 32513 renamed by prefixing the basenames with a comma. These renamed files should be
 32514 removed after it has been verified that the SCCS files have been created successfully.
- 32515 **delget** Perform a *delta* on the named files and then *get* new versions. The new versions shall
 32516 have ID keywords expanded and shall not be editable. Any **-m**, **-p**, **-r**, **-s**, and **-y**
 32517 options shall be passed to *delta*, and any **-b**, **-c**, **-e**, **-i**, **-k**, **-l**, **-s**, and **-x** options shall be
 32518 passed to *get*.
- 32519 **deledit** Equivalent to **delget**, except that the *get* phase shall include the **-e** option. This option
 32520 is useful for making a checkpoint of the current editing phase. The same options shall
 32521 be passed to *delta* as described above, and all the options listed for *get* above except **-e**
 32522 shall be passed to **edit**.
- 32523 **diffs** Write a difference listing between the current version of the files checked out for
 32524 editing and the versions in SCCS format. Any **-r**, **-c**, **-i**, **-x**, and **-t** options shall be
 32525 passed to *get*; any **-l**, **-s**, **-e**, **-f**, **-h**, and **-b** options shall be passed to *diff*. A **-C** option
 32526 shall be passed to *diff* as **-c**.
- 32527 **edit** Equivalent to *get -e*.
- 32528 **fix** Remove the named delta, but leave a copy of the delta with the changes that were in it.
 32529 It is useful for fixing small compiler bugs, and so on. The application shall ensure that it
 32530 is followed by a **-r SID** option. Since **fix** does not leave audit trails, it should be used
 32531 carefully.
- 32532 **info** Write a listing of all files being edited. If the **-b** option is given, branches (that is, SIDs
 32533 with two or fewer components) shall be ignored. If a **-u user** option is given, then only
 32534 files being edited by the named user shall be listed. A **-U** option shall be equivalent to
 32535 **-u<current user>**.
- 32536 **print** Write out verbose information about the named files, equivalent to *sccs prs*.
- 32537 **tell** Write a <newline>-separated list of the files being edited to standard output. Takes the
 32538 **-b**, **-u**, and **-U** options like **info** and **check**.
- 32539 **unedit** This is the opposite of an **edit** or a *get -e*. It should be used with caution, since any
 32540 changes made since the *get* are lost.

32541 EXIT STATUS

32542 The following exit values shall be returned:

32543 0 Successful completion.

32544 >0 An error occurred.

32545 CONSEQUENCES OF ERRORS

32546 Default.

32547 APPLICATION USAGE

32548 Many of the SCCS utilities take directory names as operands as well as specific filenames. The
32549 pseudo-utilities supported by sccs are not described as having this capability, but are not
32550 prohibited from doing so.

32551 EXAMPLES

32552 1. To get a file for editing, edit it and produce a new delta:

```
32553     sccs get -e file.c
32554     ex file.c
32555     sccs delta file.c
```

32556 2. To get a file from another directory:

```
32557     sccs -p /usr/src/sccs/s. get cc.c
```

32558 or:

```
32559     sccs get /usr/src/sccs/s.cc.c
```

32560 3. To make a delta of a large number of files in the current directory:

```
32561     sccs delta *.c
```

32562 4. To get a list of files being edited that are not on branches:

```
32563     sccs info -b
```

32564 5. To delta everything being edited by the current user:

```
32565     sccs delta $(sccs tell -U)
```

32566 6. In a makefile, to get source files from an SCCS file if it does not already exist:

```
32567     SRCS = <list of source files>
32568     $(SRCS):
32569         sccs get $(REL) $@
```

32570 RATIONALE

32571 SCCS and its associated utilities are part of the XSI Development Utilities option within the XSI
32572 extension.

32573 SCCS is an abbreviation for Source Code Control System. It is a maintenance and enhancement
32574 tracking tool. When a file is put under SCCS, the source code control system maintains the file
32575 and, when changes are made, identifies and stores them in the file with the original source code
32576 and/or documentation. As other changes are made, they too are identified and retained in the
32577 file.

32578 Retrieval of the original and any set of changes is possible. Any version of the file as it develops
32579 can be reconstructed for inspection or additional modification. History data can be stored with
32580 each version, documenting why the changes were made, who made them, and when they were
32581 made.

32582 FUTURE DIRECTIONS

32583 None.

32584 SEE ALSO

32585 *admin, delta, get, make, prs, rmdel, sact, unget, val, what*

32586 CHANGE HISTORY

32587 First released in Issue 4.

32588 Issue 6

32589 In the ENVIRONMENT VARIABLES section, the *PROJECTDIR* description is updated from
32590 “otherwise, the home directory of a user of that name is examined” to “otherwise, the value of
32591 *PROJECTDIR* is treated as a user name and that user’s initial working directory is examined”.

32592 The normative text is reworded to avoid use of the term “must” for application requirements.

32593 **NAME**

32594 *sed* — stream editor

32595 **SYNOPSIS**

32596 *sed* [*-n*] *script*[*file...*]

32597 *sed* [*-n*][*-e* *script*]...[*-f* *script_file*]...[*file...*]

32598 **DESCRIPTION**

32599 The *sed* utility is a stream editor that shall read one or more text files, make editing changes
32600 according to a script of editing commands, and write the results to standard output. The script
32601 shall be obtained from either the *script* operand string or a combination of the option-arguments
32602 from the *-e* *script* and *-f* *script_file* options.

32603 **OPTIONS**

32604 The *sed* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section 12.2,
32605 Utility Syntax Guidelines, except that the order of presentation of the *-e* and *-f* options is
32606 significant.

32607 The following options shall be supported:

32608 **-e** *script* Add the editing commands specified by the *script* option-argument to the end of
32609 the script of editing commands. The *script* option-argument shall have the same
32610 properties as the *script* operand, described in the OPERANDS section.

32611 **-f** *script_file* Add the editing commands in the file *script_file* to the end of the script.

32612 **-n** Suppress the default output (in which each line, after it is examined for editing, is
32613 written to standard output). Only lines explicitly selected for output are written.

32614 Multiple *-e* and *-f* options may be specified. All commands shall be added to the script in the
32615 order specified, regardless of their origin.

32616 **OPERANDS**

32617 The following operands shall be supported:

32618 *file* A pathname of a file whose contents are read and edited. If multiple *file* operands
32619 are specified, the named files shall be read in the order specified and the
32620 concatenation shall be edited. If no *file* operands are specified, the standard input
32621 shall be used.

32622 *script* A string to be used as the script of editing commands. The application shall not
32623 present a *script* that violates the restrictions of a text file except that the final
32624 character need not be a <newline>.

32625 **STDIN**

32626 The standard input shall be used only if no *file* operands are specified. See the INPUT FILES
32627 section.

32628 **INPUT FILES**

32629 The input files shall be text files. The *script_files* named by the *-f* option shall consist of editing
32630 commands.

32631 **ENVIRONMENT VARIABLES**

32632 The following environment variables shall affect the execution of *sed*:

32633 **LANG** Provide a default value for the internationalization variables that are unset or null.
32634 (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2,
32635 Internationalization Variables for the precedence of internationalization variables
32636 used to determine the values of locale categories.)

32637	<i>LC_ALL</i>	If set to a non-empty string value, override the values of all the other internationalization variables.
32639	<i>LC_COLLATE</i>	Determine the locale for the behavior of ranges, equivalence classes, and multi-character collating elements within regular expressions.
32642	<i>LC_CTYPE</i>	Determine the locale for the interpretation of sequences of bytes of text data as characters (for example, single-byte as opposed to multi-byte characters in arguments and input files), and the behavior of character classes within regular expressions.
32646	<i>LC_MESSAGES</i>	Determine the locale that should be used to affect the format and contents of diagnostic messages written to standard error.
32649 XSI	<i>NLSPATH</i>	Determine the location of message catalogs for the processing of <i>LC_MESSAGES</i> .
32650	ASYNCHRONOUS EVENTS	
32651		Default.
32652	STDOUT	
32653		The input files shall be written to standard output, with the editing commands specified in the script applied. If the -n option is specified, only those input lines selected by the script shall be written to standard output.
32656	STDERR	
32657		The standard error shall be used only for diagnostic messages.
32658	OUTPUT FILES	
32659		The output files shall be text files whose formats are dependent on the editing commands given.
32660	EXTENDED DESCRIPTION	
32661		The <i>script</i> shall consist of editing commands of the following form:
32662		<i>[address[,address]]function</i>
32663		where <i>function</i> represents a single-character command verb from the list in Editing Commands in sed (on page 844), followed by any applicable arguments.
32665		The command can be preceded by <blank>s and/or semicolons. The function can be preceded by <blank>s. These optional characters shall have no effect.
32667		In default operation, <i>sed</i> cyclically shall append a line of input, less its terminating <newline>, into the pattern space. Normally the pattern space will be empty, unless a D command terminated the last cycle. The <i>sed</i> utility shall then apply in sequence all commands whose addresses select that pattern space, and at the end of the script copy the pattern space to standard output (except when -n is specified) and delete the pattern space. Whenever the pattern space is written to standard output or a named file, <i>sed</i> shall immediately follow it with a <newline>.
32674		Some of the editing commands use a hold space to save all or part of the pattern space for subsequent retrieval. The pattern and hold spaces shall each be able to hold at least 8 192 bytes.
32675		

32676 **Addresses in sed**

32677 An address is either a decimal number that counts input lines cumulatively across files, a '\$' character that addresses the last line of input, or a context address (which consists of a BRE, as described in **Regular Expressions in sed**, preceded and followed by a delimiter, usually a slash).

32680 An editing command with no addresses shall select every pattern space.

32681 An editing command with one address shall select each pattern space that matches the address.

32682 An editing command with two addresses shall select the inclusive range from the first pattern space that matches the first address through the next pattern space that matches the second. (If the second address is a number less than or equal to the line number first selected, only one line shall be selected.) Starting at the first line following the selected range, *sed* shall look again for the first address. Thereafter, the process shall be repeated. Omitting either or both of the address components in the following form produces undefined results:

32688 [address[, address]]

32689 **Regular Expressions in sed**

32690 The *sed* utility shall support the BREs described in the Base Definitions volume of
32691 IEEE Std 1003.1-2001, Section 9.3, Basic Regular Expressions, with the following additions:

- In a context address, the construction "\cBREc", where c is any character other than backslash or <newline>, shall be identical to "/BRE/". If the character designated by c appears following a backslash, then it shall be considered to be that literal character, which shall not terminate the BRE. For example, in the context address "\xabc\xdefx", the second x stands for itself, so that the BRE is "abcxdef".
- The escape sequence '\n' shall match a <newline> embedded in the pattern space. A literal <newline> shall not be used in the BRE of a context address or in the substitute function.
- If an RE is empty (that is, no pattern is specified) *sed* shall behave as if the last RE used in the last command applied (either as an address or as part of a substitute command) was specified.

32702 **Editing Commands in sed**

32703 In the following list of editing commands, the maximum number of permissible addresses for
32704 each function is indicated by [0addr], [1addr], or [2addr], representing zero, one, or two
32705 addresses.

32706 The argument *text* shall consist of one or more lines. Each embedded <newline> in the text shall
32707 be preceded by a backslash. Other backslashes in text shall be removed, and the following
32708 character shall be treated literally.

32709 The r and w command verbs, and the w flag to the s command, take an optional *rfile* (or *wfile*)
32710 parameter, separated from the command verb letter or flag by one or more <blank>s;
32711 implementations may allow zero separation as an extension.

32712 The argument *rfile* or the argument *wfile* shall terminate the editing command. Each *wfile* shall be
32713 created before processing begins. Implementations shall support at least ten *wfile* arguments in
32714 the script; the actual number (greater than or equal to 10) that is supported by the
32715 implementation is unspecified. The use of the *wfile* parameter shall cause that file to be initially
32716 created, if it does not exist, or shall replace the contents of an existing file.

32717 The b, r, s, t, w, y, and : command verbs shall accept additional arguments. The following
32718 synopses indicate which arguments shall be separated from the command verbs by a single

32719 <space>.

32720 The **a** and **r** commands schedule text for later output. The text specified for the **a** command, and the contents of the file specified for the **r** command, shall be written to standard output just before the next attempt to fetch a line of input when executing the **N** or **n** commands, or when reaching the end of the script. If written when reaching the end of the script, and the **-n** option was not specified, the text shall be written after copying the pattern space to standard output. The contents of the file specified for the **r** command shall be as of the time the output is written, not the time the **r** command is applied. The text shall be output in the order in which the **a** and **r** commands were applied to the input.

32728 Command verbs other than **{**, **a**, **b**, **c**, **i**, **r**, **t**, **w**, **:**, and **#** can be followed by a semicolon, optional **<blank>**s, and another command verb. However, when the **s** command verb is used with the **w** flag, following it with another command in this manner produces undefined results.

32731 A function can be preceded by one or more '!' characters, in which case the function shall be applied if the addresses do not select the pattern space. Zero or more **<blank>**s shall be accepted before the first '!' character. It is unspecified whether **<blank>**s can follow a '!' character, and conforming applications shall not follow a '!' character with **<blank>**s.

32735 **[2addr] {function**
 32736 **function**
 32737 ...
 32738 **}** Execute a list of *sed* functions only when the pattern space is selected. The list of *sed* functions shall be surrounded by braces and separated by **<newline>**s, and conform to the following rules. The braces can be preceded or followed by **<blank>**s. The functions can be preceded by **<blank>**s, but shall not be followed by **<blank>**s. The **<right-brace>** shall be preceded by a **<newline>** and can be preceded or followed by **<blank>**s.

32744 **[1addr]a**
 32745 **text** Write text to standard output as described previously.

32746 **[2addr]b [label]**
 32747 Branch to the : function bearing the *label*. If *label* is not specified, branch to the end of the script. The implementation shall support *labels* recognized as unique up to at least 8 characters; the actual length (greater than or equal to 8) that shall be supported by the implementation is unspecified. It is unspecified whether exceeding a label length causes an error or a silent truncation.

32752 **[2addr]c**
 32753 **text** Delete the pattern space. With a 0 or 1 address or at the end of a 2-address range, place *text* on the output and start the next cycle.

32755 **[2addr]d** Delete the pattern space and start the next cycle.

32756 **[2addr]D** Delete the initial segment of the pattern space through the first **<newline>** and start the next cycle.

32758 **[2addr]g** Replace the contents of the pattern space by the contents of the hold space.

32759 **[2addr]G** Append to the pattern space a **<newline>** followed by the contents of the hold space.

32761 **[2addr]h** Replace the contents of the hold space with the contents of the pattern space.

32762 **[2addr]H** Append to the hold space a **<newline>** followed by the contents of the pattern space.

32764	[1addr]i\text	Write <i>text</i> to standard output.
32766	[2addr]l	(The letter ell.) Write the pattern space to standard output in a visually unambiguous form. The characters listed in the Base Definitions volume of IEEE Std 1003.1-2001, Table 5-1, Escape Sequences and Associated Actions ('\\', '\\a', '\\b', '\\f', '\\r', '\\t', '\\v') shall be written as the corresponding escape sequence; the '\\n' in that table is not applicable. Non-printable characters not in that table shall be written as one three-digit octal number (with a preceding backslash) for each byte in the character (most significant byte first). 2
32773		Long lines shall be folded, with the point of folding indicated by writing a backslash followed by a <newline>; the length at which folding occurs is unspecified, but should be appropriate for the output device. The end of each line shall be marked with a '\$'.
32777	[2addr]n	Write the pattern space to standard output if the default output has not been suppressed, and replace the pattern space with the next line of input, less its terminating <newline>.
32780		If no next line of input is available, the n command verb shall branch to the end of the script and quit without starting a new cycle.
32782	[2addr]N	Append the next line of input, less its terminating <newline>, to the pattern space, using an embedded <newline> to separate the appended material from the original material. Note that the current line number changes.
32785		If no next line of input is available, the N command verb shall branch to the end of the script and quit without starting a new cycle or copying the pattern space to standard output.
32788	[2addr]p	Write the pattern space to standard output.
32789	[2addr]P	Write the pattern space, up to the first <newline>, to standard output.
32790	[1addr]q	Branch to the end of the script and quit without starting a new cycle.
32791	[1addr]r rfile	Copy the contents of <i>rfile</i> to standard output as described previously. If <i>rfile</i> does not exist or cannot be read, it shall be treated as if it were an empty file, causing no error condition.
32794	[2addr]s/BRE/replacement/flags	Substitute the replacement string for instances of the BRE in the pattern space. Any character other than backslash or <newline> can be used instead of a slash to delimit the BRE and the replacement. Within the BRE and the replacement, the BRE delimiter itself can be used as a literal character if it is preceded by a backslash.
32800		The replacement string shall be scanned from beginning to end. An ampersand ('&') appearing in the replacement shall be replaced by the string matching the BRE. The special meaning of '&' in this context can be suppressed by preceding it by a backslash. The characters "\n", where <i>n</i> is a digit, shall be replaced by the text matched by the corresponding backreference expression. The special meaning of "\n" where <i>n</i> is a digit in this context, can be suppressed by preceding it by a backslash. For each other backslash ('\') encountered, the following character shall lose its special meaning (if any). The meaning of a '\' immediately followed by any character other than '&', '\', a digit, or the delimiter character used for this command, is unspecified.

32810 A line can be split by substituting a <newline> into it. The application shall escape
 32811 the <newline> in the replacement by preceding it by a backslash. A substitution
 32812 shall be considered to have been performed even if the replacement string is
 32813 identical to the string that it replaces. Any backslash used to alter the default
 32814 meaning of a subsequent character shall be discarded from the BRE or the
 32815 replacement before evaluating the BRE or using the replacement.

32816 The value of *flags* shall be zero or more of:

- 32817 *n* Substitute for the *n*th occurrence only of the BRE found within the
 pattern space.
- 32819 *g* Globally substitute for all non-overlapping instances of the BRE
 rather than just the first one. If both **g** and *n* are specified, the results
 are unspecified.
- 32822 *p* Write the pattern space to standard output if a replacement was
 made.
- 32824 *w wfile* Write. Append the pattern space to *wfile* if a replacement was made.
 A conforming application shall precede the *wfile* argument with one
 or more <blank>s. If the **w** flag is not the last flag value given in a
 concatenation of multiple flag values, the results are undefined.

32828 **[2addr]t [label]**

32829 Test. Branch to the : command verb bearing the *label* if any substitutions have been
 32830 made since the most recent reading of an input line or execution of a t. If *label* is
 32831 not specified, branch to the end of the script.

32832 **[2addr]w wfile**

32833 Append (write) the pattern space to *wfile*.

32834 **[2addr]x** Exchange the contents of the pattern and hold spaces.

32835 **[2addr]y/string1/string2/**

32836 Replace all occurrences of characters in *string1* with the corresponding characters
 32837 in *string2*. If a backslash followed by an 'n' appear in *string1* or *string2*, the two
 32838 characters shall be handled as a single <newline>. If the number of characters in
 32839 *string1* and *string2* are not equal, or if any of the characters in *string1* appear more
 32840 than once, the results are undefined. Any character other than backslash or
 32841 <newline> can be used instead of slash to delimit the strings. If the delimiter is not
 32842 'n', within *string1* and *string2*, the delimiter itself can be used as a literal character
 32843 if it is preceded by a backslash. If a backslash character is immediately followed by
 32844 a backslash character in *string1* or *string2*, the two backslash characters shall be
 32845 counted as a single literal backslash character. The meaning of a backslash
 32846 followed by any character that is not 'n', a backslash, or the delimiter character is
 32847 undefined.

32848 **[0addr]:label** Do nothing. This command bears a *label* to which the **b** and **t** commands branch.

32849 **[1addr]=** Write the following to standard output:

32850 "%d\n", <current line number>

32851 **[0addr]** Ignore this empty command.

32852 **[0addr]#** Ignore the '#' and the remainder of the line (treat them as a comment), with the
 32853 single exception that if the first two characters in the script are "#n", the default
 32854 output shall be suppressed; this shall be the equivalent of specifying -n on the

2

32855 command line.

32856 EXIT STATUS

32857 The following exit values shall be returned:

32858 0 Successful completion.

32859 >0 An error occurred.

32860 CONSEQUENCES OF ERRORS

32861 Default.

32862 APPLICATION USAGE

32863 Regular expressions match entire strings, not just individual lines, but a <newline> is matched
32864 by '\n' in a sed RE; a <newline> is not allowed by the general definition of regular expression in
32865 IEEE Std 1003.1-2001. Also note that '\n' cannot be used to match a <newline> at the end of an
32866 arbitrary input line; <newline>s appear in the pattern space as a result of the N editing
32867 command.

32868 EXAMPLES

32869 This sed script simulates the BSD cat -s command, squeezing excess blank lines from standard
32870 input.

```
32871     sed -n '
32872         # Write non-empty lines.
32873         ./ {
32874             p
32875             d
32876         }
32877         # Write a single empty line, then look for more empty lines.
32878         /^$/      p
32879         # Get next line, discard the held <newline> (empty line),
32880         # and look for more empty lines.
32881         :Empty
32882         /^$/      {
32883             N
32884             s/.//'
32885             b Empty
32886         }
32887         # Write the non-empty line before going back to search
32888         # for the first in a set of empty lines.
32889         p
32890     '
```

32891 RATIONALE

32892 This volume of IEEE Std 1003.1-2001 requires implementations to support at least ten distinct
32893 *wfiles*, matching historical practice on many implementations. Implementations are encouraged
32894 to support more, but conforming applications should not exceed this limit.

32895 The exit status codes specified here are different from those in System V. System V returns 2 for
32896 garbled sed commands, but returns zero with its usage message or if the input file could not be
32897 opened. The standard developers considered this to be a bug.

32898 The manner in which the l command writes non-printable characters was changed to avoid the
32899 historical backspace-overstrike method, and other requirements to achieve unambiguous output
32900 were added. See the RATIONALE for ed for details of the format chosen, which is the same as
32901 that chosen for sed.

32902 This volume of IEEE Std 1003.1-2001 requires implementations to provide pattern and hold
32903 spaces of at least 8192 bytes, larger than the 4000 bytes spaces used by some historical
32904 implementations, but less than the 20480 bytes limit used in an early proposal. Implementations
32905 are encouraged to allocate dynamically larger pattern and hold spaces as needed.

32906 The requirements for acceptance of <blank>s and <space>s in command lines has been made
32907 more explicit than in early proposals to describe clearly the historical practice and to remove
32908 confusion about the phrase “protect initial blanks [sic] and tabs from the stripping that is done
32909 on every script line” that appears in much of the historical documentation of the *sed* utility
32910 description of text. (Not all implementations are known to have stripped <blank>s from text
32911 lines, although they all have allowed leading <blank>s preceding the address on a command
32912 line.)

32913 The treatment of ‘#’ comments differs from the SVID which only allows a comment as the first
32914 line of the script, but matches BSD-derived implementations. The comment character is treated
32915 as a command, and it has the same properties in terms of being accepted with leading <blank>s;
32916 the BSD implementation has historically supported this.

32917 Early proposals required that a *script_file* have at least one non-comment line. Some historical
32918 implementations have behaved in unexpected ways if this were not the case. The standard
32919 developers considered that this was incorrect behavior and that application developers should
32920 not have to avoid this feature. A correct implementation of this volume of IEEE Std 1003.1-2001
32921 shall permit *script_files* that consist only of comment lines.

32922 Early proposals indicated that if **-e** and **-f** options were intermixed, all **-e** options were
32923 processed before any **-f** options. This has been changed to process them in the order presented
32924 because it matches historical practice and is more intuitive.

32925 The treatment of the **p** flag to the **s** command differs between System V and BSD-based systems
32926 when the default output is suppressed. In the two examples:

```
32927 echo a | sed 's/a/A/p'  
32928 echo a | sed -n 's/a/A/p'
```

32929 this volume of IEEE Std 1003.1-2001, BSD, System V documentation, and the SVID indicate that
32930 the first example should write two lines with **A**, whereas the second should write one. Some
32931 System V systems write the **A** only once in both examples because the **p** flag is ignored if the **-n**
32932 option is not specified.

32933 This is a case of a diametrical difference between systems that could not be reconciled through
32934 the compromise of declaring the behavior to be unspecified. The SVID/BSD/System V
32935 documentation behavior was adopted for this volume of IEEE Std 1003.1-2001 because:

- 32936 • No known documentation for any historic system describes the interaction between the **p**
32937 flag and the **-n** option.
- 32938 • The selected behavior is more correct as there is no technical justification for any interaction
32939 between the **p** flag and the **-n** option. A relationship between **-n** and the **p** flag might imply
32940 that they are only used together, but this ignores valid scripts that interrupt the cyclical
32941 nature of the processing through the use of the **D**, **d**, **q**, or branching commands. Such scripts
32942 rely on the **p** suffix to write the pattern space because they do not make use of the default
32943 output at the “bottom” of the script.
- 32944 • Because the **-n** option makes the **p** flag unnecessary, any interaction would only be useful if
32945 *sed* scripts were written to run both with and without the **-n** option. This is believed to be
32946 unlikely. It is even more unlikely that programmers have coded the **p** flag expecting it to be
32947 unnecessary. Because the interaction was not documented, the likelihood of a programmer
32948 discovering the interaction and depending on it is further decreased.

- 32949 • Finally, scripts that break under the specified behavior produce too much output instead of
32950 too little, which is easier to diagnose and correct.
- 32951 The form of the substitute command that uses the **n** suffix was limited to the first 512 matches in
32952 an early proposal. This limit has been removed because there is no reason an editor processing
32953 lines of {LINE_MAX} length should have this restriction. The command **s/a/A/2047** should be
32954 able to substitute the 2 047th occurrence of **a** on a line.
- 32955 The **b**, **t**, and **:** commands are documented to ignore leading white space, but no mention is
32956 made of trailing white space. Historical implementations of **sed** assigned different locations to
32957 the labels '**x**' and "**x**". This is not useful, and leads to subtle programming errors, but it is
32958 historical practice, and changing it could theoretically break working scripts. Implementors are
32959 encouraged to provide warning messages about labels that are never used or jumps to labels
32960 that do not exist.
- 32961 Historically, the **sed !** and **}** editing commands did not permit multiple commands on a single
32962 line using a semicolon as a command delimiter. Implementations are permitted, but not
32963 required, to support this extension.
- 32964 Previous versions of this standard allowed for implementations with bytes other than eight bits, 2
32965 but this has been modified in this version. 2
- 32966 **FUTURE DIRECTIONS**
- 32967 None.
- 32968 **SEE ALSO**
- 32969 **awk**, **ed**, **grep**
- 32970 **CHANGE HISTORY**
- 32971 First released in Issue 2.
- 32972 **Issue 5**
- 32973 The FUTURE DIRECTIONS section is added.
- 32974 **Issue 6**
- 32975 The following new requirements on POSIX implementations derive from alignment with the
32976 Single UNIX Specification:
- 32977 • Implementations are required to support at least ten *wfile* arguments in an editing command.
- 32978 The EXTENDED DESCRIPTION is changed to align with the IEEE P1003.2b draft standard.
- 32979 IEEE PASC Interpretation 1003.2 #190 is applied.
- 32980 IEEE PASC Interpretation 1003.2 #203 is applied, clarifying the meaning of the backslash escape
32981 sequences in a replacement string for a BRE.
- 32982 IEEE Std 1003.1-2001/Cor 2-2004, item XCU/TC2/D6/28 is applied, removing text describing 2
32983 behavior on systems with bytes consisting of more than eight bits. 2
- 32984 IEEE Std 1003.1-2001/Cor 2-2004, item XCU/TC2/D6/29 is applied, making an editorial 2
32985 correction within the Editing Commands in **sed** section. 2

32986 NAME

32987 sh — shell, the standard command language interpreter

32988 SYNOPSIS

```
32989     sh [-abCefhimnuvx][ -o option ][ +abCefhimnuvx ][ +o option ]
32990             [ command_file [ argument... ] ]
32991     sh -c[-abCefhimnuvx][ -o option ][ +abCefhimnuvx ][ +o option ]command_string
32992             [ command_name [ argument... ] ]
32993     sh -s[-abCefhimnuvx][ -o option ][ +abCefhimnuvx ][ +o option ][ argument ]
```

32994 DESCRIPTION

32995 The *sh* utility is a command language interpreter that shall execute commands read from a
32996 command line string, the standard input, or a specified file. The application shall ensure that the
32997 commands to be executed are expressed in the language described in Chapter 2 (on page 29).

32998 Pathname expansion shall not fail due to the size of a file.

32999 Shell input and output redirections have an implementation-defined offset maximum that is
33000 established in the open file description.

33001 OPTIONS

33002 The *sh* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section 12.2,
33003 Utility Syntax Guidelines, with an extension for support of a leading plus sign ('+') as noted
33004 below.

33005 The **-a**, **-b**, **-C**, **-e**, **-f**, **-m**, **-n**, **-o option**, **-u**, **-v**, and **-x** options are described as part of the *set*
33006 utility in Section 2.14 (on page 64). The option letters derived from the *set* special built-in shall
33007 also be accepted with a leading plus sign ('+') instead of a leading hyphen (meaning the reverse
33008 case of the option as described in this volume of IEEE Std 1003.1-2001).

33009 The following additional options shall be supported:

33010 **-c** Read commands from the *command_string* operand. Set the value of special
33011 parameter 0 (see Section 2.5.2 (on page 34)) from the value of the *command_name*
33012 operand and the positional parameters (\$1, \$2, and so on) in sequence from the
33013 remaining *argument* operands. No commands shall be read from the standard
33014 input.

33015 **-i** Specify that the shell is *interactive*; see below. An implementation may treat
33016 specifying the **-i** option as an error if the real user ID of the calling process does
33017 not equal the effective user ID or if the real group ID does not equal the effective
33018 group ID.

33019 **-s** Read commands from the standard input.

33020 If there are no operands and the **-c** option is not specified, the **-s** option shall be assumed.

33021 If the **-i** option is present, or if there are no operands and the shell's standard input and standard
33022 error are attached to a terminal, the shell is considered to be *interactive*.

33023 OPERANDS

33024 The following operands shall be supported:

33025 **-** A single hyphen shall be treated as the first operand and then ignored. If both '**-**'
33026 and "**--**" are given as arguments, or if other operands precede the single hyphen,
33027 the results are undefined.

33028 **argument** The positional parameters (\$1, \$2, and so on) shall be set to *arguments*, if any.

33029 *command_file* The pathname of a file containing commands. If the pathname contains one or
33030 more slash characters, the implementation attempts to read that file; the file need
33031 not be executable. If the pathname does not contain a slash character:

- 33032 • The implementation shall attempt to read that file from the current working
33033 directory; the file need not be executable.
- 33034 • If the file is not in the current working directory, the implementation may
33035 perform a search for an executable file using the value of *PATH*, as described in
33036 Section 2.9.1.1 (on page 48).

33037 Special parameter 0 (see Section 2.5.2 (on page 34)) shall be set to the value of
33038 *command_file*. If *sh* is called using a synopsis form that omits *command_file*, special
33039 parameter 0 shall be set to the value of the first argument passed to *sh* from its
33040 parent (for example, *argv[0]* for a C program), which is normally a pathname used
33041 to execute the *sh* utility.

33042 *command_name*
33043 A string assigned to special parameter 0 when executing the commands in
33044 *command_string*. If *command_name* is not specified, special parameter 0 shall be set
33045 to the value of the first argument passed to *sh* from its parent (for example, *argv[0]*
33046 for a C program), which is normally a pathname used to execute the *sh* utility.

33047 *command_string*
33048 A string that shall be interpreted by the shell as one or more commands, as if the
33049 string were the argument to the *system()* function defined in the System Interfaces
33050 volume of IEEE Std 1003.1-2001. If the *command_string* operand is an empty string,
33051 *sh* shall exit with a zero exit status.

33052 STDIN

33053 The standard input shall be used only if one of the following is true:

- 33054 • The *-s* option is specified.
- 33055 • The *-c* option is not specified and no operands are specified.
- 33056 • The script executes one or more commands that require input from standard input (such as a
33057 *read* command that does not redirect its input).

33058 See the INPUT FILES section.

33059 When the shell is using standard input and it invokes a command that also uses standard input,
33060 the shell shall ensure that the standard input file pointer points directly after the command it has
33061 read when the command begins execution. It shall not read ahead in such a manner that any
33062 characters intended to be read by the invoked command are consumed by the shell (whether
33063 interpreted by the shell or not) or that characters that are not read by the invoked command are
33064 not seen by the shell. When the command expecting to read standard input is started
33065 asynchronously by an interactive shell, it is unspecified whether characters are read by the
33066 command or interpreted by the shell.

33067 If the standard input to *sh* is a FIFO or terminal device and is set to non-blocking reads, then *sh*
33068 shall enable blocking reads on standard input. This shall remain in effect when the command
33069 completes.

33070 INPUT FILES

33071 The input file shall be a text file, except that line lengths shall be unlimited. If the input file is
33072 empty or consists solely of blank lines or comments, or both, *sh* shall exit with a zero exit status.

33073 ENVIRONMENT VARIABLES

33074 The following environment variables shall affect the execution of *sh*:

33075	<i>ENV</i>	This variable, when and only when an interactive shell is invoked, shall be subjected to parameter expansion (see Section 2.6.2 (on page 37)) by the shell, and the resulting value shall be used as a pathname of a file containing shell commands to execute in the current environment. The file need not be executable. If the expanded value of <i>ENV</i> is not an absolute pathname, the results are unspecified. <i>ENV</i> shall be ignored if the real and effective user IDs or real and effective group IDs of the process are different.
33082	<i>FCEDIT</i>	This variable, when expanded by the shell, shall determine the default value for the <i>-e editor</i> option's <i>editor</i> option-argument. If <i>FCEDIT</i> is null or unset, <i>ed</i> shall be used as the editor. This volume of IEEE Std 1003.1-2001 specifies the effects of this variable only for systems supporting the User Portability Utilities option.
33086	<i>HISTFILE</i>	Determine a pathname naming a command history file. If the <i>HISTFILE</i> variable is not set, the shell may attempt to access or create a file <i>.sh_history</i> in the directory referred to by the <i>HOME</i> environment variable. If the shell cannot obtain both read and write access to, or create, the history file, it shall use an unspecified mechanism that allows the history to operate properly. (References to history "file" in this section shall be understood to mean this unspecified mechanism in such cases.) An implementation may choose to access this variable only when initializing the history file; this initialization shall occur when <i>fc</i> or <i>sh</i> first attempt to retrieve entries from, or add entries to, the file, as the result of commands issued by the user, the file named by the <i>ENV</i> variable, or implementation-defined system start-up files. Implementations may choose to disable the history list mechanism for users with appropriate privileges who do not set <i>HISTFILE</i> ; the specific circumstances under which this occurs are implementation-defined. If more than one instance of the shell is using the same history file, it is unspecified how updates to the history file from those shells interact. As entries are deleted from the history file, they shall be deleted oldest first. It is unspecified when history file entries are physically removed from the history file. This volume of IEEE Std 1003.1-2001 specifies the effects of this variable only for systems supporting the User Portability Utilities option.
33105	<i>HISTSIZE</i>	Determine a decimal number representing the limit to the number of previous commands that are accessible. If this variable is unset, an unspecified default greater than or equal to 128 shall be used. The maximum number of commands in the history list is unspecified, but shall be at least 128. An implementation may choose to access this variable only when initializing the history file, as described under <i>HISTFILE</i> . Therefore, it is unspecified whether changes made to <i>HISTSIZE</i> after the history file has been initialized are effective.
33112	<i>HOME</i>	Determine the pathname of the user's home directory. The contents of <i>HOME</i> are used in tilde expansion as described in Section 2.6.1 (on page 37). This volume of IEEE Std 1003.1-2001 specifies the effects of this variable only for systems supporting the User Portability Utilities option.
33116	<i>IFS</i>	(Input Field Separators.) A string treated as a list of characters that shall be used for field splitting and to split lines into words with the <i>read</i> command. See Section 2.6.5 (on page 42). If <i>IFS</i> is not set, the shell shall behave as if the value of <i>IFS</i> were <i><space></i> , <i><tab></i> , and <i><newline></i> . Implementations may ignore the value of <i>IFS</i> in the environment at the time <i>sh</i> is invoked, treating <i>IFS</i> as if it were not set.

33121	<i>LANG</i>	Provide a default value for the internationalization variables that are unset or null. (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2, Internationalization Variables for the precedence of internationalization variables used to determine the values of locale categories.)
33122		
33123		
33124		
33125	<i>LC_ALL</i>	If set to a non-empty string value, override the values of all the other internationalization variables.
33126		
33127	<i>LC_COLLATE</i>	Determine the behavior of range expressions, equivalence classes, and multi-character collating elements within pattern matching.
33128		
33129		
33130	<i>LC_CTYPE</i>	Determine the locale for the interpretation of sequences of bytes of text data as characters (for example, single-byte as opposed to multi-byte characters in arguments and input files), which characters are defined as letters (character class alpha), and the behavior of character classes within pattern matching.
33131		
33132		
33133		
33134	<i>LC_MESSAGES</i>	Determine the locale that should be used to affect the format and contents of diagnostic messages written to standard error.
33135		
33136		
33137	<i>MAIL</i>	Determine a pathname of the user's mailbox file for purposes of incoming mail notification. If this variable is set, the shell shall inform the user if the file named by the variable is created or if its modification time has changed. Informing the user shall be accomplished by writing a string of unspecified format to standard error prior to the writing of the next primary prompt string. Such check shall be performed only after the completion of the interval defined by the <i>MAILCHECK</i> variable after the last such check. The user shall be informed only if <i>MAIL</i> is set and <i>MAILPATH</i> is not set. This volume of IEEE Std 1003.1-2001 specifies the effects of this variable only for systems supporting the User Portability Utilities option.
33138		
33139		
33140		
33141		
33142		
33143		
33144		
33145		
33146	<i>MAILCHECK</i>	Establish a decimal integer value that specifies how often (in seconds) the shell shall check for the arrival of mail in the files specified by the <i>MAILPATH</i> or <i>MAIL</i> variables. The default value shall be 600 seconds. If set to zero, the shell shall check before issuing each primary prompt. This volume of IEEE Std 1003.1-2001 specifies the effects of this variable only for systems supporting the User Portability Utilities option.
33147		
33148		
33149		
33150		
33151		
33152		
33153	<i>MAILPATH</i>	Provide a list of pathnames and optional messages separated by colons. If this variable is set, the shell shall inform the user if any of the files named by the variable are created or if any of their modification times change. (See the preceding entry for <i>MAIL</i> for descriptions of mail arrival and user informing.) Each pathname can be followed by '%' and a string that shall be subjected to parameter expansion and written to standard error when the modification time changes. If a '%' character in the pathname is preceded by a backslash, it shall be treated as a literal '%' in the pathname. The default message is unspecified.
33154		
33155		
33156		
33157		
33158		
33159		
33160		
33161		The <i>MAILPATH</i> environment variable takes precedence over the <i>MAIL</i> variable. This volume of IEEE Std 1003.1-2001 specifies the effects of this variable only for systems supporting the User Portability Utilities option.
33162		
33163		
33164	<i>xsi</i>	Determine the location of message catalogs for the processing of <i>LC_MESSAGES</i> .
33165	<i>PATH</i>	Establish a string formatted as described in the Base Definitions volume of IEEE Std 1003.1-2001, Chapter 8, Environment Variables, used to effect command interpretation; see Section 2.9.1.1 (on page 48).
33166		
33167		

33168 **PWD** This variable shall represent an absolute pathname of the current working
33169 directory. Assignments to this variable may be ignored unless the value is an
33170 absolute pathname of the current working directory and there are no filename
33171 components of dot or dot-dot.

33172 ASYNCHRONOUS EVENTS

33173 Default.

33174 STDOUT

33175 See the STDERR section.

33176 STDERR

33177 Except as otherwise stated (by the descriptions of any invoked utilities or in interactive mode),
33178 standard error shall be used only for diagnostic messages.

33179 OUTPUT FILES

33180 None.

33181 EXTENDED DESCRIPTION

33182 See Chapter 2. The following additional capabilities are supported on systems supporting the
33183 User Portability Utilities option.

33184 Command History List

33185 When the *sh* utility is being used interactively, it shall maintain a list of commands previously
33186 entered from the terminal in the file named by the *HISTFILE* environment variable. The type,
33187 size, and internal format of this file are unspecified. Multiple *sh* processes can share access to the
33188 file for a user, if file access permissions allow this; see the description of the *HISTFILE*
33189 environment variable.

33190 Command Line Editing

33191 When *sh* is being used interactively from a terminal, the current command and the command
33192 history (see *fc*) can be edited using vi-mode command line editing. This mode uses commands,
33193 described below, similar to a subset of those described in the *vi* utility. Implementations may
33194 offer other command line editing modes corresponding to other editing utilities.

33195 The command *set -o vi* shall enable vi-mode editing and place *sh* into *vi* insert mode (see
33196 **Command Line Editing (vi-mode)**). This command also shall disable any other editing mode
33197 that the implementation may provide. The command *set +o vi* disables vi-mode editing.

33198 Certain block-mode terminals may be unable to support shell command line editing. If a
33199 terminal is unable to provide either edit mode, it need not be possible to *set -o vi* when using the
33200 shell on this terminal.

33201 In the following sections, the characters *erase*, *interrupt*, *kill*, and *end-of-file* are those set by the
33202 *stty* utility.

33203 Command Line Editing (vi-mode)

33204 In *vi* editing mode, there shall be a distinguished line, the edit line. All the editing operations
33205 which modify a line affect the edit line. The edit line is always the newest line in the command
33206 history buffer.

33207 With vi-mode enabled, *sh* can be switched between insert mode and command mode.

33208 When in insert mode, an entered character shall be inserted into the command line, except as
33209 noted in **vi Line Editing Insert Mode** (on page 856). Upon entering *sh* and after termination of
33210 the previous command, *sh* shall be in insert mode.

33211 Typing an escape character shall switch *sh* into command mode (see **vi Line Editing Command**
33212 **Mode** (on page 857)). In command mode, an entered character shall either invoke a defined
33213 operation, be used as part of a multi-character operation, or be treated as an error. A character
33214 that is not recognized as part of an editing command shall terminate any specific editing
33215 command and shall alert the terminal. Typing the *interrupt* character in command mode shall
33216 cause *sh* to terminate command line editing on the current command line, reissue the prompt on
33217 the next line of the terminal, and reset the command history (see *fc*) so that the most recently
33218 executed command is the previous command (that is, the command that was being edited when
33219 it was interrupted is not reentered into the history).

33220 In the following sections, the phrase “move the cursor to the beginning of the word” shall mean
33221 “move the cursor to the first character of the current word” and the phrase “move the cursor to
33222 the end of the word” shall mean “move the cursor to the last character of the current word”. The
33223 phrase “beginning of the command line” indicates the point between the end of the prompt
33224 string issued by the shell (or the beginning of the terminal line, if there is no prompt string) and
33225 the first character of the command text.

33226 **vi Line Editing Insert Mode**

33227 While in insert mode, any character typed shall be inserted in the current command line, unless
33228 it is from the following set.

- 33229 <newline> Execute the current command line. If the current command line is not empty, this
33230 line shall be entered into the command history (see *fc*).
- 33231 *erase* Delete the character previous to the current cursor position and move the current
33232 cursor position back one character. In insert mode, characters shall be erased from
33233 both the screen and the buffer when backspacing.
- 33234 *interrupt* Terminate command line editing with the same effects as described for
33235 interrupting command mode; see **Command Line Editing (vi-mode)** (on page
33236 855).
- 33237 *kill* Clear all the characters from the input line.
- 33238 <control>-V Insert the next character input, even if the character is otherwise a special insert
33239 mode character.
- 33240 <control>-W Delete the characters from the one preceding the cursor to the preceding word
33241 boundary. The word boundary in this case is the closer to the cursor of either the
33242 beginning of the line or a character that is in neither the **blank** nor **punct** character
33243 classification of the current locale.
- 33244 *end-of-file* Interpreted as the end of input in *sh*. This interpretation shall occur only at the
33245 beginning of an input line. If *end-of-file* is entered other than at the beginning of the
33246 line, the results are unspecified.
- 33247 <ESC> Place *sh* into command mode.

33248 **vi Line Editing Command Mode**

33249 In command mode for the command line editing feature, decimal digits not beginning with 0
33250 that precede a command letter shall be remembered. Some commands use these decimal digits
33251 as a count number that affects the operation.

33252 The term *motion command* represents one of the commands:

33253 <space> 0 b F l W ^ \$; E f T w | , B e h t

33254 If the current line is not the edit line, any command that modifies the current line shall cause the
33255 content of the current line to replace the content of the edit line, and the current line shall
33256 become the edit line. This replacement cannot be undone (see the **u** and **U** commands below).
33257 The modification requested shall then be performed to the edit line. When the current line is the
33258 edit line, the modification shall be done directly to the edit line.

33259 Any command that is preceded by *count* shall take a count (the numeric value of any preceding
33260 decimal digits). Unless otherwise noted, this count shall cause the specified operation to repeat
33261 by the number of times specified by the count. Also unless otherwise noted, a *count* that is out of
33262 range is considered an error condition and shall alert the terminal, but neither the cursor
33263 position, nor the command line, shall change.

33264 The terms *word* and *bigword* are used as defined in the *vi* description. The term *save buffer*
33265 corresponds to the term *unnamed buffer* in *vi*.

33266 The following commands shall be recognized in command mode:

33267 <newline> Execute the current command line. If the current command line is not empty, this
33268 line shall be entered into the command history (see *fc*).

33269 <control>-L Redraw the current command line. Position the cursor at the same location on the
33270 redrawn line.

33271 # Insert the character '#' at the beginning of the current command line and treat the
33272 resulting edit line as a comment. This line shall be entered into the command
33273 history; see *fc*.

33274 = Display the possible shell word expansions (see Section 2.6 (on page 36)) of the
33275 bigword at the current command line position.

33276 **Note:** This does not modify the content of the current line, and therefore does not
33277 cause the current line to become the edit line.

33278 These expansions shall be displayed on subsequent terminal lines. If the bigword
33279 contains none of the characters '?', '*', or '[', an asterisk ('*') shall be
33280 implicitly assumed at the end. If any directories are matched, these expansions
33281 shall have a '/' character appended. After the expansion, the line shall be
33282 redrawn, the cursor repositioned at the current cursor position, and *sh* shall be
33283 placed in command mode.

33284 \ Perform pathname expansion (see Section 2.6.6 (on page 42)) on the current
33285 bigword, up to the largest set of characters that can be matched uniquely. If the
33286 bigword contains none of the characters '?', '*', or '[', an asterisk ('*') shall
33287 be implicitly assumed at the end. This maximal expansion then shall replace the
33288 original bigword in the command line, and the cursor shall be placed after this
33289 expansion. If the resulting bigword completely and uniquely matches a directory, a
33290 '/' character shall be inserted directly after the bigword. If some other file is
33291 completely matched, a single <space> shall be inserted after the bigword. After
33292 this operation, *sh* shall be placed in insert mode.

33293	*	Perform pathname expansion on the current bigword and insert all expansions into the command to replace the current bigword, with each expansion separated by a single <space>. If at the end of the line, the current cursor position shall be moved to the first column position following the expansions and <i>sh</i> shall be placed in insert mode. Otherwise, the current cursor position shall be the last column position of the first character after the expansions and <i>sh</i> shall be placed in insert mode. If the current bigword contains none of the characters '?', '*', or '[', before the operation, an asterisk shall be implicitly assumed at the end.
33301	@ <i>letter</i>	Insert the value of the alias named <i>_letter</i> . The symbol <i>letter</i> represents a single alphabetic character from the portable character set; implementations may support additional characters as an extension. If the alias <i>_letter</i> contains other editing commands, these commands shall be performed as part of the insertion. If no alias <i>_letter</i> is enabled, this command shall have no effect.
33306	[<i>count</i>]~	Convert, if the current character is a lowercase letter, to the equivalent uppercase letter and <i>vice versa</i> , as prescribed by the current locale. The current cursor position then shall be advanced by one character. If the cursor was positioned on the last character of the line, the case conversion shall occur, but the cursor shall not advance. If the '~~' command is preceded by a <i>count</i> , that number of characters shall be converted, and the cursor shall be advanced to the character position after the last character converted. If the <i>count</i> is larger than the number of characters after the cursor, this shall not be considered an error; the cursor shall advance to the last character on the line.
33315	[<i>count</i>].	Repeat the most recent non-motion command, even if it was executed on an earlier command line. If the previous command was preceded by a <i>count</i> , and no count is given on the '...' command, the count from the previous command shall be included as part of the repeated command. If the '...' command is preceded by a <i>count</i> , this shall override any <i>count</i> argument to the previous command. The <i>count</i> specified in the '...' command shall become the count for subsequent '...' commands issued without a count.
33322	[<i>number</i>]v	Invoke the <i>vi</i> editor to edit the current command line in a temporary file. When the editor exits, the commands in the temporary file shall be executed and placed in the command history. If a <i>number</i> is included, it specifies the command number in the command history to be edited, rather than the current command line.
33326	[<i>count</i>]l (ell)	Move the current cursor position to the next character position. If the cursor was positioned on the last character of the line, the terminal shall be alerted and the cursor shall not be advanced. If the <i>count</i> is larger than the number of characters after the cursor, this shall not be considered an error; the cursor shall advance to the last character on the line.
33327	[<i>count</i>]<space>	
33328	[<i>count</i>]h	Move the current cursor position to the <i>count</i> th (default 1) previous character position. If the cursor was positioned on the first character of the line, the terminal shall be alerted and the cursor shall not be moved. If the <i>count</i> is larger than the number of characters before the cursor, this shall not be considered an error; the cursor shall move to the first character on the line.
33333	[<i>count</i>]w	Move to the start of the next word. If the cursor was positioned on the last character of the line, the terminal shall be alerted and the cursor shall not be advanced. If the <i>count</i> is larger than the number of words after the cursor, this shall not be considered an error; the cursor shall advance to the last character on the
33339		
33340		
33341		

33342		line.
33343	[count]W	Move to the start of the next bigword. If the cursor was positioned on the last character of the line, the terminal shall be alerted and the cursor shall not be advanced. If the <i>count</i> is larger than the number of bigwords after the cursor, this shall not be considered an error; the cursor shall advance to the last character on the line.
33344		
33345		
33346		
33347		
33348	[count]e	Move to the end of the current word. If at the end of a word, move to the end of the next word. If the cursor was positioned on the last character of the line, the terminal shall be alerted and the cursor shall not be advanced. If the <i>count</i> is larger than the number of words after the cursor, this shall not be considered an error; the cursor shall advance to the last character on the line.
33349		
33350		
33351		
33352		
33353	[count]E	Move to the end of the current bigword. If at the end of a bigword, move to the end of the next bigword. If the cursor was positioned on the last character of the line, the terminal shall be alerted and the cursor shall not be advanced. If the <i>count</i> is larger than the number of bigwords after the cursor, this shall not be considered an error; the cursor shall advance to the last character on the line.
33354		
33355		
33356		
33357		
33358	[count]b	Move to the beginning of the current word. If at the beginning of a word, move to the beginning of the previous word. If the cursor was positioned on the first character of the line, the terminal shall be alerted and the cursor shall not be moved. If the <i>count</i> is larger than the number of words preceding the cursor, this shall not be considered an error; the cursor shall return to the first character on the line.
33359		
33360		
33361		
33362		
33363		
33364	[count]B	Move to the beginning of the current bigword. If at the beginning of a bigword, move to the beginning of the previous bigword. If the cursor was positioned on the first character of the line, the terminal shall be alerted and the cursor shall not be moved. If the <i>count</i> is larger than the number of bigwords preceding the cursor, this shall not be considered an error; the cursor shall return to the first character on the line.
33365		
33366		
33367		
33368		
33369		
33370	^	Move the current cursor position to the first character on the input line that is not a <blank>.
33371		
33372	\$	Move to the last character position on the current command line.
33373	0	(Zero.) Move to the first character position on the current command line.
33374	[count]	Move to the <i>count</i> th character position on the current command line. If no number is specified, move to the first position. The first character position shall be numbered 1. If the count is larger than the number of characters on the line, this shall not be considered an error; the cursor shall be placed on the last character on the line.
33375		
33376		
33377		
33378		
33379	[count]fc	Move to the first occurrence of the character 'c' that occurs after the current cursor position. If the cursor was positioned on the last character of the line, the terminal shall be alerted and the cursor shall not be advanced. If the character 'c' does not occur in the line after the current cursor position, the terminal shall be alerted and the cursor shall not be moved.
33380		
33381		
33382		
33383		
33384	[count]Fc	Move to the first occurrence of the character 'c' that occurs before the current cursor position. If the cursor was positioned on the first character of the line, the terminal shall be alerted and the cursor shall not be moved. If the character 'c' does not occur in the line before the current cursor position, the terminal shall be alerted and the cursor shall not be moved.
33385		
33386		
33387		
33388		

33389	[count]tc	Move to the character before the first occurrence of the character 'c' that occurs after the current cursor position. If the cursor was positioned on the last character of the line, the terminal shall be alerted and the cursor shall not be advanced. If the character 'c' does not occur in the line after the current cursor position, the terminal shall be alerted and the cursor shall not be moved.
33390		
33391		
33392		
33393		
33394	[count]Tc	Move to the character after the first occurrence of the character 'c' that occurs before the current cursor position. If the cursor was positioned on the first character of the line, the terminal shall be alerted and the cursor shall not be moved. If the character 'c' does not occur in the line before the current cursor position, the terminal shall be alerted and the cursor shall not be moved.
33395		
33396		
33397		
33398		
33399	[count];	Repeat the most recent f, F, t, or T command. Any number argument on that previous command shall be ignored. Errors are those described for the repeated command.
33400		
33401		
33402	[count],	Repeat the most recent f, F, t, or T command. Any number argument on that previous command shall be ignored. However, reverse the direction of that command.
33403		
33404		
33405	a	Enter insert mode after the current cursor position. Characters that are entered shall be inserted before the next character.
33406		
33407	A	Enter insert mode after the end of the current command line.
33408	i	Enter insert mode at the current cursor position. Characters that are entered shall be inserted before the current character.
33409		
33410	I	Enter insert mode at the beginning of the current command line.
33411	R	Enter insert mode, replacing characters from the command line beginning at the current cursor position.
33412		
33413	[count]cmotion	Delete the characters between the current cursor position and the cursor position that would result from the specified motion command. Then enter insert mode before the first character following any deleted characters. If <i>count</i> is specified, it shall be applied to the motion command. A <i>count</i> shall be ignored for the following motion commands:
33414		
33415		
33416		
33417		
33418		
33419	0 ^ \$ c	
33420		If the motion command is the character 'c', the current command line shall be cleared and insert mode shall be entered. If the motion command would move the current cursor position toward the beginning of the command line, the character under the current cursor position shall not be deleted. If the motion command would move the current cursor position toward the end of the command line, the character under the current cursor position shall be deleted. If the <i>count</i> is larger than the number of characters between the current cursor position and the end of the command line toward which the motion command would move the cursor, this shall not be considered an error; all of the remaining characters in the aforementioned range shall be deleted and insert mode shall be entered. If the motion command is invalid, the terminal shall be alerted, the cursor shall not be moved, and no text shall be deleted.
33421		
33422		
33423		
33424		
33425		
33426		
33427		
33428		
33429		
33430		
33431		
33432	C	Delete from the current character to the end of the line and enter insert mode at the new end-of-line.
33433		

33434	S	Clear the entire edit line and enter insert mode.
33435	[count]rc	Replace the current character with the character ' <i>c</i> '. With a number <i>count</i> , replace the current and the following <i>count</i> -1 characters. After this command, the current cursor position shall be on the last character that was changed. If the <i>count</i> is larger than the number of characters after the cursor, this shall not be considered an error; all of the remaining characters shall be changed.
33436		
33437		
33438		
33439		
33440	[count]_	Append a <space> after the current character position and then append the last bigword in the previous input line after the <space>. Then enter insert mode after the last character just appended. With a number <i>count</i> , append the <i>count</i> th bigword in the previous line.
33441		
33442		
33443		
33444	[count]x	Delete the character at the current cursor position and place the deleted characters in the save buffer. If the cursor was positioned on the last character of the line, the character shall be deleted and the cursor position shall be moved to the previous character (the new last character). If the <i>count</i> is larger than the number of characters after the cursor, this shall not be considered an error; all the characters from the cursor to the end of the line shall be deleted.
33445		
33446		
33447		
33448		
33449		
33450	[count]X	Delete the character before the current cursor position and place the deleted characters in the save buffer. The character under the current cursor position shall not change. If the cursor was positioned on the first character of the line, the terminal shall be alerted, and the X command shall have no effect. If the line contained a single character, the X command shall have no effect. If the line contained no characters, the terminal shall be alerted and the cursor shall not be moved. If the <i>count</i> is larger than the number of characters before the cursor, this shall not be considered an error; all the characters from before the cursor to the beginning of the line shall be deleted.
33451		
33452		
33453		
33454		
33455		
33456		
33457		
33458		
33459	[count]dmotion	Delete the characters between the current cursor position and the character position that would result from the motion command. A number <i>count</i> repeats the motion command <i>count</i> times. If the motion command would move toward the beginning of the command line, the character under the current cursor position shall not be deleted. If the motion command is d, the entire current command line shall be cleared. If the <i>count</i> is larger than the number of characters between the current cursor position and the end of the command line toward which the motion command would move the cursor, this shall not be considered an error; all of the remaining characters in the aforementioned range shall be deleted. The deleted characters shall be placed in the save buffer.
33460		
33461		
33462		
33463		
33464		
33465		
33466		
33467		
33468		
33469		
33470	D	Delete all characters from the current cursor position to the end of the line. The deleted characters shall be placed in the save buffer.
33471		
33472	[count]ymotion	Yank (that is, copy) the characters from the current cursor position to the position resulting from the motion command into the save buffer. A number <i>count</i> shall be applied to the motion command. If the motion command would move toward the beginning of the command line, the character under the current cursor position shall not be included in the set of yanked characters. If the motion command is y, the entire current command line shall be yanked into the save buffer. The current cursor position shall be unchanged. If the <i>count</i> is larger than the number of characters between the current cursor position and the end of the command line toward which the motion command would move the cursor, this shall not be considered an error; all of the remaining characters in the aforementioned range
33473		
33474		
33475		
33476		
33477		
33478		
33479		
33480		
33481		
33482		

33483		shall be yanked.
33484	Y	Yank the characters from the current cursor position to the end of the line into the save buffer. The current character position shall be unchanged.
33485		
33486	[count]p	Put a copy of the current contents of the save buffer after the current cursor position. The current cursor position shall be advanced to the last character put from the save buffer. A <i>count</i> shall indicate how many copies of the save buffer shall be put.
33487		
33488		
33489		
33490	[count]P	Put a copy of the current contents of the save buffer before the current cursor position. The current cursor position shall be moved to the last character put from the save buffer. A <i>count</i> shall indicate how many copies of the save buffer shall be put.
33491		
33492		
33493		
33494	u	Undo the last command that changed the edit line. This operation shall not undo the copy of any command line to the edit line.
33495		
33496	U	Undo all changes made to the edit line. This operation shall not undo the copy of any command line to the edit line.
33497		
33498	[count]k	
33499	[count]–	Set the current command line to be the <i>count</i> th previous command line in the shell command history. If <i>count</i> is not specified, it shall default to 1. The cursor shall be positioned on the first character of the new command. If a k or – command would retreat past the maximum number of commands in effect for this shell (affected by the <i>HISTSIZE</i> environment variable), the terminal shall be alerted, and the command shall have no effect.
33500		
33501		
33502		
33503		
33504		
33505	[count]j	
33506	[count]+	Set the current command line to be the <i>count</i> th next command line in the shell command history. If <i>count</i> is not specified, it shall default to 1. The cursor shall be positioned on the first character of the new command. If a j or + command advances past the edit line, the current command line shall be restored to the edit line and the terminal shall be alerted.
33507		
33508		
33509		
33510		
33511	[number]G	Set the current command line to be the oldest command line stored in the shell command history. With a number <i>number</i> , set the current command line to be the command line <i>number</i> in the history. If command line <i>number</i> does not exist, the terminal shall be alerted and the command line shall not be changed.
33512		
33513		
33514		
33515	/pattern<newline>	Move backwards through the command history, searching for the specified pattern, beginning with the previous command line. Patterns use the pattern matching notation described in Section 2.13 (on page 62), except that the '^' character shall have special meaning when it appears as the first character of <i>pattern</i> . In this case, the '^' is discarded and the characters after the '^' shall be matched only at the beginning of a line. Commands in the command history shall be treated as strings, not as filenames. If the pattern is not found, the current command line shall be unchanged and the terminal is alerted. If it is found in a previous line, the current command line shall be set to that line and the cursor shall be set to the first character of the new command line.
33516		
33517		
33518		
33519		
33520		
33521		
33522		
33523		
33524		
33525		
33526	If <i>pattern</i> is empty, the last non-empty pattern provided to / or ? shall be used. If there is no previous non-empty pattern, the terminal shall be alerted and the current command line shall remain unchanged.	
33527		
33528		

- 33529 **?pattern<newline>**
33530 Move forwards through the command history, searching for the specified pattern,
33531 beginning with the next command line. Patterns use the pattern matching notation
33532 described in Section 2.13 (on page 62), except that the '^' character shall have
33533 special meaning when it appears as the first character of *pattern*. In this case, the
33534 '^' is discarded and the characters after the '^' shall be matched only at the
33535 beginning of a line. Commands in the command history shall be treated as strings,
33536 not as filenames. If the pattern is not found, the current command line shall be
33537 unchanged and the terminal alerted. If it is found in a following line, the current
33538 command line shall be set to that line and the cursor shall be set to the fist
33539 character of the new command line.
- 33540 If *pattern* is empty, the last non-empty pattern provided to / or ? shall be used. If
33541 there is no previous non-empty pattern, the terminal shall be alerted and the
33542 current command line shall remain unchanged.
- 33543 **n** Repeat the most recent / or ? command. If there is no previous / or ?, the terminal
33544 shall be alerted and the current command line shall remain unchanged.
- 33545 **N** Repeat the most recent / or ? command, reversing the direction of the search. If
33546 there is no previous / or ?, the terminal shall be alerted and the current command
33547 line shall remain unchanged.

33548 EXIT STATUS

33549 The following exit values shall be returned:

- 33550 0 The script to be executed consisted solely of zero or more blank lines or comments, or
33551 both.
- 33552 1-125 A non-interactive shell detected a syntax, redirection, or variable assignment error.
- 33553 127 A specified *command_file* could not be found by a non-interactive shell.
- 33554 Otherwise, the shell shall return the exit status of the last command it invoked or attempted to
33555 invoke (see also the *exit* utility in Section 2.14 (on page 64)).

33556 CONSEQUENCES OF ERRORS

33557 See Section 2.8.1 (on page 46).

33558 APPLICATION USAGE

33559 Standard input and standard error are the files that determine whether a shell is interactive
33560 when **-i** is not specified. For example:

33561 `sh > file`

33562 and:

33563 `sh 2> file`

33564 create interactive and non-interactive shells, respectively. Although both accept terminal input,
33565 the results of error conditions are different, as described in Section 2.8.1 (on page 46); in the
33566 second example a redirection error encountered by a special built-in utility aborts the shell.

33567 A conforming application must protect its first operand, if it starts with a plus sign, by preceding
33568 it with the **--** argument that denotes the end of the options.

33569 Applications should note that the standard *PATH* to the shell cannot be assumed to be either
33570 **/bin/sh** or **/usr/bin/sh**, and should be determined by interrogation of the *PATH* returned by
33571 *getconf PATH*, ensuring that the returned pathname is an absolute pathname and not a shell
33572 built-in.

33573 For example, to determine the location of the standard *sh* utility:

33574 command -v sh

33575 On some implementations this might return:

33576 /usr/xpg4/bin/sh

33577 Furthermore, on systems that support executable scripts (the "#!" construct), it is
33578 recommended that applications using executable scripts install them using *getconf -v* to
33579 determine the shell pathname and update the "#!" script appropriately as it is being installed
33580 (for example, with *sed*). For example:

```
#  
# Installation time script to install correct POSIX shell pathname  
#  
33584 # Get list of paths to check  
#  
33586 Sifs=$IFS  
33587 IFS=:  
33588 set $(getconf PATH)  
33589 IFS=$Sifs  
#  
33591 # Check each path for 'sh'  
#  
33593 for i in $@  
do  
    if [ -f ${i}/sh ];  
    then  
        Pshell=${i}/sh  
    fi  
done  
#  
33601 # This is the list of scripts to update. They should be of the  
# form '${name}.source' and will be transformed to '${name}'.  
# Each script should begin:  
#  
33605 # !INSTALLSHELLPATH -p  
#  
33607 scripts="a b c"  
#  
33609 # Transform each script  
#  
33611 for i in ${scripts}  
do  
    sed -e "s|INSTALLSHELLPATH|${Pshell}|" < ${i}.source > ${i}  
done
```

33615 EXAMPLES

33616 1. Execute a shell command from a string:

33617 sh -c "cat myfile"

33618 2. Execute a shell script from a file in the current directory:

33619 sh my_shell_cmds

33620 **RATIONALE**

33621 The *sh* utility and the *set* special built-in utility share a common set of options.

33622 The KornShell ignores the contents of *IFS* upon entry to the script. A conforming application
33623 cannot rely on importing *IFS*. One justification for this, beyond security considerations, is to
33624 assist possible future shell compilers. Allowing *IFS* to be imported from the environment
33625 prevents many optimizations that might otherwise be performed via dataflow analysis of the
33626 script itself.

33627 The text in the STDIN section about non-blocking reads concerns an instance of *sh* that has been
33628 invoked, probably by a C-language program, with standard input that has been opened using
33629 the O_NONBLOCK flag; see *open()* in the System Interfaces volume of IEEE Std 1003.1-2001. If
33630 the shell did not reset this flag, it would immediately terminate because no input data would be
33631 available yet and that would be considered the same as end-of-file.

33632 The options associated with a *restricted shell* (command name *rsh* and the *-r* option) were
33633 excluded because the standard developers considered that the implied level of security could
33634 not be achieved and they did not want to raise false expectations.

33635 On systems that support set-user-ID scripts, a historical trapdoor has been to link a script to the
33636 name *-i*. When it is called by a sequence such as:

33637 sh -

33638 or by:

33639 #! usr/bin/sh -

33640 the historical systems have assumed that no option letters follow. Thus, this volume of
33641 IEEE Std 1003.1-2001 allows the single hyphen to mark the end of the options, in addition to the
33642 use of the regular "--" argument, because it was considered that the older practice was so
33643 pervasive. An alternative approach is taken by the KornShell, where real and effective
33644 user/group IDs must match for an interactive shell; this behavior is specifically allowed by this
33645 volume of IEEE Std 1003.1-2001.

33646 **Note:** There are other problems with set-user-ID scripts that the two approaches described here do
33647 not resolve.

33648 The initialization process for the history file can be dependent on the system start-up files, in
33649 that they may contain commands that effectively preempt the user's settings of *HISTFILE* and
33650 *HISTSIZE*. For example, function definition commands are recorded in the history file, unless
33651 the *set -o nolog* option is set. If the system administrator includes function definitions in some
33652 system start-up file called before the *ENV* file, the history file is initialized before the user gets a
33653 chance to influence its characteristics. In some historical shells, the history file is initialized just
33654 after the *ENV* file has been processed. Therefore, it is implementation-defined whether changes
33655 made to *HISTFILE* after the history file has been initialized are effective.

33656 The default messages for the various *MAIL*-related messages are unspecified because they vary
33657 across implementations. Typical messages are:

33658 "you have mail\n"

33659 or:

33660 "you have new mail\n"

33661 It is important that the descriptions of command line editing refer to the same shell as that in
33662 IEEE Std 1003.1-2001 so that interactive users can also be application programmers without
33663 having to deal with programmatic differences in their two environments. It is also essential that

33664 the utility name *sh* be specified because this explicit utility name is too firmly rooted in historical
33665 practice of application programs for it to change.

33666 Consideration was given to mandating a diagnostic message when attempting to set *vi*-mode on
33667 terminals that do not support command line editing. However, it is not historical practice for the
33668 shell to be cognizant of all terminal types and thus be able to detect inappropriate terminals in
33669 all cases. Implementations are encouraged to supply diagnostics in this case whenever possible,
33670 rather than leaving the user in a state where editing commands work incorrectly.

33671 In early proposals, the KornShell-derived *emacs* mode of command line editing was included,
33672 even though the *emacs* editor itself was not. The community of *emacs* proponents was adamant
33673 that the full *emacs* editor not be standardized because they were concerned that an attempt to
33674 standardize this very powerful environment would encourage vendors to ship strictly
33675 conforming versions lacking the extensibility required by the community. The author of the
33676 original *emacs* program also expressed his desire to omit the program. Furthermore, there were a
33677 number of historical systems that did not include *emacs*, or included it without supporting it, but
33678 there were very few that did not include and support *vi*. The shell *emacs* command line editing
33679 mode was finally omitted because it became apparent that the KornShell version and the editor
33680 being distributed with the GNU system had diverged in some respects. The author of *emacs*
33681 requested that the POSIX *emacs* mode either be deleted or have a significant number of
33682 unspecified conditions. Although the KornShell author agreed to consider changes to bring the
33683 shell into alignment, the standard developers decided to defer specification at that time. At the
33684 time, it was assumed that convergence on an acceptable definition would occur for a subsequent
33685 draft, but that has not happened, and there appears to be no impetus to do so. In any case,
33686 implementations are free to offer additional command line editing modes based on the exact
33687 models of editors their users are most comfortable with.

33688 Early proposals had the following list entry in **vi Line Editing Insert Mode** (on page 856):

33689 \ If followed by the *erase* or *kill* character, that character shall be inserted into the input line.
33690 Otherwise, the backslash itself shall be inserted into the input line.

33691 However, this is not actually a feature of *sh* command line editing insert mode, but one of some
33692 historical terminal line drivers. Some conforming implementations continue to do this when the
33693 *stty iexten* flag is set.

33694 FUTURE DIRECTIONS

33695 None.

33696 SEE ALSO

33697 Chapter 2 (on page 29), *cd*, *echo*, *exit*, *fc*, *pwd*, *read*, *set*, *stty*, *test*, *umask*, *vi*, the System Interfaces
33698 volume of IEEE Std 1003.1-2001, *dup()*, *exec*, *exit()*, *fork()*, *open()*, *pipe()*, *signal()*, *system()*,
33699 *ulimit()*, *umask()*, *wait()*

33700 CHANGE HISTORY

33701 First released in Issue 2.

33702 Issue 5

33703 The FUTURE DIRECTIONS section is added.

33704 Text is added to the DESCRIPTION for the Large File Summit proposal.

33705 Issue 6

33706 The Open Group Corrigendum U029/2 is applied, correcting the second SYNOPSIS.

33707 The Open Group Corrigendum U027/3 is applied, correcting a typographical error.

33708 The following new requirements on POSIX implementations derive from alignment with the
33709 Single UNIX Specification:

- 33710 • The option letters derived from the *set* special built-in are also accepted with a leading plus
33711 sign ('+')
33712 • Large file extensions are added:
33713 — Pathname expansion does not fail due to the size of a file.
33714 — Shell input and output redirections have an implementation-defined offset maximum
33715 that is established in the open file description.
33716 In the ENVIRONMENT VARIABLES section, the text “user’s home directory” is updated to
33717 “directory referred to by the *HOME* environment variable”.
33718 Descriptions for the *ENV* and *PWD* environment variables are included to align with the
33719 IEEE P1003.2b draft standard.
33720 The normative text is reworded to avoid use of the term “must” for application requirements.

33721 NAME

33722 sleep — suspend execution for an interval

33723 SYNOPSIS

33724 sleep *time*

33725 DESCRIPTION

33726 The *sleep* utility shall suspend execution for at least the integral number of seconds specified by
33727 the *time* operand.

33728 OPTIONS

33729 None.

33730 OPERANDS

33731 The following operand shall be supported:

33732 *time* A non-negative decimal integer specifying the number of seconds for which to
33733 suspend execution.

33734 STDIN

33735 Not used.

33736 INPUT FILES

33737 None.

33738 ENVIRONMENT VARIABLES

33739 The following environment variables shall affect the execution of *sleep*:

33740 *LANG* Provide a default value for the internationalization variables that are unset or null.
33741 (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2,
33742 Internationalization Variables for the precedence of internationalization variables
33743 used to determine the values of locale categories.)

33744 *LC_ALL* If set to a non-empty string value, override the values of all the other
33745 internationalization variables.

33746 *LC_CTYPE* Determine the locale for the interpretation of sequences of bytes of text data as
33747 characters (for example, single-byte as opposed to multi-byte characters in
33748 arguments).

33749 *LC_MESSAGES*

33750 Determine the locale that should be used to affect the format and contents of
33751 diagnostic messages written to standard error.

33752 XSI *NLSPATH* Determine the location of message catalogs for the processing of *LC_MESSAGES*.

33753 ASYNCHRONOUS EVENTS

33754 If the *sleep* utility receives a SIGALRM signal, one of the following actions shall be taken:

- 33755 1. Terminate normally with a zero exit status.
- 33756 2. Effectively ignore the signal.
- 33757 3. Provide the default behavior for signals described in the ASYNCHRONOUS EVENTS
33758 section of Section 1.11 (on page 20). This could include terminating with a non-zero exit
33759 status.

33760 The *sleep* utility shall take the standard action for all other signals.

33761 STDOUT

33762 Not used.

33763 STDERR

33764 The standard error shall be used only for diagnostic messages.

33765 OUTPUT FILES

33766 None.

33767 EXTENDED DESCRIPTION

33768 None.

33769 EXIT STATUS

33770 The following exit values shall be returned:

33771 0 The execution was successfully suspended for at least *time* seconds, or a SIGALRM signal
33772 was received. See the ASYNCHRONOUS EVENTS section.

33773 >0 An error occurred.

3.5.1 CONSEQUENCES OF ERRORS

32776 APPLICATION USAGE

33776 APPLICATION

22778 EXAMPLES

The `sleep` utility can be used to execute a command after a certain amount of time, as in:

33780 (sleep 105; command) &

33781 or to execute a command every so often, as in:

```
33782     while true
33783     do
33784         command
33785         sleep
33786     done
```

33787 RATIONALE

The exit status is allowed to be zero when *sleep* is interrupted by the SIGALRM signal because most implementations of this utility rely on the arrival of that signal to notify them that the requested finishing time has been successfully attained. Such implementations thus do not distinguish this situation from the successful completion case. Other implementations are allowed to catch the signal and go back to sleep until the requested time expires or to provide the normal signal termination procedures.

As with all other utilities that take integral operands and do not specify subranges of allowed values, *sleep* is required by this volume of IEEE Std 1003.1-2001 to deal with *time* requests of up to 2 147 483 647 seconds. This may mean that some implementations have to make multiple calls to the delay mechanism of the underlying operating system if its argument range is less than this.

33799 FUTURE DIRECTIONS

33800 None.

33801 SEE ALSO

33802 *wait*, the System Interfaces volume of IEEE Std 1003.1-2001, *alarm()*, *sleep()*

33803 **CHANGE HISTORY**

33804 First released in Issue 2.

33805 NAME

33806 sort — sort, merge, or sequence check text files

33807 SYNOPSIS

33808 sort [-m][-o *output*][-bdfinru][-t *char*][-k *keydef*]... [*file...*]

33809 sort -c [-bdfinru][-t *char*][-k *keydef*][*file*]

33810 DESCRIPTION

33811 The *sort* utility shall perform one of the following functions:

- 33812 1. Sort lines of all the named files together and write the result to the specified output.
- 33813 2. Merge lines of all the named (presorted) files together and write the result to the specified output.
- 33815 3. Check that a single input file is correctly presorted.

33816 Comparisons shall be based on one or more sort keys extracted from each line of input (or, if no
33817 sort keys are specified, the entire line up to, but not including, the terminating <newline>), and
33818 shall be performed using the collating sequence of the current locale.

33819 OPTIONS

33820 The *sort* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section
33821 12.2, Utility Syntax Guidelines, and the **-k** *keydef* option should follow the **-b**, **-d**, **-f**, **-i**, **-n**, and
33822 **-r** options.

33823 The following options shall be supported:

- 33824 **-c** Check that the single input file is ordered as specified by the arguments and the
33825 collating sequence of the current locale. No output shall be produced; only the exit
33826 code shall be affected.
- 33827 **-m** Merge only; the input file shall be assumed to be already sorted.
- 33828 **-o *output*** Specify the name of an output file to be used instead of the standard output. This
33829 file can be the same as one of the input files.
- 33830 **-u** Unique: suppress all but one in each set of lines having equal keys. If used with
33831 the **-c** option, check that there are no lines with duplicate keys, in addition to
33832 checking that the input file is sorted.

33833 The following options shall override the default ordering rules. When ordering options appear
33834 independent of any key field specifications, the requested field ordering rules shall be applied
33835 globally to all sort keys. When attached to a specific key (see **-k**), the specified ordering options
33836 shall override all global ordering options for that key.

- 33837 **-d** Specify that only <blank>s and alphanumeric characters, according to the current
33838 setting of *LC_CTYPE*, shall be significant in comparisons. The behavior is
33839 undefined for a sort key to which **-i** or **-n** also applies.
- 33840 **-f** Consider all lowercase characters that have uppercase equivalents, according to
33841 the current setting of *LC_CTYPE*, to be the uppercase equivalent for the purposes
33842 of comparison.
- 33843 **-i** Ignore all characters that are non-printable, according to the current setting of
33844 *LC_CTYPE*.
- 33845 **-n** Restrict the sort key to an initial numeric string, consisting of optional <blank>s,
33846 optional minus sign, and zero or more digits with an optional radix character and
33847 thousands separators (as defined in the current locale), which shall be sorted by

33848 arithmetic value. An empty digit string shall be treated as zero. Leading zeros and
 33849 signs on zeros shall not affect ordering.

33850 **-r** Reverse the sense of comparisons.

33851 The treatment of field separators can be altered using the options:

33852 **-b** Ignore leading **<blank>**s when determining the starting and ending positions of a
 33853 restricted sort key. If the **-b** option is specified before the first **-k** option, it shall be
 33854 applied to all **-k** options. Otherwise, the **-b** option can be attached independently
 33855 to each **-k field_start** or **field_end** option-argument (see below).

33856 **-t char** Use *char* as the field separator character; *char* shall not be considered to be part of a
 33857 field (although it can be included in a sort key). Each occurrence of *char* shall be
 33858 significant (for example, **<char><char>** delimits an empty field). If **-t** is not
 33859 specified, **<blank>**s shall be used as default field separators; each maximal non-
 33860 empty sequence of **<blank>**s that follows a non-**<blank>** shall be a field separator.

33861 Sort keys can be specified using the options:

33862 **-k keydef** The *keydef* argument is a restricted sort key field definition. The format of this
 33863 definition is:

33864 *field_start[type][,field_end[type]]*

33865 where *field_start* and *field_end* define a key field restricted to a portion of the line
 33866 (see the EXTENDED DESCRIPTION section), and *type* is a modifier from the list of
 33867 characters '**b**', '**d**', '**f**', '**i**', '**n**', '**r**'. The '**b**' modifier shall behave like the
 33868 **-b** option, but shall apply only to the *field_start* or *field_end* to which it is attached.
 33869 The other modifiers shall behave like the corresponding options, but shall apply
 33870 only to the key field to which they are attached; they shall have this effect if
 33871 specified with *field_start*, *field_end*, or both. If any modifier is attached to a
 33872 *field_start* or to a *field_end*, no option shall apply to either. Implementations shall
 33873 support at least nine occurrences of the **-k** option, which shall be significant in
 33874 command line order. If no **-k** option is specified, a default sort key of the entire
 33875 line shall be used.

33876 When there are multiple key fields, later keys shall be compared only after all
 33877 earlier keys compare equal. Except when the **-u** option is specified, lines that
 33878 otherwise compare equal shall be ordered as if none of the options **-d**, **-f**, **-i**, **-n**, or
 33879 **-k** were present (but with **-r** still in effect, if it was specified) and with all bytes in
 33880 the lines significant to the comparison. The order in which lines that still compare
 33881 equal are written is unspecified.

33882 OPERANDS

33883 The following operand shall be supported:

33884 **file** A pathname of a file to be sorted, merged, or checked. If no *file* operands are
 33885 specified, or if a *file* operand is '**-**', the standard input shall be used.

33886 STDIN

33887 The standard input shall be used only if no *file* operands are specified, or if a *file* operand is '**-**'.
 33888 See the INPUT FILES section.

33889 INPUT FILES

33890 The input files shall be text files, except that the *sort* utility shall add a **<newline>** to the end of a
 33891 file ending with an incomplete last line.

33892 ENVIRONMENT VARIABLES

33893 The following environment variables shall affect the execution of *sort*:

33894 **LANG** Provide a default value for the internationalization variables that are unset or null.
33895 (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2,
33896 Internationalization Variables for the precedence of internationalization variables
33897 used to determine the values of locale categories.)

33898 **LC_ALL** If set to a non-empty string value, override the values of all the other
33899 internationalization variables.

33900 *LC_COLLATE*

33901 Determine the locale for ordering rules.

33902 **LC_CTYPE** Determine the locale for the interpretation of sequences of bytes of text data as
33903 characters (for example, single-byte as opposed to multi-byte characters in
33904 arguments and input files) and the behavior of character classification for the **-b**,
33905 **-d**, **-f**, **-i**, and **-n** options.

33906 *LC_MESSAGES*

33907 Determine the locale that should be used to affect the format and contents of
33908 diagnostic messages written to standard error.

33909 *LC_NUMERIC*

33910 Determine the locale for the definition of the radix character and thousands
33911 separator for the **-n** option.

33912 XSI **NLSPATH** Determine the location of message catalogs for the processing of *LC_MESSAGES*.

33913 ASYNCHRONOUS EVENTS

33914 Default.

33915 STDOUT

33916 Unless the **-o** or **-c** options are in effect, the standard output shall contain the sorted input.

33917 STDERR

33918 The standard error shall be used for diagnostic messages. A warning message about correcting
33919 an incomplete last line of an input file may be generated, but need not affect the final exit status.

33920 OUTPUT FILES

33921 If the **-o** option is in effect, the sorted input shall be written to the file *output*.

33922 EXTENDED DESCRIPTION

33923 The notation:

33924 **-k** *field_start[type]*[**,***field_end[type]*]**]**

33925 shall define a key field that begins at *field_start* and ends at *field_end* inclusive, unless *field_start*
33926 falls beyond the end of the line or after *field_end*, in which case the key field is empty. A missing
33927 *field_end* shall mean the last character of the line.

33928 A field comprises a maximal sequence of non-separating characters and, in the absence of option
33929 **-t**, any preceding field separator.

33930 The *field_start* portion of the *keydef* option-argument shall have the form:

33931 *field_number*[**.***first_character*]**]**

33932 Fields and characters within fields shall be numbered starting with 1. The *field_number* and
33933 *first_character* pieces, interpreted as positive decimal integers, shall specify the first character to
33934 be used as part of a sort key. If *first_character* is omitted, it shall refer to the first character of the

33935 field.

33936 The *field_end* portion of the *keydef* option-argument shall have the form:

33937 *field_number*[.*last_character*]

33938 The *field_number* shall be as described above for *field_start*. The *last_character* piece, interpreted
33939 as a non-negative decimal integer, shall specify the last character to be used as part of the sort
33940 key. If *last_character* evaluates to zero or *.last_character* is omitted, it shall refer to the last
33941 character of the field specified by *field_number*.

33942 If the **-b** option or **b** type modifier is in effect, characters within a field shall be counted from the
33943 first non-<blank> in the field. (This shall apply separately to *first_character* and *last_character*.)

33944 EXIT STATUS

33945 The following exit values shall be returned:

- 33946 0 All input files were output successfully, or **-c** was specified and the input file was correctly
33947 sorted.
- 33948 1 Under the **-c** option, the file was not ordered as specified, or if the **-c** and **-u** options were
33949 both specified, two input lines were found with equal keys.
- 33950 >1 An error occurred.

33951 CONSEQUENCES OF ERRORS

33952 Default.

33953 APPLICATION USAGE

33954 The default value for **-t**, <blank>, has different properties from, for example, **-t "<space>"**. If a
33955 line contains:

33956 <space><space>foo

33957 the following treatment would occur with default separation as opposed to specifically selecting
33958 a <space>:

Field	Default	-t "<space>"
1	<space><space>foo	<i>empty</i>
2	<i>empty</i>	<i>empty</i>
3	<i>empty</i>	foo

33963 The leading field separator itself is included in a field when **-t** is not used. For example, this
33964 command returns an exit status of zero, meaning the input was already sorted:

```
33965 sort -c -k 2 <<eof
33966 y<tab>b
33967 x<space>a
33968 eof
```

33969 (assuming that a <tab> precedes the <space> in the current collating sequence). The field
33970 separator is not included in a field when it is explicitly set via **-t**. This is historical practice and
33971 allows usage such as:

```
33972 sort -t " | " -k 2n <<eof
33973 Atlanta|425022|Georgia
33974 Birmingham|284413|Alabama
33975 Columbia|100385|South Carolina
33976 eof
```

33977 where the second field can be correctly sorted numerically without regard to the non-numeric
33978 field separator.

33979 The wording in the OPTIONS section clarifies that the **-b**, **-d**, **-f**, **-i**, **-n**, and **-r** options have to
33980 come before the first sort key specified if they are intended to apply to all specified keys. The
33981 way it is described in this volume of IEEE Std 1003.1-2001 matches historical practice, not
33982 historical documentation. The results are unspecified if these options are specified after a **-k**
33983 option.

33984 The **-f** option might not work as expected in locales where there is not a one-to-one mapping
33985 between an uppercase and a lowercase letter.

33986 EXAMPLES

1. The following command sorts the contents of **infile** with the second field as the sort key:

```
33988     sort -k 2,2 infile
```

2. The following command sorts, in reverse order, the contents of **infile1** and **infile2**, placing
33990 the output in **outfile** and using the second character of the second field as the sort key
33991 (assuming that the first character of the second field is the field separator):

```
33992     sort -r -o outfile -k 2.2,2.2 infile1 infile2
```

3. The following command sorts the contents of **infile1** and **infile2** using the second non-
33994 <blank> of the second field as the sort key:

```
33995     sort -k 2.2b,2.2b infile1 infile2
```

4. The following command prints the System V password file (user database) sorted by the
33996 numeric user ID (the third colon-separated field):

```
33998     sort -t : -k 3,3n /etc/passwd
```

5. The following command prints the lines of the already sorted file **infile**, suppressing all
34000 but one occurrence of lines having the same third field:

```
34001     sort -um -k 3.1,3.0 infile
```

34002 RATIONALE

34003 Examples in some historical documentation state that options **-um** with one input file keep the
34004 first in each set of lines with equal keys. This behavior was deemed to be an implementation
34005 artifact and was not standardized.

34006 The **-z** option was omitted; it is not standard practice on most systems and is inconsistent with
34007 using **sort** to sort several files individually and then merge them together. The text concerning **-z**
34008 in historical documentation appeared to require implementations to determine the proper buffer
34009 length during the sort phase of operation, but not during the merge.

34010 The **-y** option was omitted because of non-portability. The **-M** option, present in System V, was
34011 omitted because of non-portability in international usage.

34012 An undocumented **-T** option exists in some implementations. It is used to specify a directory for
34013 intermediate files. Implementations are encouraged to support the use of the **TMPDIR**
34014 environment variable instead of adding an option to support this functionality.

34015 The **-k** option was added to satisfy two objections. First, the zero-based counting used by **sort** is
34016 not consistent with other utility conventions. Second, it did not meet syntax guideline
34017 requirements.

34018 Historical documentation indicates that “setting **-n** implies **-b**”. The description of **-n** already
34019 states that optional leading <blank>s are tolerated in doing the comparison. If **-b** is enabled,

34020 rather than implied, by **-n**, this has unusual side effects. When a character offset is used in a
34021 column of numbers (for example, to sort modulo 100), that offset is measured relative to the
34022 most significant digit, not to the column. Based upon a recommendation from the author of the
34023 original *sort* utility, the **-b** implication has been omitted from this volume of
34024 IEEE Std 1003.1-2001, and an application wishing to achieve the previously mentioned side
34025 effects has to code the **-b** flag explicitly.

34026 **FUTURE DIRECTIONS**

34027 None.

34028 **SEE ALSO**

34029 *comm*, *join*, *uniq*, the System Interfaces volume of IEEE Std 1003.1-2001, *toupper()*

34030 **CHANGE HISTORY**

34031 First released in Issue 2.

34032 **Issue 6**

34033 IEEE PASC Interpretation 1003.2 #174 is applied, updating the DESCRIPTION of comparisons.

34034 IEEE PASC Interpretation 1003.2 #168 is applied.

34035 NAME

34036 split — split files into pieces

34037 SYNOPSIS

34038 UP `split [-l line_count][-a suffix_length][file[name]]`34039 `split -b n[k|m][-a suffix_length][file[name]]`

34040

34041 DESCRIPTION

34042 The *split* utility shall read an input file and write one or more output files. The default size of
 34043 each output file shall be 1 000 lines. The size of the output files can be modified by specification
 34044 of the **-b** or **-l** options. Each output file shall be created with a unique suffix. The suffix shall
 34045 consist of exactly *suffix_length* lowercase letters from the POSIX locale. The letters of the suffix
 34046 shall be used as if they were a base-26 digit system, with the first suffix to be created consisting
 34047 of all 'a' characters, the second with a 'b' replacing the last 'a', and so on, until a name of all
 34048 'z' characters is created. By default, the names of the output files shall be 'x', followed by a
 34049 two-character suffix from the character set as described above, starting with "aa", "ab", "ac",
 34050 and so on, and continuing until the suffix "zz", for a maximum of 676 files.

34051 If the number of files required exceeds the maximum allowed by the suffix length provided,
 34052 such that the last allowable file would be larger than the requested size, the *split* utility shall fail
 34053 after creating the last file with a valid suffix; *split* shall not delete the files it created with valid
 34054 suffixes. If the file limit is not exceeded, the last file created shall contain the remainder of the
 34055 input file, and may be smaller than the requested size.

34056 OPTIONS

34057 The *split* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section
 34058 12.2, Utility Syntax Guidelines.

34059 The following options shall be supported:

34060 **-a** *suffix_length*

34061 Use *suffix_length* letters to form the suffix portion of the filenames of the split file.
 34062 If **-a** is not specified, the default suffix length shall be two. If the sum of the *name*
 34063 operand and the *suffix_length* option-argument would create a filename exceeding
 34064 {NAME_MAX} bytes, an error shall result; *split* shall exit with a diagnostic
 34065 message and no files shall be created.

34066 **-b** *n* Split a file into pieces *n* bytes in size.

34067 **-b** *nk* Split a file into pieces *n**1 024 bytes in size.

34068 **-b** *nm* Split a file into pieces *n**1 048 576 bytes in size.

34069 **-l** *line_count* Specify the number of lines in each resulting file piece. The *line_count* argument is
 34070 an unsigned decimal integer. The default is 1 000. If the input does not end with a
 34071 <newline>, the partial line shall be included in the last output file.

34072 OPERANDS

34073 The following operands shall be supported:

34074 *file* The pathname of the ordinary file to be split. If no input file is given or *file* is '**-**',
 34075 the standard input shall be used.

34076 *name* The prefix to be used for each of the files resulting from the split operation. If no
 34077 *name* argument is given, 'x' shall be used as the prefix of the output files. The
 34078 combined length of the basename of *prefix* and *suffix_length* cannot exceed
 34079 {NAME_MAX} bytes. See the OPTIONS section.

34080 STDIN

34081 See the INPUT FILES section.

34082 INPUT FILES

34083 Any file can be used as input.

34084 ENVIRONMENT VARIABLES

34085 The following environment variables shall affect the execution of *split*:

34086 *LANG* Provide a default value for the internationalization variables that are unset or null.
(See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2, Internationalization Variables for the precedence of internationalization variables used to determine the values of locale categories.)

34090 *LC_ALL* If set to a non-empty string value, override the values of all the other internationalization variables.

34092 *LC_CTYPE* Determine the locale for the interpretation of sequences of bytes of text data as characters (for example, single-byte as opposed to multi-byte characters in arguments and input files).

34095 *LC_MESSAGES*

34096 Determine the locale that should be used to affect the format and contents of diagnostic messages written to standard error.

34098 XSI *NLSPATH* Determine the location of message catalogs for the processing of *LC_MESSAGES*.

34099 ASYNCHRONOUS EVENTS

34100 Default.

34101 STDOUT

34102 Not used.

34103 STDERR

34104 The standard error shall be used only for diagnostic messages.

34105 OUTPUT FILES

34106 The output files contain portions of the original input file; otherwise, unchanged.

34107 EXTENDED DESCRIPTION

34108 None.

34109 EXIT STATUS

34110 The following exit values shall be returned:

34111 0 Successful completion.

34112 >0 An error occurred.

34113 CONSEQUENCES OF ERRORS

34114 Default.

34115 APPLICATION USAGE

34116 None.

34117 EXAMPLES

34118 In the following examples **foo** is a text file that contains 5 000 lines.

- 34119 1. Create five files, **xaa**, **xab**, **xac**, **xad**, and **xae**:

34120 `split foo`

- 34121 2. Create five files, but the suffixed portion of the created files consists of three letters, **aaaa**,
34122 **xaab**, **xaac**, **xaad**, and **xaae**:

34123 `split -a 3 foo`

- 34124 3. Create three files with four-letter suffixes and a supplied prefix, **bar_aaaa**, **bar_aaab**, and
34125 **bar_aaac**:

34126 `split -a 4 -l 2000 foo bar_`

- 34127 4. Create as many files as are necessary to contain at most 20*1 024 bytes, each with the
34128 default prefix of **x** and a five-letter suffix:

34129 `split -a 5 -b 20k foo`

34130 RATIONALE

34131 The **-b** option was added to provide a mechanism for splitting files other than by lines. While
34132 most uses of the **-b** option are for transmitting files over networks, some believed it would have
34133 additional uses.

34134 The **-a** option was added to overcome the limitation of being able to create only 676 files.

34135 Consideration was given to deleting this utility, using the rationale that the functionality
34136 provided by this utility is available via the *csplit* utility (see *csplit*). Upon reconsideration of the
34137 purpose of the User Portability Extension, it was decided to retain both this utility and the *csplit*
34138 utility because users use both utilities and have historical expectations of their behavior.
34139 Furthermore, the splitting on byte boundaries in *split* cannot be duplicated with the historical
34140 *csplit*.

34141 The text “*split* shall not delete the files it created with valid suffixes” would normally be
34142 assumed, but since the related utility, *csplit*, does delete files under some circumstances, the
34143 historical behavior of *split* is made explicit to avoid misinterpretation.

34144 FUTURE DIRECTIONS

34145 None.

34146 SEE ALSO

34147 *csplit*

34148 CHANGE HISTORY

34149 First released in Issue 2.

34150 Issue 6

34151 This utility is marked as part of the User Portability Utilities option.

34152 The APPLICATION USAGE section is added.

34153 The obsolescent SYNOPSIS is removed.

34154 NAME

34155 strings — find printable strings in files

34156 SYNOPSIS

34157 UP strings [-a][-t *format*][-n *number*][*file...*]

34158

34159 DESCRIPTION

34160 The *strings* utility shall look for printable strings in regular files and shall write those strings to
34161 standard output. A printable string is any sequence of four (by default) or more printable
34162 characters terminated by a <newline> or NUL character. Additional implementation-defined
34163 strings may be written; see *localedef*.

34164 OPTIONS

34165 The *strings* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section
34166 12.2, Utility Syntax Guidelines.

34167 The following options shall be supported:

34168 **-a** Scan files in their entirety. If **-a** is not specified, it is implementation-defined what
34169 portion of each file is scanned for strings.

34170 **-n** *number* Specify the minimum string length, where the *number* argument is a positive
34171 decimal integer. The default shall be 4.

34172 **-t** *format* Write each string preceded by its byte offset from the start of the file. The format
34173 shall be dependent on the single character used as the *format* option-argument:

34174 d The offset shall be written in decimal.

34175 o The offset shall be written in octal.

34176 x The offset shall be written in hexadecimal.

34177 OPERANDS

34178 The following operand shall be supported:

34179 **file** A pathname of a regular file to be used as input. If no *file* operand is specified, the
34180 *strings* utility shall read from the standard input.

34181 STDIN

34182 See the INPUT FILES section.

34183 INPUT FILES

34184 The input files named by the utility arguments or the standard input shall be regular files of any
34185 format.

34186 ENVIRONMENT VARIABLES

34187 The following environment variables shall affect the execution of *strings*:

34188 **LANG** Provide a default value for the internationalization variables that are unset or null.
34189 (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2,
34190 Internationalization Variables for the precedence of internationalization variables
34191 used to determine the values of locale categories.)

34192 **LC_ALL** If set to a non-empty string value, override the values of all the other
34193 internationalization variables.

34194 **LC_CTYPE** Determine the locale for the interpretation of sequences of bytes of text data as
34195 characters (for example, single-byte as opposed to multi-byte characters in
34196 arguments and input files) and to identify printable strings.

34197	LC_MESSAGES	Determine the locale that should be used to affect the format and contents of diagnostic messages written to standard error.
34198		
34199		
34200 XSI	NLSPATH	Determine the location of message catalogs for the processing of <i>LC_MESSAGES</i> .
34201	ASYNCHRONOUS EVENTS	
34202		Default.
34203	STDOUT	
34204		Strings found shall be written to the standard output, one per line.
34205		When the -t option is not specified, the format of the output shall be:
34206		"%s", <string>
34207		With the -t o option, the format of the output shall be:
34208		"%o %s", <byte offset>, <string>
34209		With the -t x option, the format of the output shall be:
34210		"%x %s", <byte offset>, <string>
34211		With the -t d option, the format of the output shall be:
34212		"%d %s", <byte offset>, <string>
34213	STDERR	
34214		The standard error shall be used only for diagnostic messages.
34215	OUTPUT FILES	
34216		None.
34217	EXTENDED DESCRIPTION	
34218		None.
34219	EXIT STATUS	
34220		The following exit values shall be returned:
34221		0 Successful completion.
34222		>0 An error occurred.
34223	CONSEQUENCES OF ERRORS	
34224		Default.
34225	APPLICATION USAGE	
34226		By default the data area (as opposed to the text, "bss", or header areas) of a binary executable file is scanned. Implementations document which areas are scanned.
34227		
34228		Some historical implementations do not require NUL or <newline> terminators for strings to permit those languages that do not use NUL as a string terminator to have their strings written.
34229		
34230	EXAMPLES	
34231		None.
34232	RATIONALE	
34233		Apart from rationalizing the option syntax and slight difficulties with object and executable binary files, <i>strings</i> is specified to match historical practice closely. The -a and -n options were introduced to replace the non-conforming - and -number options.
34234		
34235		
34236		The -o option historically means different things on different implementations. Some use it to mean " <i>offset</i> in decimal", while others use it as " <i>offset</i> in octal". Instead of trying to decide which
34237		

34238 way would be least objectionable, the **-t** option was added. It was originally named **-O** to mean
34239 “offset”, but was changed to **-t** to be consistent with *od*.

34240 The ISO C standard function *isprint()* is restricted to a domain of **unsigned char**. This volume of
34241 IEEE Std 1003.1-2001 requires implementations to write strings as defined by the current locale.

34242 FUTURE DIRECTIONS

34243 None.

34244 SEE ALSO

34245 *localedef, nm*

34246 CHANGE HISTORY

34247 First released in Issue 4.

34248 Issue 6

34249 This utility is marked as part of the User Portability Utilities option.

34250 The obsolescent SYNOPSIS is removed.

34251 The normative text is reworded to avoid use of the term “must” for application requirements.

34252 NAME

34253 strip — remove unnecessary information from executable files (**DEVELOPMENT**)

34254 SYNOPSIS

34255 SD `strip file...`

34256

34257 DESCRIPTION

34258 The *strip* utility shall remove from executable files named by the *file* operands any information
34259 the implementor deems unnecessary for execution of those files. The nature of that information
34260 is unspecified. The effect of *strip* shall be similar to the use of the *-s* option to *c99* or *fort77*.

34261 OPTIONS

34262 None.

34263 OPERANDS

34264 The following operand shall be supported:

34265 *file* A pathname referring to an executable file.

34266 STDIN

34267 Not used.

34268 INPUT FILES

34269 The input files shall be in the form of executable files successfully produced by any compiler
34270 defined by this volume of IEEE Std 1003.1-2001.

34271 ENVIRONMENT VARIABLES

34272 The following environment variables shall affect the execution of *strip*:

34273 *LANG* Provide a default value for the internationalization variables that are unset or null.
34274 (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2,
34275 Internationalization Variables for the precedence of internationalization variables
34276 used to determine the values of locale categories.)

34277 *LC_ALL* If set to a non-empty string value, override the values of all the other
34278 internationalization variables.

34279 *LC_CTYPE* Determine the locale for the interpretation of sequences of bytes of text data as
34280 characters (for example, single-byte as opposed to multi-byte characters in
34281 arguments).

34282 *LC_MESSAGES*

34283 Determine the locale that should be used to affect the format and contents of
34284 diagnostic messages written to standard error.

34285 XSI *NLSPATH* Determine the location of message catalogs for the processing of *LC_MESSAGES*.

34286 ASYNCHRONOUS EVENTS

34287 Default.

34288 STDOUT

34289 Not used.

34290 STDERR

34291 The standard error shall be used only for diagnostic messages.

34292 OUTPUT FILES

34293 The *strip* utility shall produce executable files of unspecified format.

34294 EXTENDED DESCRIPTION

34295 None.

34296 EXIT STATUS

34297 The following exit values shall be returned:

34298 0 Successful completion.

34299 >0 An error occurred.

34300 CONSEQUENCES OF ERRORS

34301 Default.

34302 APPLICATION USAGE

34303 None.

34304 EXAMPLES

34305 None.

34306 RATIONALE

34307 Historically, this utility has been used to remove the symbol table from an executable file. It was
34308 included since it is known that the amount of symbolic information can amount to several
34309 megabytes; the ability to remove it in a portable manner was deemed important, especially for
34310 smaller systems.

34311 The behavior of *strip* is said to be the same as the *-s* option to a compiler. While the end result is
34312 essentially the same, it is not required to be identical.

34313 FUTURE DIRECTIONS

34314 None.

34315 SEE ALSO

34316 *ar*, *c99*, *fort77*

34317 CHANGE HISTORY

34318 First released in Issue 2.

34319 Issue 6

34320 This utility is marked as part of the Software Development Utilities option.

34321 NAME

34322 **stty** — set the options for a terminal

34323 SYNOPSIS

34324 **stty** [**-a** | **-g**]

34325 **stty** *operands*

34326 DESCRIPTION

34327 The **stty** utility shall set or report on terminal I/O characteristics for the device that is its
34328 standard input. Without options or operands specified, it shall report the settings of certain
34329 characteristics, usually those that differ from implementation-defined defaults. Otherwise, it
34330 shall modify the terminal state according to the specified operands. Detailed information about
34331 the modes listed in the first five groups below are described in the Base Definitions volume of
34332 IEEE Std 1003.1-2001, Chapter 11, General Terminal Interface. Operands in the Combination
34333 Modes group (see **Combination Modes** (on page 890)) are implemented using operands in the
34334 previous groups. Some combinations of operands are mutually-exclusive on some terminal
34335 types; the results of using such combinations are unspecified.

34336 Typical implementations of this utility require a communications line configured to use the
34337 **termios** interface defined in the System Interfaces volume of IEEE Std 1003.1-2001. On systems
34338 where none of these lines are available, and on lines not currently configured to support the
34339 **termios** interface, some of the operands need not affect terminal characteristics.

34340 OPTIONS

34341 The **stty** utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section
34342 12.2, Utility Syntax Guidelines.

34343 The following options shall be supported:

34344 **-a** Write to standard output all the current settings for the terminal.

34345 **-g** Write to standard output all the current settings in an unspecified form that can be
34346 used as arguments to another invocation of the **stty** utility on the same system. The
34347 form used shall not contain any characters that would require quoting to avoid
34348 word expansion by the shell; see Section 2.6 (on page 36).

34349 OPERANDS

34350 The following operands shall be supported to set the terminal characteristics.

34351 Control Modes

34352 **parenb** (**-parenb**) Enable (disable) parity generation and detection. This shall have the effect of
34353 setting (not setting) PARENB in the **termios** *c_cflag* field, as defined in the
34354 Base Definitions volume of IEEE Std 1003.1-2001, Chapter 11, General
34355 Terminal Interface.

34356 parodd (**-parodd**)

34357 Select odd (even) parity. This shall have the effect of setting (not setting)
34358 PARODD in the **termios** *c_cflag* field, as defined in the Base Definitions
34359 volume of IEEE Std 1003.1-2001, Chapter 11, General Terminal Interface.

34360 cs5 cs6 cs7 cs8

34361 Select character size, if possible. This shall have the effect of setting CS5, CS6,
34362 CS7, and CS8, respectively, in the **termios** *c_cflag* field, as defined in the Base
34363 Definitions volume of IEEE Std 1003.1-2001, Chapter 11, General Terminal
Interface.

34364 number

34365 Set terminal baud rate to the number given, if possible. If the baud rate is set
to zero, the modem control lines shall no longer be asserted. This shall have

34366	the effect of setting the input and output termios baud rate values as defined in the Base Definitions volume of IEEE Std 1003.1-2001, Chapter 11, General Terminal Interface.	
34367		
34368		
34369	ispeed <i>number</i>	Set terminal input baud rate to the number given, if possible. If the input baud rate is set to zero, the input baud rate shall be specified by the value of the output baud rate. This shall have the effect of setting the input termios baud rate values as defined in the Base Definitions volume of IEEE Std 1003.1-2001, Chapter 11, General Terminal Interface.
34370		
34371		
34372		
34373		
34374	ospeed <i>number</i>	Set terminal output baud rate to the number given, if possible. If the output baud rate is set to zero, the modem control lines shall no longer be asserted. This shall have the effect of setting the output termios baud rate values as defined in the Base Definitions volume of IEEE Std 1003.1-2001, Chapter 11, General Terminal Interface.
34375		
34376		
34377		
34378		
34379	hupcl (-hupcl)	Stop asserting modem control lines (do not stop asserting modem control lines) on last close. This shall have the effect of setting (not setting) HUPCL in the termios <i>c_cflag</i> field, as defined in the Base Definitions volume of IEEE Std 1003.1-2001, Chapter 11, General Terminal Interface.
34380		
34381		
34382		
34383	hup (-hup)	Equivalent to hupcl (-hupcl).
34384	cstopb (-cstopb)	Use two (one) stop bits per character. This shall have the effect of setting (not setting) CSTOPB in the termios <i>c_cflag</i> field, as defined in the Base Definitions volume of IEEE Std 1003.1-2001, Chapter 11, General Terminal Interface.
34385		
34386		
34387	cread (-cread)	Enable (disable) the receiver. This shall have the effect of setting (not setting) CREAD in the termios <i>c_cflag</i> field, as defined in the Base Definitions volume of IEEE Std 1003.1-2001, Chapter 11, General Terminal Interface.
34388		
34389		
34390	clocal (-clocal)	Assume a line without (with) modem control. This shall have the effect of setting (not setting) CLOCAL in the termios <i>c_cflag</i> field, as defined in the Base Definitions volume of IEEE Std 1003.1-2001, Chapter 11, General Terminal Interface.
34391		
34392		
34393		
34394	It is unspecified whether <i>stty</i> shall report an error if an attempt to set a Control Mode fails.	
34395	Input Modes	
34396	ignbrk (-ignbrk)	Ignore (do not ignore) break on input. This shall have the effect of setting (not setting) IGNBRK in the termios <i>c_iflag</i> field, as defined in the Base Definitions volume of IEEE Std 1003.1-2001, Chapter 11, General Terminal Interface.
34397		
34398		
34399	brkint (-brkint)	Signal (do not signal) INTR on break. This shall have the effect of setting (not setting) BRKINT in the termios <i>c_iflag</i> field, as defined in the Base Definitions volume of IEEE Std 1003.1-2001, Chapter 11, General Terminal Interface.
34400		
34401		
34402	ignpar (-ignpar)	Ignore (do not ignore) bytes with parity errors. This shall have the effect of setting (not setting) IGNPAR in the termios <i>c_iflag</i> field, as defined in the Base Definitions volume of IEEE Std 1003.1-2001, Chapter 11, General Terminal Interface.
34403		
34404		
34405		
34406	parmrk (-parmrk)	Mark (do not mark) parity errors. This shall have the effect of setting (not setting) PARMRK in the termios <i>c_iflag</i> field, as defined in the Base Definitions volume of IEEE Std 1003.1-2001, Chapter 11, General Terminal Interface.
34407		
34408		
34409		
34410		

34411	inpck (-inpck)	Enable (disable) input parity checking. This shall have the effect of setting (not setting) INPCK in the termios c_iflag field, as defined in the Base Definitions volume of IEEE Std 1003.1-2001, Chapter 11, General Terminal Interface.
34412		
34413		
34414	istrip (-istrip)	Strip (do not strip) input characters to seven bits. This shall have the effect of setting (not setting) ISTRIP in the termios c_iflag field, as defined in the Base Definitions volume of IEEE Std 1003.1-2001, Chapter 11, General Terminal Interface.
34415		
34416		
34417		
34418	inlcr (-inlcr)	Map (do not map) NL to CR on input. This shall have the effect of setting (not setting) INLCR in the termios c_iflag field, as defined in the Base Definitions volume of IEEE Std 1003.1-2001, Chapter 11, General Terminal Interface.
34419		
34420		
34421	igncr (-igncr)	Ignore (do not ignore) CR on input. This shall have the effect of setting (not setting) IGNCR in the termios c_iflag field, as defined in the Base Definitions volume of IEEE Std 1003.1-2001, Chapter 11, General Terminal Interface.
34422		
34423		
34424	icrnl (-icrnl)	Map (do not map) CR to NL on input. This shall have the effect of setting (not setting) ICRNL in the termios c_iflag field, as defined in the Base Definitions volume of IEEE Std 1003.1-2001, Chapter 11, General Terminal Interface.
34425		
34426		
34427	ixon (-ixon)	Enable (disable) START/STOP output control. Output from the system is stopped when the system receives STOP and started when the system receives START. This shall have the effect of setting (not setting) IXON in the termios c_iflag field, as defined in the Base Definitions volume of IEEE Std 1003.1-2001, Chapter 11, General Terminal Interface.
34428		
34429		
34430		
34431		
34432 XSI	ixany (-ixany)	Allow any character to restart output. This shall have the effect of setting (not setting) IXANY in the termios c_iflag field, as defined in the Base Definitions volume of IEEE Std 1003.1-2001, Chapter 11, General Terminal Interface.
34433		
34434		
34435	ixoff (-ixoff)	Request that the system send (not send) STOP characters when the input queue is nearly full and START characters to resume data transmission. This shall have the effect of setting (not setting) IXOFF in the termios c_iflag field, as defined in the Base Definitions volume of IEEE Std 1003.1-2001, Chapter 11, General Terminal Interface.
34436		
34437		
34438		
34439		
34440	Output Modes	
34441	opost (-opost)	Post-process output (do not post-process output; ignore all other output modes). This shall have the effect of setting (not setting) OPOST in the termios c_oflag field, as defined in the Base Definitions volume of IEEE Std 1003.1-2001, Chapter 11, General Terminal Interface.
34442		
34443		
34444		
34445 XSI	ocrnl (-ocrnl)	Map (do not map) CR to NL on output. This shall have the effect of setting (not setting) OCRNL in the termios c_oflag field, as defined in the Base Definitions volume of IEEE Std 1003.1-2001, Chapter 11, General Terminal Interface.
34446		
34447		
34448		
34449	onocr (-onocr)	Do not (do) output CR at column zero. This shall have the effect of setting (not setting) ONOCR in the termios c_oflag field, as defined in the Base Definitions volume of IEEE Std 1003.1-2001, Chapter 11, General Terminal Interface.
34450		
34451		
34452	onlret (-onlret)	The terminal newline key performs (does not perform) the CR function. This shall have the effect of setting (not setting) ONLRET in the termios c_oflag field, as defined in the Base Definitions volume of IEEE Std 1003.1-2001, Chapter 11, General Terminal Interface.
34453		
34454		
34455		

34456	ofill (-ofill)	Use fill characters (use timing) for delays. This shall have the effect of setting (not setting) OFILL in the termios c_oflag field, as defined in the Base Definitions volume of IEEE Std 1003.1-2001, Chapter 11, General Terminal Interface.
34460	ofdel (-ofdel)	Fill characters are DELs (NULs). This shall have the effect of setting (not setting) OFDEL in the termios c_oflag field, as defined in the Base Definitions volume of IEEE Std 1003.1-2001, Chapter 11, General Terminal Interface.
34463	cr0 cr1 cr2 cr3	Select the style of delay for CRs. This shall have the effect of setting CRDLY to CR0, CR1, CR2, or CR3, respectively, in the termios c_oflag field, as defined in the Base Definitions volume of IEEE Std 1003.1-2001, Chapter 11, General Terminal Interface.
34467	nl0 nl1	Select the style of delay for NL. This shall have the effect of setting NLDLY to NL0 or NL1, respectively, in the termios c_oflag field, as defined in the Base Definitions volume of IEEE Std 1003.1-2001, Chapter 11, General Terminal Interface.
34471	tab0 tab1 tab2 tab3	Select the style of delay for horizontal tabs. This shall have the effect of setting TABDLY to TAB0, TAB1, TAB2, or TAB3, respectively, in the termios c_oflag field, as defined in the Base Definitions volume of IEEE Std 1003.1-2001, Chapter 11, General Terminal Interface. Note that TAB3 has the effect of expanding <tab>s to <space>s.
34477	tabs (-tabs)	Synonym for tab0 (tab3) .
34478	bs0 bs1	Select the style of delay for backspaces. This shall have the effect of setting BSDLY to BS0 or BS1, respectively, in the termios c_oflag field, as defined in the Base Definitions volume of IEEE Std 1003.1-2001, Chapter 11, General Terminal Interface.
34482	ff0 ff1	Select the style of delay for form-feeds. This shall have the effect of setting FFDLY to FF0 or FF1, respectively, in the termios c_oflag field, as defined in the Base Definitions volume of IEEE Std 1003.1-2001, Chapter 11, General Terminal Interface.
34486	vt0 vt1	Select the style of delay for vertical-tabs. This shall have the effect of setting VTDLY to VT0 or VT1, respectively, in the termios c_oflag field, as defined in the Base Definitions volume of IEEE Std 1003.1-2001, Chapter 11, General Terminal Interface.

34490 Local Modes

34491	isig (-isig)	Enable (disable) the checking of characters against the special control characters INTR, QUIT, and SUSP. This shall have the effect of setting (not setting) ISIG in the termios c_lflag field, as defined in the Base Definitions volume of IEEE Std 1003.1-2001, Chapter 11, General Terminal Interface.
34495	icanon (-icanon)	Enable (disable) canonical input (ERASE and KILL processing). This shall have the effect of setting (not setting) ICANON in the termios c_lflag field, as defined in the Base Definitions volume of IEEE Std 1003.1-2001, Chapter 11, General Terminal Interface.
34499	iexten (-iexten)	Enable (disable) any implementation-defined special control characters not currently controlled by icanon , isig , ixon , or ixoff . This shall have the effect of setting (not setting) IEXTEN in the termios c_lflag field, as defined in the Base

34502		Definitions volume of IEEE Std 1003.1-2001, Chapter 11, General Terminal Interface.
34503		
34504	echo (-echo)	Echo back (do not echo back) every character typed. This shall have the effect of setting (not setting) ECHO in the termios c_lflag field, as defined in the Base Definitions volume of IEEE Std 1003.1-2001, Chapter 11, General Terminal Interface.
34505		
34506		
34507		
34508	echoe (-echoe)	The ERASE character visually erases (does not erase) the last character in the current line from the display, if possible. This shall have the effect of setting (not setting) ECHOE in the termios c_lflag field, as defined in the Base Definitions volume of IEEE Std 1003.1-2001, Chapter 11, General Terminal Interface.
34509		
34510		
34511		
34512		
34513	echok (-echok)	Echo (do not echo) NL after KILL character. This shall have the effect of setting (not setting) ECHOK in the termios c_lflag field, as defined in the Base Definitions volume of IEEE Std 1003.1-2001, Chapter 11, General Terminal Interface.
34514		
34515		
34516		
34517	echonl (-echonl)	Echo (do not echo) NL, even if echo is disabled. This shall have the effect of setting (not setting) ECHONL in the termios c_lflag field, as defined in the Base Definitions volume of IEEE Std 1003.1-2001, Chapter 11, General Terminal Interface.
34518		
34519		
34520		
34521	noflsh (-noflsh)	Disable (enable) flush after INTR, QUIT, SUSP. This shall have the effect of setting (not setting) NOFLSH in the termios c_lflag field, as defined in the Base Definitions volume of IEEE Std 1003.1-2001, Chapter 11, General Terminal Interface.
34522		
34523		
34524		
34525	tostop (-tostop)	Send SIGTTOU for background output. This shall have the effect of setting (not setting) TOSTOP in the termios c_lflag field, as defined in the Base Definitions volume of IEEE Std 1003.1-2001, Chapter 11, General Terminal Interface.
34526		
34527		
34528		

34529 Special Control Character Assignments

34530 <*control>-character string*

34531 Set <*control>-character* to *string*. If <*control>-character* is one of the character sequences in
34532 the first column of the following table, the corresponding Base Definitions volume of
34533 IEEE Std 1003.1-2001, Chapter 11, General Terminal Interface control character from the
34534 second column shall be recognized. This has the effect of setting the corresponding element
34535 of the **termios c_cc** array (see the Base Definitions volume of IEEE Std 1003.1-2001, Chapter
34536 13, Headers, <termios.h>).

34537

Table 4-19 Control Character Names in *stty*

Control Character	c_cc Subscript	Description
eof	VEOF	EOF character
eol	VEOL	EOL character
erase	VERASE	ERASE character
intr	VINTR	INTR character
kill	VKILL	KILL character
quit	VQUIT	QUIT character
susp	VSUSP	SUSP character
start	VSTART	START character
stop	VSTOP	STOP character

If *string* is a single character, the control character shall be set to that character. If *string* is the two-character sequence " ^- " or the string *undef*, the control character shall be set to _POSIX_VDISABLE , if it is in effect for the device; if _POSIX_VDISABLE is not in effect for the device, it shall be treated as an error. In the POSIX locale, if *string* is a two-character sequence beginning with circumflex (' ^'), and the second character is one of those listed in the " ^c " column of the following table, the control character shall be set to the corresponding character value in the Value column of the table.

34555

Table 4-20 Circumflex Control Characters in *stty*

^c	Value	^c	Value	^c	Value
a, A	<SOH>	l, L	<FF>	w, W	<ETB>
b, B	<STX>	m, M	<CR>	x, X	<CAN>
c, C	<ETX>	n, N	<SO>	y, Y	
d, D	<EOT>	o, O	<SI>	z, Z	<SUB>
e, E	<ENQ>	p, P	<DLE>	[<ESC>
f, F	<ACK>	q, Q	<DC1>	\	<FS>
g, G	<BEL>	r, R	<DC2>]	<GS>
h, H	<BS>	s, S	<DC3>	^	<RS>
i, I	<HT>	t, T	<DC4>	—	<US>
j, J	<LF>	u, U	<NAK>	?	
k, K	<VT>	v, V	<SYN>		

34568

min number34569
34570

Set the value of MIN to *number*. MIN is used in non-canonical mode input processing (*icanon*).

34571

time number34572
34573

Set the value of TIME to *number*. TIME is used in non-canonical mode input processing (*icanon*).

34574

Combination Modes34575
34576**saved settings**

Set the current terminal characteristics to the saved settings produced by the -g option.

34577
34578**evenp or parity**

Enable **parenb** and **cs7**; disable **parodd**.

34579
34580**oddp**

Enable **parenb**, **cs7**, and **parodd**.

34581	-parity, -evenp, or -oddp	
34582	Disable parenb , and set cs8 .	
34583 XSI	raw (-raw or cooked)	
34584	Enable (disable) raw input and output. Raw mode shall be equivalent to setting:	
34585	<code>stty cs8 erase ^- kill ^- intr ^- \ quit ^- eof ^- eol ^- -post -inpck</code>	
34586		
34587	nl (-nl)	
34588	Disable (enable) icrl . In addition, -nl unsets inlcr and igncr .	1
34589	ek Reset ERASE and KILL characters back to system defaults.	
34590	sane	
34591	Reset all modes to some reasonable, unspecified, values.	
34592	STDIN	
34593	Although no input is read from standard input, standard input shall be used to get the current	
34594	terminal I/O characteristics and to set new terminal I/O characteristics.	
34595	INPUT FILES	
34596	None.	
34597	ENVIRONMENT VARIABLES	
34598	The following environment variables shall affect the execution of stty :	
34599	LANG	Provide a default value for the internationalization variables that are unset or null. (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2, Internationalization Variables for the precedence of internationalization variables used to determine the values of locale categories.)
34600		
34601		
34602		
34603	LC_ALL	If set to a non-empty string value, override the values of all the other internationalization variables.
34604		
34605	LC_CTYPE	This variable determines the locale for the interpretation of sequences of bytes of text data as characters (for example, single-byte as opposed to multi-byte characters in arguments) and which characters are in the class print .
34606		
34607		
34608	LC_MESSAGES	
34609	Determine the locale that should be used to affect the format and contents of diagnostic messages written to standard error.	
34610		
34611 XSI	NLSPATH	Determine the location of message catalogs for the processing of LC_MESSAGES .
34612	ASYNCHRONOUS EVENTS	
34613	Default.	
34614	STDOUT	
34615	If operands are specified, no output shall be produced.	
34616	If the -g option is specified, stty shall write to standard output the current settings in a form that can be used as arguments to another instance of stty on the same system.	
34617		
34618	If the -a option is specified, all of the information as described in the OPERANDS section shall be written to standard output. Unless otherwise specified, this information shall be written as <space>-separated tokens in an unspecified format, on one or more lines, with an unspecified number of tokens per line. Additional information may be written.	
34619		
34620		
34621		
34622	If no options or operands are specified, an unspecified subset of the information written for the -a option shall be written.	
34623		

34624 If speed information is written as part of the default output, or if the **-a** option is specified and if
34625 the terminal input speed and output speed are the same, the speed information shall be written
34626 as follows:

34627 "speed %d baud; ", <speed>

34628 Otherwise, speeds shall be written as:

34629 "ispeed %d baud; ospeed %d baud; ", <ispeed>, <ospeed>

34630 In locales other than the POSIX locale, the word **baud** may be changed to something more
34631 appropriate in those locales.

34632 If control characters are written as part of the default output, or if the **-a** option is specified,
34633 control characters shall be written as:

34634 "%s = %s; ", <control-character name>, <value>

34635 where <value> is either the character, or some visual representation of the character if it is non-
34636 printable, or the string *undef* if the character is disabled.

34637 STDERR

34638 The standard error shall be used only for diagnostic messages.

34639 OUTPUT FILES

34640 None.

34641 EXTENDED DESCRIPTION

34642 None.

34643 EXIT STATUS

34644 The following exit values shall be returned:

34645 0 The terminal options were read or set successfully.

34646 >0 An error occurred.

34647 CONSEQUENCES OF ERRORS

34648 Default.

34649 APPLICATION USAGE

34650 The **-g** flag is designed to facilitate the saving and restoring of terminal state from the shell level.

34651 For example, a program may:

```
34652 saveterm="$(stty -g)"      # save terminal state
34653 stty (new settings)       # set new state
34654 ...
34655 stty $saveterm          # restore terminal state
```

34656 Since the format is unspecified, the saved value is not portable across systems.

34657 Since the **-a** format is so loosely specified, scripts that save and restore terminal settings should
34658 use the **-g** option.

34659 EXAMPLES

34660 None.

34661 RATIONALE

34662 The original *stty* description was taken directly from System V and reflected the System V
34663 terminal driver **termio**. It has been modified to correspond to the terminal driver **termios**.

34664 Output modes are specified only for XSI-conformant systems. All implementations are expected
34665 to provide *stty* operands corresponding to all of the output modes they support.

34666 The **stty** utility is primarily used to tailor the user interface of the terminal, such as selecting the
34667 preferred ERASE and KILL characters. As an application programming utility, **stty** can be used
34668 within shell scripts to alter the terminal settings for the duration of the script.

34669 The **termios** section states that individual disabling of control characters is possible through the
34670 option **_POSIX_VDISABLE**. If enabled, two conventions currently exist for specifying this:
34671 System V uses "**^-**", and BSD uses **undef**. Both are accepted by **stty** in this volume of
34672 IEEE Std 1003.1-2001. The other BSD convention of using the letter '**u**' was rejected because it
34673 conflicts with the actual letter '**u**', which is an acceptable value for a control character.

34674 Early proposals did not specify the mapping of "**^c**" to control characters because the control
34675 characters were not specified in the POSIX locale character set description file requirements. The
34676 control character set is now specified in the Base Definitions volume of IEEE Std 1003.1-2001,
34677 Chapter 3, Definitions so the historical mapping is specified. Note that although the mapping
34678 corresponds to control-character key assignments on many terminals that use the
34679 ISO/IEC 646: 1991 standard (or ASCII) character encodings, the mapping specified here is to the
34680 control characters, not their keyboard encodings.

34681 Since **termios** supports separate speeds for input and output, two new options were added to
34682 specify each distinctly.

34683 Some historical implementations use standard input to get and set terminal characteristics;
34684 others use standard output. Since input from a login TTY is usually restricted to the owner while
34685 output to a TTY is frequently open to anyone, using standard input provides fewer chances of
34686 accidentally (or maliciously) altering the terminal settings of other users. Using standard input
34687 also allows **stty -a** and **stty -g** output to be redirected for later use. Therefore, usage of standard
34688 input is required by this volume of IEEE Std 1003.1-2001.

34689 FUTURE DIRECTIONS

34690 None.

34691 SEE ALSO

34692 Chapter 2 (on page 29), the Base Definitions volume of IEEE Std 1003.1-2001, Chapter 11,
34693 General Terminal Interface, <**termios.h**>

34694 CHANGE HISTORY

34695 First released in Issue 2.

34696 Issue 5

34697 The description of **tabs** is clarified.

34698 The FUTURE DIRECTIONS section is added.

34699 Issue 6

34700 The legacy items **iuclc(-iuclc)**, **xcase**, **olcuc(-olcuc)**, **lcase(-lcase)**, and **LCASE(-LCASE)** are
34701 removed.

34702 IEEE Std 1003.1-2001/Cor 1-2002, item XCU/TC1/D6/37 is applied, applying IEEE PASC 1
34703 Interpretation 1003.2 #133, fixing an error in the OPERANDS section for the Combination Modes 1
34704 **nl(-nl)**. 1

34705 NAME

34706 tabs — set terminal tabs

34707 SYNOPSIS

34708 UP XSI tabs [-n | -a | -a2 | -c | -c2 | -c3 | -f | -p | -s | -u][+m[n]] [-T type]

34709 tabs [-T type][+[n]] n1[,n2,...]

34710

34711 DESCRIPTION

34712 The *tabs* utility shall display a series of characters that first clears the hardware terminal tab settings and then initializes the tab stops at the specified positions and optionally adjusts the margin.

34715 The phrase “tab-stop position *N*” 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. The maximum number of tab stops allowed is terminal-dependent.

34718 It need not be possible to implement *tabs* on certain terminals. If the terminal type obtained from the *TERM* environment variable or *-T* option represents such a terminal, an appropriate diagnostic message shall be written to standard error and *tabs* shall exit with a status greater than zero.

34722 OPTIONS

34723 The *tabs* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section 12.2, Utility Syntax Guidelines, except for various extensions: the options **-a2**, **-c2**, and **-c3** are multi-character.

34726 The following options shall be supported:

34727 **-n** Specify repetitive tab stops separated by a uniform number of column positions, *n*, where *n* is a single-digit decimal number. The default usage of *tabs* with no arguments shall be equivalent to **tabs-8**. When **-0** is used, the tab stops shall be cleared and no new ones set.

34731 XSI **-a** 1,10,16,36,72
34732 Assembler, applicable to some mainframes.

34733 XSI **-a2** 1,10,16,40,72
34734 Assembler, applicable to some mainframes.

34735 XSI **-c** 1,8,12,16,20,55
34736 COBOL, normal format.

34737 XSI **-c2** 1,6,10,14,49
34738 COBOL, compact format (columns 1 to 6 omitted).

34739 XSI **-c3** 1,6,10,14,18,22,26,30,34,38,42,46,50,54,58,62,67
34740 COBOL compact format (columns 1 to 6 omitted), with more tabs than **-c2**.

34741 XSI **-f** 1,7,11,15,19,23
34742 FORTRAN

34743 XSI **-p** 1,5,9,13,17,21,25,29,33,37,41,45,49,53,57,61
34744 PL/1

34745 XSI **-s** 1,10,55
34746 SNOBOL

34747 XSI **-u** 1,12,20,44
34748 Assembler, applicable to some mainframes.

34749 **-T type** Indicate the type of terminal. If this option is not supplied and the *TERM* variable is unset or null, an unspecified default terminal type shall be used. The setting of *type* shall take precedence over the value in *TERM*.

34752 OPERANDS

34753 The following operand shall be supported:

34754 **n1[,n2,...]** A single command line argument that consists of tab-stop values separated using either commas or <blank>s. The application shall ensure that the tab-stop values are positive decimal integers in strictly ascending order. If any number (except the first one) is preceded by a plus sign, it is taken as an increment to be added to the previous value. For example, the tab lists 1,10,20,30 and 1,10,+10,+10 are considered to be identical.

34760 STDIN

34761 Not used.

34762 INPUT FILES

34763 None.

34764 ENVIRONMENT VARIABLES

34765 The following environment variables shall affect the execution of *tabs*:

34766 **LANG** Provide a default value for the internationalization variables that are unset or null. (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2, Internationalization Variables for the precedence of internationalization variables used to determine the values of locale categories.)

34770 **LC_ALL** If set to a non-empty string value, override the values of all the other internationalization variables.

34772 **LC_CTYPE** Determine the locale for the interpretation of sequences of bytes of text data as characters (for example, single-byte as opposed to multi-byte characters in arguments).

34775 **LC_MESSAGES**

34776 Determine the locale that should be used to affect the format and contents of diagnostic messages written to standard error.

34778 XSI **NLSPATH** Determine the location of message catalogs for the processing of *LC_MESSAGES*.

34779 **TERM** Determine the terminal type. If this variable is unset or null, and if the **-T** option is not specified, an unspecified default terminal type shall be used.

34781 ASYNCHRONOUS EVENTS

34782 Default.

34783 STDOUT

34784 If standard output is a terminal, the appropriate sequence to clear and set the tab stops may be written to standard output in an unspecified format. If standard output is not a terminal, undefined results occur.

34787 STDERR

34788 The standard error shall be used only for diagnostic messages.

34789 OUTPUT FILES

34790 None.

34791 **EXTENDED DESCRIPTION**

34792 None.

34793 **EXIT STATUS**

34794 The following exit values shall be returned:

34795 0 Successful completion.

34796 >0 An error occurred.

34797 **CONSEQUENCES OF ERRORS**

34798 Default.

34799 **APPLICATION USAGE**

34800 This utility makes use of the terminal's hardware tabs and the *stty tabs* option.

34801 This utility is not recommended for application use.

34802 Some integrated display units might not have escape sequences to set tab stops, but may be set
34803 by internal system calls. On these terminals, *tabs* works if standard output is directed to the
34804 terminal; if output is directed to another file, however, *tabs* fails.

34805 **EXAMPLES**

34806 None.

34807 **RATIONALE**

34808 Consideration was given to having the *tput* utility handle all of the functions described in *tabs*.
34809 However, the separate *tabs* utility was retained because it seems more intuitive to use a
34810 command named *tabs* than *tput* with a new option. The *tput* utility does not support setting or
34811 clearing tabs, and no known historical version of *tabs* supports the capability of setting arbitrary
34812 tab stops.

34813 The System V *tabs* interface is very complex; the version in this volume of IEEE Std 1003.1-2001
34814 has a reduced feature list, but many of the features omitted were restored as XSI extensions even
34815 though the supported languages and coding styles are primarily historical.

34816 There was considerable sentiment for specifying only a means of resetting the tabs back to a
34817 known state—presumably the “standard” of tabs every eight positions. The following features
34818 were omitted:

- 34819 • Setting tab stops via the first line in a file, using *--file*. Since even the SVID has no complete
34820 explanation of this feature, it is doubtful that it is in widespread use.

34821 In an early proposal, a *-t tablist* option was added for consistency with *expand*; this was later
34822 removed when inconsistencies with the historical list of tabs were identified.

34823 Consideration was given to adding a *-p* option that would output the current tab settings so
34824 that they could be saved and then later restored. This was not accepted because querying the tab
34825 stops of the terminal is not a capability in historical *terminfo* or *termcap* facilities and might not be
34826 supported on a wide range of terminals.

34827 **FUTURE DIRECTIONS**

34828 None.

34829 **SEE ALSO**

34830 *expand*, *stty*, *tput*, *unexpand*

34831 CHANGE HISTORY

34832 First released in Issue 2.

34833 Issue 6

34834 This utility is marked as part of the User Portability Utilities option.

34835 The normative text is reworded to avoid use of the term “must” for application requirements.

34836 NAME

34837 tail — copy the last part of a file

34838 SYNOPSIS

34839 tail [-f][-c number| -n number][file]

34840 DESCRIPTION

34841 The *tail* utility shall copy its input file to the standard output beginning at a designated place.

34842 Copying shall begin at the point in the file indicated by the **-c** *number* or **-n** *number* options. The
34843 option-argument *number* shall be counted in units of lines or bytes, according to the options **-n**
34844 and **-c**. Both line and byte counts start from 1.

34845 Tails relative to the end of the file may be saved in an internal buffer, and thus may be limited in
34846 length. Such a buffer, if any, shall be no smaller than {LINE_MAX}*10 bytes.

34847 OPTIONS

34848 The *tail* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section
34849 12.2, Utility Syntax Guidelines.

34850 The following options shall be supported:

34851 **-c** *number* The application shall ensure that the *number* option-argument is a decimal integer
34852 whose sign affects the location in the file, measured in bytes, to begin the copying:

Sign	Copying Starts
+	Relative to the beginning of the file.
-	Relative to the end of the file.
none	Relative to the end of the file.

34853 The origin for counting shall be 1; that is, **-c** +1 represents the first byte of the file,
34854 **-c** -1 the last.

34855 **-f** If the input file is a regular file or if the *file* operand specifies a FIFO, do not
34856 terminate after the last line of the input file has been copied, but read and copy
34857 further bytes from the input file when they become available. If no *file* operand is
34858 specified and standard input is a pipe, the **-f** option shall be ignored. If the input
34859 file is not a FIFO, pipe, or regular file, it is unspecified whether or not the **-f** option
34860 shall be ignored.

34861 **-n** *number* This option shall be equivalent to **-c** *number*, except the starting location in the file
34862 shall be measured in lines instead of bytes. The origin for counting shall be 1; that
34863 is, **-n** +1 represents the first line of the file, **-n** -1 the last.

34864 If neither **-c** nor **-n** is specified, **-n** 10 shall be assumed.

34865 OPERANDS

34866 The following operand shall be supported:

34867 *file* A pathname of an input file. If no *file* operands are specified, the standard input
34868 shall be used.

34869 STDIN

34870 The standard input shall be used only if no *file* operands are specified. See the INPUT FILES
34871 section.

34876 INPUT FILES

34877 If the **-c** option is specified, the input file can contain arbitrary data; otherwise, the input file
34878 shall be a text file.

34879 ENVIRONMENT VARIABLES

34880 The following environment variables shall affect the execution of *tail*:

34881 **LANG** Provide a default value for the internationalization variables that are unset or null.
34882 (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2,
34883 Internationalization Variables for the precedence of internationalization variables
34884 used to determine the values of locale categories.)

34885 **LC_ALL** If set to a non-empty string value, override the values of all the other
34886 internationalization variables.

34887 **LC_CTYPE** Determine the locale for the interpretation of sequences of bytes of text data as
34888 characters (for example, single-byte as opposed to multi-byte characters in
34889 arguments and input files).

34890 *LC_MESSAGES*

34891 Determine the locale that should be used to affect the format and contents of
34892 diagnostic messages written to standard error.

34893 XSI **NLSPATH** Determine the location of message catalogs for the processing of *LC_MESSAGES*.

34894 ASYNCHRONOUS EVENTS

34895 Default.

34896 STDOUT

34897 The designated portion of the input file shall be written to standard output.

34898 STDERR

34899 The standard error shall be used only for diagnostic messages.

34900 OUTPUT FILES

34901 None.

34902 EXTENDED DESCRIPTION

34903 None.

34904 EXIT STATUS

34905 The following exit values shall be returned:

34906 0 Successful completion.

34907 >0 An error occurred.

34908 CONSEQUENCES OF ERRORS

34909 Default.

34910 APPLICATION USAGE

34911 The **-c** option should be used with caution when the input is a text file containing multi-byte
34912 characters; it may produce output that does not start on a character boundary.

34913 Although the input file to *tail* can be any type, the results might not be what would be expected
34914 on some character special device files or on file types not described by the System Interfaces
34915 volume of IEEE Std 1003.1-2001. Since this volume of IEEE Std 1003.1-2001 does not specify the
34916 block size used when doing input, *tail* need not read all of the data from devices that only
34917 perform block transfers.

34918 EXAMPLES

34919 The **-f** option can be used to monitor the growth of a file that is being written by some other
34920 process. For example, the command:

34921 `tail -f fred`

34922 prints the last ten lines of the file **fred**, followed by any lines that are appended to **fred** between
34923 the time *tail* is initiated and killed. As another example, the command:

34924 `tail -f -c 15 fred`

34925 prints the last 15 bytes of the file **fred**, followed by any bytes that are appended to **fred** between
34926 the time *tail* is initiated and killed.

34927 RATIONALE

34928 This version of *tail* was created to allow conformance to the Utility Syntax Guidelines. The
34929 historical **-b** option was omitted because of the general non-portability of block-sized units of
34930 text. The **-c** option historically meant “characters”, but this volume of IEEE Std 1003.1-2001
34931 indicates that it means “bytes”. This was selected to allow reasonable implementations when
34932 multi-byte characters are possible; it was not named **-b** to avoid confusion with the historical
34933 **-b**.

34934 The origin of counting both lines and bytes is 1, matching all widespread historical
34935 implementations.

34936 The restriction on the internal buffer is a compromise between the historical System V
34937 implementation of 4 096 bytes and the BSD 32 768 bytes.

34938 The **-f** option has been implemented as a loop that sleeps for 1 second and copies any bytes that
34939 are available. This is sufficient, but if more efficient methods of determining when new data are
34940 available are developed, implementations are encouraged to use them.

34941 Historical documentation indicates that *tail* ignores the **-f** option if the input file is a pipe (pipe
34942 and FIFO on systems that support FIFOs). On BSD-based systems, this has been true; on System
34943 V-based systems, this was true when input was taken from standard input, but it did not ignore
34944 the **-f** flag if a FIFO was named as the *file* operand. Since the **-f** option is not useful on pipes and
34945 all historical implementations ignore **-f** if no *file* operand is specified and standard input is a
34946 pipe, this volume of IEEE Std 1003.1-2001 requires this behavior. However, since the **-f** option is
34947 useful on a FIFO, this volume of IEEE Std 1003.1-2001 also requires that if standard input is a
34948 FIFO or a FIFO is named, the **-f** option shall not be ignored. Although historical behavior does
34949 not ignore the **-f** option for other file types, this is unspecified so that implementations are
34950 allowed to ignore the **-f** option if it is known that the file cannot be extended.

34951 This was changed to the current form based on comments noting that **-c** was almost never used
34952 without specifying a number and that there was no need to specify **-l** if **-n number** was given.

34953 FUTURE DIRECTIONS

34954 None.

34955 SEE ALSO

34956 `head`

34957 CHANGE HISTORY

34958 First released in Issue 2.

34959 Issue 6

34960 The obsolescent SYNOPSIS lines and associated text are removed.

34961 The normative text is reworded to avoid use of the term “must” for application requirements.

34962 NAME

34963 talk — talk to another user

34964 SYNOPSIS

34965 UP talk address [terminal]

34966

34967 DESCRIPTION

34968 The *talk* utility is a two-way, screen-oriented communication program.

34969 When first invoked, *talk* shall send a message similar to:

34970 Message from <unspecified string>
34971 talk: connection requested by *your_address*
34972 talk: respond with: talk *your_address*

34973 to the specified *address*. At this point, the recipient of the message can reply by typing:

34974 talk *your_address*

34975 Once communication is established, the two parties can type simultaneously, with their output
34976 displayed in separate regions of the screen. Characters shall be processed as follows:

- 34977 • Typing the alert character shall alert the recipient's terminal.
- 34978 • Typing <control>-L shall cause the sender's screen regions to be refreshed.
- 34979 • Typing the erase and kill characters shall affect the sender's terminal in the manner described
34980 by the **termios** interface in the Base Definitions volume of IEEE Std 1003.1-2001, Chapter 11,
34981 General Terminal Interface.
- 34982 • Typing the interrupt or end-of-file characters shall terminate the local *talk* utility. Once the
34983 session has been terminated on one side, the other side of the *talk* session shall be notified
34984 that the *talk* session has been terminated and shall be able to do nothing except exit.
- 34985 • Typing characters from *LC_CTYPE* classifications **print** or **space** shall cause those characters
34986 to be sent to the recipient's terminal.
- 34987 • When and only when the **stty iexten** local mode is enabled, the existence and processing of
34988 additional special control characters and multi-byte or single-byte functions shall be
34989 implementation-defined.
- 34990 • Typing other non-printable characters shall cause implementation-defined sequences of
34991 printable characters to be sent to the recipient's terminal.

34992 Permission to be a recipient of a *talk* message can be denied or granted by use of the *mesg* utility.
34993 However, a user's privilege may further constrain the domain of accessibility of other users'
34994 terminals. The *talk* utility shall fail when the user lacks the appropriate privileges to perform the
34995 requested action.

34996 Certain block-mode terminals do not have all the capabilities necessary to support the
34997 simultaneous exchange of messages required for *talk*. When this type of exchange cannot be
34998 supported on such terminals, the implementation may support an exchange with reduced levels
34999 of simultaneous interaction or it may report an error describing the terminal-related deficiency.

35000 OPTIONS

35001 None.

35002 OPERANDS

35003 The following operands shall be supported:

35004 **address** The recipient of the *talk* session. One form of *address* is the <*user name*>, as returned by the *who* utility. Other address formats and how they are handled are unspecified.

35007 **terminal** If the recipient is logged in more than once, the *terminal* argument can be used to indicate the appropriate terminal name. If *terminal* is not specified, the *talk* message shall be displayed on one or more accessible terminals in use by the recipient. The format of *terminal* shall be the same as that returned by the *who* utility.

35011 STDIN

35012 Characters read from standard input shall be copied to the recipient's terminal in an unspecified manner. If standard input is not a terminal, *talk* shall write a diagnostic message and exit with a non-zero status.

35015 INPUT FILES

35016 None.

35017 ENVIRONMENT VARIABLES

35018 The following environment variables shall affect the execution of *talk*:

35019 **LANG** Provide a default value for the internationalization variables that are unset or null. (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2, Internationalization Variables for the precedence of internationalization variables used to determine the values of locale categories.)

35023 **LC_ALL** If set to a non-empty string value, override the values of all the other internationalization variables.

35025 **LC_CTYPE** Determine the locale for the interpretation of sequences of bytes of text data as characters (for example, single-byte as opposed to multi-byte characters in arguments and input files). If the recipient's locale does not use an *LC_CTYPE* equivalent to the sender's, the results are undefined.

35029 LC_MESSAGES

35030 Determine the locale that should be used to affect the format and contents of diagnostic messages written to standard error and informative messages written to standard output.

35033 XSI **NLSPATH** Determine the location of message catalogs for the processing of *LC_MESSAGES*.

35034 **TERM** Determine the name of the invoker's terminal type. If this variable is unset or null, an unspecified default terminal type shall be used.

35036 ASYNCHRONOUS EVENTS

35037 When the *talk* utility receives a SIGINT signal, the utility shall terminate and exit with a zero status. It shall take the standard action for all other signals.

35039 STDOUT

35040 If standard output is a terminal, characters copied from the recipient's standard input may be written to standard output. Standard output also may be used for diagnostic messages. If standard output is not a terminal, *talk* shall exit with a non-zero status.

35043 STDERR

35044 None.

35045 OUTPUT FILES

35046 None.

35047 EXTENDED DESCRIPTION

35048 None.

35049 EXIT STATUS

35050 The following exit values shall be returned:

35051 0 Successful completion.

35052 >0 An error occurred or *talk* was invoked on a terminal incapable of supporting it.

35053 CONSEQUENCES OF ERRORS

35054 Default.

35055 APPLICATION USAGE

35056 Because the handling of non-printable, non-<space>s is tied to the *stty* description of **iexten**,
35057 implementation extensions within the terminal driver can be accessed. For example, some
35058 implementations provide line editing functions with certain control character sequences.

35059 EXAMPLES

35060 None.

35061 RATIONALE

35062 The *write* utility was included in this volume of IEEE Std 1003.1-2001 since it can be
35063 implemented on all terminal types. The *talk* utility, which cannot be implemented on certain
35064 terminals, was considered to be a “better” communications interface. Both of these programs are
35065 in widespread use on historical implementations. Therefore, both utilities have been specified.

35066 All references to networking abilities (*talking* to a user on another system) were removed as
35067 being outside the scope of this volume of IEEE Std 1003.1-2001.

35068 Historical BSD and System V versions of *talk* terminate both of the conversations when either
35069 user breaks out of the session. This can lead to adverse consequences if a user unwittingly
35070 continues to enter text that is interpreted by the shell when the other terminates the session.
35071 Therefore, the version of *talk* specified by this volume of IEEE Std 1003.1-2001 requires both
35072 users to terminate their end of the session explicitly.

35073 Only messages sent to the terminal of the invoking user can be internationalized in any way:

- 35074 • The original “Message from *<unspecified string>* ...” message sent to the terminal of the
35075 recipient cannot be internationalized because the environment of the recipient is as yet
35076 inaccessible to the *talk* utility. The environment of the invoking party is irrelevant.
- 35077 • Subsequent communication between the two parties cannot be internationalized because the
35078 two parties may specify different languages in their environment (and non-portable
35079 characters cannot be mapped from one language to another).
- 35080 • Neither party can be required to communicate in a language other than C and/or the one
35081 specified by their environment because unavailable terminal hardware support (for example,
35082 fonts) may be required.

35083 The text in the STDOOUT section reflects the usage of the verb “display” in this section; some *talk*
35084 implementations actually use standard output to write to the terminal, but this volume of
35085 IEEE Std 1003.1-2001 does not require that to be the case.

35086 The format of the terminal name is unspecified, but the descriptions of *ps*, *talk*, *who*, and *write*
35087 require that they all use or accept the same format.

35088 The handling of non-printable characters is partially implementation-defined because the details
35089 of mapping them to printable sequences is not needed by the user. Historical implementations,
35090 for security reasons, disallow the transmission of non-printable characters that may send
35091 commands to the other terminal.

35092 FUTURE DIRECTIONS

35093 None.

35094 SEE ALSO

35095 *mesg*, *stty*, *who*, *write*, the Base Definitions volume of IEEE Std 1003.1-2001, Chapter 11, General
35096 Terminal Interface

35097 CHANGE HISTORY

35098 First released in Issue 4.

35099 Issue 6

35100 This utility is marked as part of the User Portability Utilities option.

35101 NAME

35102 tee — duplicate standard input

35103 SYNOPSIS

35104 tee [-ai][*file*...]

35105 DESCRIPTION

35106 The *tee* utility shall copy standard input to standard output, making a copy in zero or more files.

35107 The *tee* utility shall not buffer output.

35108 If the **-a** option is not specified, output files shall be written (see Section 1.7.1.4 (on page 4)).

35109 OPTIONS

35110 The *tee* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section 12.2, Utility Syntax Guidelines.

35112 The following options shall be supported:

35113 **-a** Append the output to the files.

35114 **-i** Ignore the SIGINT signal.

35115 OPERANDS

35116 The following operands shall be supported:

35117 *file* A pathname of an output file. Processing of at least 13 *file* operands shall be supported.

35119 STDIN

35120 The standard input can be of any type.

35121 INPUT FILES

35122 None.

35123 ENVIRONMENT VARIABLES

35124 The following environment variables shall affect the execution of *tee*:

35125 **LANG** Provide a default value for the internationalization variables that are unset or null. (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2, Internationalization Variables for the precedence of internationalization variables used to determine the values of locale categories.)

35129 **LC_ALL** If set to a non-empty string value, override the values of all the other internationalization variables.

35131 **LC_CTYPE** Determine the locale for the interpretation of sequences of bytes of text data as characters (for example, single-byte as opposed to multi-byte characters in arguments).

35134 LC_MESSAGES

35135 Determine the locale that should be used to affect the format and contents of diagnostic messages written to standard error.

35137 XSI **NLSPATH** Determine the location of message catalogs for the processing of *LC_MESSAGES*.

35138 ASYNCHRONOUS EVENTS

35139 Default, except that if the **-i** option was specified, SIGINT shall be ignored.

35140 STDOUT

35141 The standard output shall be a copy of the standard input.

35142 STDERR

35143 The standard error shall be used only for diagnostic messages.

35144 OUTPUT FILES

35145 If any *file* operands are specified, the standard input shall be copied to each named file.

35146 EXTENDED DESCRIPTION

35147 None.

35148 EXIT STATUS

35149 The following exit values shall be returned:

35150 0 The standard input was successfully copied to all output files.

35151 >0 An error occurred.

35152 CONSEQUENCES OF ERRORS

35153 If a write to any successfully opened *file* operand fails, writes to other successfully opened *file* operands and standard output shall continue, but the exit status shall be non-zero. Otherwise, the default actions specified in Section 1.11 (on page 20) apply.

35156 APPLICATION USAGE

35157 The *tee* utility is usually used in a pipeline, to make a copy of the output of some utility.

35158 The *file* operand is technically optional, but *tee* is no more useful than *cat* when none is specified.

35159 EXAMPLES

35160 Save an unsorted intermediate form of the data in a pipeline:

35161 ... | tee unsorted | sort > sorted

35162 RATIONALE

35163 The buffering requirement means that *tee* is not allowed to use ISO C standard fully buffered or line-buffered writes. It does not mean that *tee* has to do 1-byte reads followed by 1-byte writes.

35165 It should be noted that early versions of BSD ignore any invalid options and accept a single '-' as an alternative to -i. They also print a message if unable to open a file:

35167 "tee: cannot access %s\n", <pathname>

35168 Historical implementations ignore write errors. This is explicitly not permitted by this volume of IEEE Std 1003.1-2001.

35170 Some historical implementations use O_APPEND when providing append mode; others use the lseek() function to seek to the end-of-file after opening the file without O_APPEND. This volume of IEEE Std 1003.1-2001 requires functionality equivalent to using O_APPEND; see Section 1.7.1.4 (on page 4).

35174 FUTURE DIRECTIONS

35175 None.

35176 SEE ALSO

35177 Chapter 1 (on page 1), *cat*, the System Interfaces volume of IEEE Std 1003.1-2001, *lseek()*

35178 CHANGE HISTORY

35179 First released in Issue 2.

35180 **Issue 6**

35181 IEEE PASC Interpretation 1003.2 #168 is applied.

35182 NAME

35183 test — evaluate expression

35184 SYNOPSIS

35185 test [*expression*]35186 [[*expression*]]

35187 DESCRIPTION

35188 The *test* utility shall evaluate the *expression* and indicate the result of the evaluation by its exit status. An exit status of zero indicates that the expression evaluated as true and an exit status of 1 indicates that the expression evaluated as false.

35191 In the second form of the utility, which uses "[]" rather than *test*, the application shall ensure
35192 that the square brackets are separate arguments.

35193 OPTIONS

35194 The *test* utility shall not recognize the "--" argument in the manner specified by guideline 10 in
35195 the Base Definitions volume of IEEE Std 1003.1-2001, Section 12.2, Utility Syntax Guidelines.

35196 No options shall be supported.

35197 OPERANDS

35198 The application shall ensure that all operators and elements of primaries are presented as
35199 separate arguments to the *test* utility.

35200 The following primaries can be used to construct *expression*:

35201	-b <i>pathname</i>	True if <i>pathname</i> resolves to a file that exists and is a block special file. False if <i>pathname</i> cannot be resolved, or if <i>pathname</i> resolves to a file that exists but is not a block special file.	2
35204	-c <i>pathname</i>	True if <i>pathname</i> resolves to a file that exists and is a character special file. False if <i>pathname</i> cannot be resolved, or if <i>pathname</i> resolves to a file that exists but is not a character special file.	2
35207	-d <i>pathname</i>	True if <i>pathname</i> resolves to a file that exists and is a directory. False if <i>pathname</i> cannot be resolved, or if <i>pathname</i> resolves to a file that exists but is not a directory.	2
35209	-e <i>pathname</i>	True if <i>pathname</i> resolves to a file that exists. False if <i>pathname</i> cannot be resolved.	2
35210	-f <i>pathname</i>	True if <i>pathname</i> resolves to a file that exists and is a regular file. False if <i>pathname</i> cannot be resolved, or if <i>pathname</i> resolves to a file that exists but is not a regular file.	2
35213	-g <i>pathname</i>	True if <i>pathname</i> resolves to a file that exists and has its set-group-ID flag set. False if <i>pathname</i> cannot be resolved, or if <i>pathname</i> resolves to a file that exists but does not have its set-group-ID flag set.	2
35216	-h <i>pathname</i>	True if <i>pathname</i> resolves to a file that exists and is a symbolic link. False if <i>pathname</i> cannot be resolved, or if <i>pathname</i> resolves to a file that exists but is not a symbolic link. If the final component of <i>pathname</i> is a symlink, that symlink is not followed.	2
35220	-L <i>pathname</i>	True if <i>pathname</i> resolves to a file that exists and is a symbolic link. False if <i>pathname</i> cannot be resolved, or if <i>pathname</i> resolves to a file that exists but is not a symbolic link. If the final component of <i>pathname</i> is a symlink, that symlink is not followed.	2
35224	-n <i>string</i>	True if the length of <i>string</i> is non-zero; otherwise, false.	2

35225	-p pathname	True if <i>pathname</i> resolves to a file that exists and is a FIFO. False if <i>pathname</i> cannot be resolved, or if <i>pathname</i> resolves to a file that exists but is not a FIFO.	2
35226			2
35227	-r pathname	True if <i>pathname</i> resolves to a file that exists and for which permission to read from the file will be granted, as defined in Section 1.7.1.4 (on page 4). False if <i>pathname</i> cannot be resolved, or if <i>pathname</i> resolves to a file for which permission to read from the file will not be granted.	2
35228			2
35229			2
35230			2
35231	-S pathname	True if <i>pathname</i> resolves to a file that exists and is a socket. False if <i>pathname</i> cannot be resolved, or if <i>pathname</i> resolves to a file that exists but is not a socket.	2
35232			2
35233	-s pathname	True if <i>pathname</i> resolves to a file that exists and has a size greater than zero. False if <i>pathname</i> cannot be resolved, or if <i>pathname</i> resolves to a file that exists but does not have a size greater than zero.	2
35234			2
35235			2
35236	-t file_descriptor	True if file descriptor number <i>file_descriptor</i> is open and is associated with a terminal. False if <i>file_descriptor</i> is not a valid file descriptor number, or if file descriptor number <i>file_descriptor</i> is not open, or if it is open but is not associated with a terminal.	2
35237			2
35238			2
35239			2
35240			2
35241	-u pathname	True if <i>pathname</i> resolves to a file that exists and has its set-user-ID flag set. False if <i>pathname</i> cannot be resolved, or if <i>pathname</i> resolves to a file that exists but does not have its set-user-ID flag set.	2
35242			2
35243			2
35244	-w pathname	True if <i>pathname</i> resolves to a file that exists and for which permission to write to the file will be granted, as defined in Section 1.7.1.4 (on page 4). False if <i>pathname</i> cannot be resolved, or if <i>pathname</i> resolves to a file for which permission to write to the file will not be granted.	2
35245			2
35246			2
35247			2
35248	-x pathname	True if <i>pathname</i> resolves to a file that exists and for which permission to execute the file (or search it, if it is a directory) will be granted, as defined in Section 1.7.1.4 (on page 4). False if <i>pathname</i> cannot be resolved, or if <i>pathname</i> resolves to a file for which permission to execute (or search) the file will not be granted.	2
35249			2
35250			2
35251			2
35252	-z string	True if the length of string <i>string</i> is zero; otherwise, false.	2
35253	<i>string</i>	True if the string <i>string</i> is not the null string; otherwise, false.	2
35254	<i>s1 = s2</i>	True if the strings <i>s1</i> and <i>s2</i> are identical; otherwise, false.	2
35255	<i>s1 != s2</i>	True if the strings <i>s1</i> and <i>s2</i> are not identical; otherwise, false.	2
35256	<i>n1 -eq n2</i>	True if the integers <i>n1</i> and <i>n2</i> are algebraically equal; otherwise, false.	2
35257	<i>n1 -ne n2</i>	True if the integers <i>n1</i> and <i>n2</i> are not algebraically equal; otherwise, false.	2
35258	<i>n1 -gt n2</i>	True if the integer <i>n1</i> is algebraically greater than the integer <i>n2</i> ; otherwise, false.	2
35259	<i>n1 -ge n2</i>	True if the integer <i>n1</i> is algebraically greater than or equal to the integer <i>n2</i> ; otherwise, false.	2
35260			2
35261	<i>n1 -lt n2</i>	True if the integer <i>n1</i> is algebraically less than the integer <i>n2</i> ; otherwise, false.	2
35262	<i>n1 -le n2</i>	True if the integer <i>n1</i> is algebraically less than or equal to the integer <i>n2</i> ; otherwise, false.	2
35263			2
35264 XSI	expression1 -a expression2	True if both <i>expression1</i> and <i>expression2</i> are true; otherwise, false. The -a binary primary is left associative. It has a higher precedence than -o .	2
35265			2
35266			2

35267 XSI	expression1 -o expression2	
35268	True if either <i>expression1</i> or <i>expression2</i> is true; otherwise, false. The -o binary primary is left associative.	2
35269		
35270	With the exception of the -h file and -L file primaries, if a <i>file</i> argument is a symbolic link, <i>test</i> shall evaluate the expression by resolving the symbolic link and using the file referenced by the link.	
35271		
35272		
35273	These primaries can be combined with the following operators:	
35274	! expression True if <i>expression</i> is false. False if <i>expression</i> is true.	2
35275 XSI	(expression) True if <i>expression</i> is true. False if <i>expression</i> is false. The parentheses can be used to alter the normal precedence and associativity.	2
35276		2
35277	The primaries with two elements of the form:	
35278	<i>-primary_operator primary_operand</i>	
35279	are known as <i>unary primaries</i> . The primaries with three elements in either of the two forms:	
35280	<i>primary_operand -primary_operator primary_operand</i>	
35281	<i>primary_operand primary_operator primary_operand</i>	
35282	are known as <i>binary primaries</i> . Additional implementation-defined operators and <i>primary_operators</i> may be provided by implementations. They shall be of the form -operator where the first character of <i>operator</i> is not a digit.	
35283		
35284		
35285	The algorithm for determining the precedence of the operators and the return value that shall be generated is based on the number of arguments presented to <i>test</i> . (However, when using the "[. . .]" form, the right-bracket final argument shall not be counted in this algorithm.)	
35286		
35287		
35288	In the following list, \$1, \$2, \$3, and \$4 represent the arguments presented to <i>test</i> :	
35289	0 arguments: Exit false (1).	
35290	1 argument: Exit true (0) if \$1 is not null; otherwise, exit false.	
35291	2 arguments:	
35292	• If \$1 is '!', exit true if \$2 is null, false if \$2 is not null.	
35293	• If \$1 is a unary primary, exit true if the unary test is true, false if the unary test is false.	
35294	• Otherwise, produce unspecified results.	
35295	3 arguments:	
35296	• If \$2 is a binary primary, perform the binary test of \$1 and \$3.	
35297 XSI	• If \$1 is '()' and \$3 is ')', perform the unary test of \$2.	1
35298	• Otherwise, produce unspecified results.	
35299	4 arguments:	
35300 XSI	• If \$1 is '!', negate the three-argument test of \$2, \$3, and \$4.	
35301	• If \$1 is '()' and \$4 is ')', perform the two-argument test of \$2 and \$3.	
35302	• Otherwise, the results are unspecified.	
35303 XSI	>4 arguments: The results are unspecified.	
35304		
35305	On XSI-conformant systems, combinations of primaries and operators shall be evaluated using the precedence and associativity rules described previously. In addition, the string comparison binary primaries '=' and "!=" shall have	

35306 a higher precedence than any unary primary.

35307 **STDIN**

35308 Not used.

35309 **INPUT FILES**

35310 None.

35311 **ENVIRONMENT VARIABLES**

35312 The following environment variables shall affect the execution of *test*:

35313 *LANG* Provide a default value for the internationalization variables that are unset or null.
(See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2, Internationalization Variables for the precedence of internationalization variables used to determine the values of locale categories.)

35317 *LC_ALL* If set to a non-empty string value, override the values of all the other internationalization variables.

35319 *LC_CTYPE* Determine the locale for the interpretation of sequences of bytes of text data as characters (for example, single-byte as opposed to multi-byte characters in arguments).

35322 *LC_MESSAGES* Determine the locale that should be used to affect the format and contents of diagnostic messages written to standard error.

35325 XSI *NLSPATH* Determine the location of message catalogs for the processing of *LC_MESSAGES*.

35326 **ASYNCHRONOUS EVENTS**

35327 Default.

35328 **STDOUT**

35329 Not used.

35330 **STDERR**

35331 The standard error shall be used only for diagnostic messages.

35332 **OUTPUT FILES**

35333 None.

35334 **EXTENDED DESCRIPTION**

35335 None.

35336 **EXIT STATUS**

35337 The following exit values shall be returned:

35338 0 *expression* evaluated to true.

35339 1 *expression* evaluated to false or *expression* was missing.

35340 >1 An error occurred.

35341 **CONSEQUENCES OF ERRORS**

35342 Default.

35343 **APPLICATION USAGE**

35344 Scripts should be careful when dealing with user-supplied input that could be confused with
35345 primaries and operators. Unless the application writer knows all the cases that produce input to
35346 the script, invocations like:

35347 test "\$1" -a "\$2"

35348 should be written as:

35349 test "\$1" && test "\$2"

35350 to avoid problems if a user supplied values such as \$1 set to '!' and \$2 set to the null string.
35351 That is, in cases where maximal portability is of concern, replace:

35352 test expr1 -a expr2

35353 with:

35354 test expr1 && test expr2

35355 and replace:

35356 test expr1 -o expr2

35357 with:

35358 test expr1 || test expr2

35359 but note that, in *test*, -a has higher precedence than -o while "&&" and "||" have equal
35360 precedence in the shell.

35361 Parentheses or braces can be used in the shell command language to effect grouping.

35362 Parentheses must be escaped when using *sh*; for example:

35363 test \(\ expr1 -a expr2 \) -o expr3

35364 This command is not always portable outside XSI-conformant systems. The following form can
35365 be used instead:

35366 (test expr1 && test expr2) || test expr3

35367 The two commands:

35368 test "\$1"

35369 test ! "\$1"

35370 could not be used reliably on some historical systems. Unexpected results would occur if such a
35371 *string* expression were used and \$1 expanded to '!', '(', or a known unary primary. Better
35372 constructs are:

35373 test -n "\$1"

35374 test -z "\$1"

35375 respectively.

35376 Historical systems have also been unreliable given the common construct:

35377 test "\$response" = "expected string"

35378 One of the following is a more reliable form:

35379 test "X\$response" = "Xexpected string"

35380 test "expected string" = "\$response"

35381 Note that the second form assumes that *expected string* could not be confused with any unary
35382 primary. If *expected string* starts with '`-`', '`(`', '`!`', or even '`=`', the first form should be used
35383 instead. Using the preceding rules without the XSI marked extensions, any of the three
35384 comparison forms is reliable, given any input. (However, note that the strings are quoted in all
35385 cases.)

35386 Because the string comparison binary primaries, '`=`' and "`!=`", have a higher precedence than
35387 any unary primary in the greater than 4 argument case, unexpected results can occur if
35388 arguments are not properly prepared. For example, in:

35389 `test -d $1 -o -d $2`

35390 If `$1` evaluates to a possible directory name of '`=`', the first three arguments are considered a
35391 string comparison, which shall cause a syntax error when the second `-d` is encountered. One of
35392 the following forms prevents this; the second is preferred:

35393 `test \(\ -d "$1" \) -o \(\ -d "$2" \)`
35394 `test -d "$1" || test -d "$2"`

35395 Also in the greater than 4 argument case:

35396 `test "$1" = "bat" -a "$2" = "ball"`

35397 syntax errors occur if `$1` evaluates to '`(`' or '`!`'. One of the following forms prevents this; the
35398 third is preferred:

35399 `test "X$1" = "Xbat" -a "X$2" = "Xball"`
35400 `test "$1" = "bat" && test "$2" = "ball"`
35401 `test "X$1" = "Xbat" && test "X$2" = "Xball"`

35402 EXAMPLES

1. Exit if there are not two or three arguments (two variations):

35404 `if [$# -ne 2 -a $# -ne 3]; then exit 1; fi`
35405 `if [$# -lt 2 -o $# -gt 3]; then exit 1; fi`

2. Perform a `mkdir` if a directory does not exist:

35407 `test ! -d tempdir && mkdir tempdir`

3. Wait for a file to become non-readable:

35409 `while test -r thefile`
35410 `do`
35411 `sleep 30`
35412 `done`
35413 `echo '"thefile" is no longer readable'`

4. Perform a command if the argument is one of three strings (two variations):

35415 `if ["$1" = "pear"] || ["$1" = "grape"] || ["$1" = "apple"]`
35416 `then`
35417 `command`
35418 `fi`
35419 `case "$1" in`
35420 `pear|grape|apple) command ;;`
35421 `esac`

35422 RATIONALE

35423 The KornShell-derived conditional command (double bracket [[]]) was removed from the shell
35424 command language description in an early proposal. Objections were raised that the real
35425 problem is misuse of the *test* command (!), and putting it into the shell is the wrong way to fix
35426 the problem. Instead, proper documentation and a new shell reserved word (!) are sufficient.

35427 Tests that require multiple *test* operations can be done at the shell level using individual
35428 invocations of the *test* command and shell logicals, rather than using the error-prone –o flag of
35429 *test*.

35430 XSI-conformant systems support more than four arguments.

35431 XSI-conformant systems support the combining of primaries with the following constructs:

35432 *expression1* –a *expression2*

35433 True if both *expression1* and *expression2* are true.

35434 *expression1* –o *expression2*

35435 True if at least one of *expression1* and *expression2* are true.

35436 (*expression*)

35437 True if *expression* is true.

35438 In evaluating these more complex combined expressions, the following precedence rules are
35439 used:

- 35440 • The unary primaries have higher precedence than the algebraic binary primaries.
- 35441 • The unary primaries have lower precedence than the string binary primaries.
- 35442 • The unary and binary primaries have higher precedence than the unary *string* primary.
- 35443 • The ! operator has higher precedence than the –a operator, and the –a operator has higher
35444 precedence than the –o operator.
- 35445 • The –a and –o operators are left associative.
- 35446 • The parentheses can be used to alter the normal precedence and associativity.

35447 The BSD and System V versions of –f are not the same. The BSD definition was:

35448 –f *file* True if *file* exists and is not a directory.

35449 The SVID version (true if the file exists and is a regular file) was chosen for this volume of
35450 IEEE Std 1003.1-2001 because its use is consistent with the –b, –c, –d, and –p operands (*file* exists
35451 and is a specific file type).

35452 The –e primary, possessing similar functionality to that provided by the C shell, was added
35453 because it provides the only way for a shell script to find out if a file exists without trying to
35454 open the file. Since implementations are allowed to add additional file types, a portable script
35455 cannot use:

35456 test –b foo –o –c foo –o –d foo –o –f foo –o –p foo

35457 to find out if **foo** is an existing file. On historical BSD systems, the existence of a file could be
35458 determined by:

35459 test –f foo –o –d foo

35460 but there was no easy way to determine that an existing file was a regular file. An early proposal
35461 used the KornShell –a primary (with the same meaning), but this was changed to –e because
35462 there were concerns about the high probability of humans confusing the –a primary with the –a
35463 binary operator.

35464 The following options were not included in this volume of IEEE Std 1003.1-2001, although they
35465 are provided by some implementations. These operands should not be used by new
35466 implementations for other purposes:

- 35467 **-k file** True if *file* exists and its sticky bit is set.
35468 **-C file** True if *file* is a contiguous file.
35469 **-V file** True if *file* is a version file.

35470 The following option was not included because it was undocumented in most implementations,
35471 has been removed from some implementations (including System V), and the functionality is
35472 provided by the shell (see Section 2.6.2 (on page 37)).

- 35473 **-l string** The length of the string *string*.

35474 The **-b**, **-c**, **-g**, **-p**, **-u**, and **-x** operands are derived from the SVID; historical BSD does not
35475 provide them. The **-k** operand is derived from System V; historical BSD does not provide it.

35476 On historical BSD systems, *test -w directory* always returned false because *test* tried to open the
35477 directory for writing, which always fails.

35478 Some additional primaries newly invented or from the KornShell appeared in an early proposal
35479 as part of the conditional command ([[]]): *s1 > s2*, *s1 < s2*, *str = pattern*, *str != pattern*, *f1 -nt f2*, *f1*
35480 **-ot f2**, and *f1 -ef f2*. They were not carried forward into the *test* utility when the conditional
35481 command was removed from the shell because they have not been included in the *test* utility
35482 built into historical implementations of the *sh* utility.

35483 The **-t file_descriptor** primary is shown with a mandatory argument because the grammar is
35484 ambiguous if it can be omitted. Historical implementations have allowed it to be omitted,
35485 providing a default of 1.

35486 FUTURE DIRECTIONS

35487 None.

35488 SEE ALSO

35489 Section 1.7.1.4 (on page 4), *find*

35490 CHANGE HISTORY

35491 First released in Issue 2.

35492 Issue 5

35493 The FUTURE DIRECTIONS section is added.

35494 Issue 6

35495 The **-h** operand is added for symbolic links, and access permission requirements are clarified for
35496 the **-r**, **-w**, and **-x** operands to align with the IEEE P1003.2b draft standard.

35497 The normative text is reworded to avoid use of the term “must” for application requirements.

35498 The **-L** and **-S** operands are added for symbolic links and sockets.

35499 IEEE Std 1003.1-2001/Cor 1-2002, item XCU/TC1/D6/38 is applied, adding XSI margin marking 1
35500 and shading to a line in the OPERANDS section referring to the use of parentheses as arguments 1
35501 to the *test* utility. 1

35502 IEEE Std 1003.1-2001/Cor 2-2004, item XCU/TC2/D6/30 is applied, rewording the existence 2
35503 primaries for the *test* utility. 2

35504 NAME

35505 time — time a simple command

35506 SYNOPSIS

35507 UP time [-p] utility [argument...]

35508

35509 DESCRIPTION

35510 The *time* utility shall invoke the utility named by the *utility* operand with arguments supplied as
35511 the *argument* operands and write a message to standard error that lists timing statistics for the
35512 utility. The message shall include the following information:

- 35513 • The elapsed (real) time between invocation of *utility* and its termination.
- 35514 • The User CPU time, equivalent to the sum of the *tms_utime* and *tms_cutime* fields returned by
35515 the *times()* function defined in the System Interfaces volume of IEEE Std 1003.1-2001 for the
35516 process in which *utility* is executed.
- 35517 • The System CPU time, equivalent to the sum of the *tms_stime* and *tms_cstime* fields returned
35518 by the *times()* function for the process in which *utility* is executed.

35519 The precision of the timing shall be no less than the granularity defined for the size of the clock
35520 tick unit on the system, but the results shall be reported in terms of standard time units (for
35521 example, 0.02 seconds, 00:00:00.02, 1m33.75s, 365.21 seconds), not numbers of clock ticks.

35522 When *time* is used as part of a pipeline, the times reported are unspecified, except when it is the
35523 sole command within a grouping command (see Section 2.9.4.1 (on page 52)) in that pipeline. For
35524 example, the commands on the left are unspecified; those on the right report on utilities **a** and **c**,
35525 respectively:

35526 time a | b | c { time a } | b | c
35527 a | b | time c a | b | (time c)

35528 OPTIONS

35529 The *time* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section
35530 12.2, Utility Syntax Guidelines.

35531 The following option shall be supported:

35532 **-p** Write the timing output to standard error in the format shown in the STDERR
35533 section.

35534 OPERANDS

35535 The following operands shall be supported:

35536 *utility* The name of a utility that is to be invoked. If the *utility* operand names any of the
35537 special built-in utilities in Section 2.14 (on page 64), the results are undefined.

35538 *argument* Any string to be supplied as an argument when invoking the utility named by the
35539 *utility* operand.

35540 STDIN

35541 Not used.

35542 INPUT FILES

35543 None.

35544 ENVIRONMENT VARIABLES

35545 The following environment variables shall affect the execution of *time*:

35546 **LANG** Provide a default value for the internationalization variables that are unset or null.
(See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2, Internationalization Variables for the precedence of internationalization variables used to determine the values of locale categories.)

35550 **LC_ALL** If set to a non-empty string value, override the values of all the other internationalization variables.

35552 **LC_CTYPE** Determine the locale for the interpretation of sequences of bytes of text data as characters (for example, single-byte as opposed to multi-byte characters in arguments).

35555 *LC_MESSAGES*

35556 Determine the locale that should be used to affect the format and contents of
35557 diagnostic and informative messages written to standard error.

35558 *LC_NUMERIC*

35559 Determine the locale for numeric formatting.

35560 XSI **NLSPATH** Determine the location of message catalogs for the processing of *LC_MESSAGES*.

35561 **PATH** Determine the search path that shall be used to locate the utility to be invoked; see
35562 the Base Definitions volume of IEEE Std 1003.1-2001, Chapter 8, Environment
35563 Variables.

35564 ASYNCHRONOUS EVENTS

35565 Default.

35566 *STDOUT*

35567 Not used.

35568 *STDERR*

35569 The standard error shall be used to write the timing statistics. If **-p** is specified, the following
35570 format shall be used in the POSIX locale:

35571 "real %f\nuser %f\nsys %f\n", <real seconds>, <user seconds>,
35572 <system seconds>

35573 where each floating-point number shall be expressed in seconds. The precision used may be less
35574 than the default six digits of %f, but shall be sufficiently precise to accommodate the size of the
35575 clock tick on the system (for example, if there were 60 clock ticks per second, at least two digits
35576 shall follow the radix character). The number of digits following the radix character shall be no
35577 less than one, even if this always results in a trailing zero. The implementation may append
35578 white space and additional information following the format shown here.

35579 *OUTPUT FILES*

35580 None.

35581 EXTENDED DESCRIPTION

35582 None.

35583 EXIT STATUS

35584 If the *utility* utility is invoked, the exit status of *time* shall be the exit status of *utility*; otherwise,
35585 the *time* utility shall exit with one of the following values:

35586 1-125 An error occurred in the *time* utility.

- 35587 126 The utility specified by *utility* was found but could not be invoked.
35588 127 The utility specified by *utility* could not be found.

35589 CONSEQUENCES OF ERRORS

35590 Default.

35591 APPLICATION USAGE

35592 The *command*, *env*, *nice*, *nohup*, *time*, and *xargs* utilities have been specified to use exit code 127 if
35593 an error occurs so that applications can distinguish “failure to find a utility” from “invoked
35594 utility exited with an error indication”. The value 127 was chosen because it is not commonly
35595 used for other meanings; most utilities use small values for “normal error conditions” and the
35596 values above 128 can be confused with termination due to receipt of a signal. The value 126 was
35597 chosen in a similar manner to indicate that the utility could be found, but not invoked. Some
35598 scripts produce meaningful error messages differentiating the 126 and 127 cases. The distinction
35599 between exit codes 126 and 127 is based on KornShell practice that uses 127 when all attempts to
35600 *exec* the utility fail with [ENOENT], and uses 126 when any attempt to *exec* the utility fails for
35601 any other reason.

35602 EXAMPLES

35603 It is frequently desirable to apply *time* to pipelines or lists of commands. This can be done by
35604 placing pipelines and command lists in a single file; this file can then be invoked as a utility, and
35605 the *time* applies to everything in the file.

35606 Alternatively, the following command can be used to apply *time* to a complex command:

35607 `time sh -c 'complex-command-line'`

35608 RATIONALE

35609 When the *time* utility was originally proposed to be included in the ISO POSIX-2:1993 standard,
35610 questions were raised about its suitability for inclusion on the grounds that it was not useful for
35611 conforming applications, specifically:

- The underlying CPU definitions from the System Interfaces volume of IEEE Std 1003.1-2001
35613 are vague, so the numeric output could not be compared accurately between systems or even
35614 between invocations.
- The creation of portable benchmark programs was outside the scope this volume of
35616 IEEE Std 1003.1-2001.

35617 However, *time* does fit in the scope of user portability. Human judgement can be applied to the
35618 analysis of the output, and it could be very useful in hands-on debugging of applications or in
35619 providing subjective measures of system performance. Hence it has been included in this
35620 volume of IEEE Std 1003.1-2001.

35621 The default output format has been left unspecified because historical implementations differ
35622 greatly in their style of depicting this numeric output. The **-p** option was invented to provide
35623 scripts with a common means of obtaining this information.

35624 In the KornShell, *time* is a shell reserved word that can be used to time an entire pipeline, rather
35625 than just a simple command. The POSIX definition has been worded to allow this
35626 implementation. Consideration was given to invalidating this approach because of the historical
35627 model from the C shell and System V shell. However, since the System V *time* utility historically
35628 has not produced accurate results in pipeline timing (because the constituent processes are not
35629 all owned by the same parent process, as allowed by POSIX), it did not seem worthwhile to
35630 break historical KornShell usage.

35631 The term *utility* is used, rather than *command*, to highlight the fact that shell compound
35632 commands, pipelines, special built-ins, and so on, cannot be used directly. However, *utility*

35633 includes user application programs and shell scripts, not just the standard utilities.

35634 FUTURE DIRECTIONS

35635 None.

35636 SEE ALSO

35637 Chapter 2 (on page 29), *sh*, the System Interfaces volume of IEEE Std 1003.1-2001, *times()*

35638 CHANGE HISTORY

35639 First released in Issue 2.

35640 Issue 6

35641 This utility is marked as part of the User Portability Utilities option.

35642 NAME

35643 touch — change file access and modification times

35644 SYNOPSIS

35645 touch [-acm][-r ref_file| -t time] file...

35646 DESCRIPTION

35647 The *touch* utility shall change the modification times, access times, or both of files. The
35648 modification time shall be equivalent to the value of the *st_mtime* member of the **stat** structure
35649 for a file, as described in the System Interfaces volume of IEEE Std 1003.1-2001; the access time
35650 shall be equivalent to the value of *st_atime*.

35651 The time used can be specified by the **-t time** option-argument, the corresponding time fields of
35652 the file referenced by the **-r ref_file** option-argument, or the *date_time* operand, as specified in the
35653 following sections. If none of these are specified, *touch* shall use the current time (the value
35654 returned by the equivalent of the *time()* function defined in the System Interfaces volume of
35655 IEEE Std 1003.1-2001).

35656 For each *file* operand, *touch* shall perform actions equivalent to the following functions defined
35657 in the System Interfaces volume of IEEE Std 1003.1-2001:

- 35658 1. If *file* does not exist, a *creat()* function call is made with the *file* operand used as the *path*
35659 argument and the value of the bitwise-inclusive OR of S_IRUSR, S_IWUSR, S_IRGRP,
35660 S_IWGRP, S_IROTH, and S_IWOTH used as the *mode* argument.
- 35661 2. The *utime()* function is called with the following arguments:
 - 35662 a. The *file* operand is used as the *path* argument.
 - 35663 b. The **utimbuf** structure members *actime* and *modtime* are determined as described in
35664 the OPTIONS section.

35665 OPTIONS

35666 The *touch* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section
35667 12.2, Utility Syntax Guidelines.

35668 The following options shall be supported:

- 35669 **-a** Change the access time of *file*. Do not change the modification time unless **-m** is
35670 also specified.
- 35671 **-c** Do not create a specified *file* if it does not exist. Do not write any diagnostic
35672 messages concerning this condition.
- 35673 **-m** Change the modification time of *file*. Do not change the access time unless **-a** is
35674 also specified.
- 35675 **-r ref_file** Use the corresponding time of the file named by the pathname *ref_file* instead of
35676 the current time.
- 35677 **-t time** Use the specified *time* instead of the current time. The option-argument shall be a
35678 decimal number of the form:

35679 $[[CC]YY]MMDDhhmm[.SS]$
35680 where each two digits represents the following:
 - 35681 **MM** The month of the year [01,12].
 - 35682 **DD** The day of the month [01,31].

35683 *hh* The hour of the day [00,23].
 35684 *mm* The minute of the hour [00,59].
 35685 *CC* The first two digits of the year (the century).
 35686 *YY* The second two digits of the year.
 35687 *SS* The second of the minute [00,60].

35688 Both *CC* and *YY* shall be optional. If neither is given, the current year shall be assumed. If *YY* is specified, but *CC* is not, *CC* shall be derived as follows:

If YY is:	<i>CC</i> becomes:
[69,99]	19
[00,68]	20

35693 **Note:** It is expected that in a future version of IEEE Std 1003.1-2001 the default century inferred from a 2-digit year will change. (This would apply to all commands accepting a 2-digit year as input.)

35696 The resulting time shall be affected by the value of the *TZ* environment variable. If
 35697 the resulting time value precedes the Epoch, *touch* shall exit immediately with an
 35698 error status. The range of valid times past the Epoch is implementation-defined,
 35699 but it shall extend to at least the time 0 hours, 0 minutes, 0 seconds, January 1,
 35700 2038, Coordinated Universal Time. Some implementations may not be able to
 35701 represent dates beyond January 18, 2038, because they use **signed int** as a time
 35702 holder.

35703 The range for *SS* is [00,60] rather than [00,59] because of leap seconds. If *SS* is 60,
 35704 and the resulting time, as affected by the *TZ* environment variable, does not refer
 35705 to a leap second, the resulting time shall be one second after a time where *SS* is 59.
 35706 If *SS* is not given a value, it is assumed to be zero.

35707 If neither the **-a** nor **-m** options were specified, *touch* shall behave as if both the **-a** and **-m**
 35708 options were specified.

35709 OPERANDS

35710 The following operands shall be supported:

35711 *file* A pathname of a file whose times shall be modified.

35712 STDIN

35713 Not used.

35714 INPUT FILES

35715 None.

35716 ENVIRONMENT VARIABLES

35717 The following environment variables shall affect the execution of *touch*:

35718 *LANG* Provide a default value for the internationalization variables that are unset or null.
 35719 (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2,
 35720 Internationalization Variables for the precedence of internationalization variables
 35721 used to determine the values of locale categories.)

35722 *LC_ALL* If set to a non-empty string value, override the values of all the other
 35723 internationalization variables.

35724 *LC_CTYPE* Determine the locale for the interpretation of sequences of bytes of text data as
 35725 characters (for example, single-byte as opposed to multi-byte characters in

35726 arguments).

35727 ***LC_MESSAGES***
35728 Determine the locale that should be used to affect the format and contents of
35729 diagnostic messages written to standard error.

35730 XSI ***NLSPATH*** Determine the location of message catalogs for the processing of ***LC_MESSAGES***.

35731 ***TZ*** Determine the timezone to be used for interpreting the *time* option-argument. If *TZ*
35732 is unset or null, an unspecified default timezone shall be used.

35733 **ASYNCHRONOUS EVENTS**

35734 Default.

35735 **STDOUT**

35736 Not used.

35737 **STDERR**

35738 The standard error shall be used only for diagnostic messages.

35739 **OUTPUT FILES**

35740 None.

35741 **EXTENDED DESCRIPTION**

35742 None.

35743 **EXIT STATUS**

35744 The following exit values shall be returned:

35745 0 The utility executed successfully and all requested changes were made.

35746 >0 An error occurred.

35747 **CONSEQUENCES OF ERRORS**

35748 Default.

35749 **APPLICATION USAGE**

35750 The interpretation of time is taken to be *seconds since the Epoch* (see the Base Definitions volume
35751 of IEEE Std 1003.1-2001, Section 4.14, Seconds Since the Epoch). It should be noted that
35752 implementations conforming to the System Interfaces volume of IEEE Std 1003.1-2001 do not
35753 take leap seconds into account when computing seconds since the Epoch. When *SS*=60 is used,
35754 the resulting time always refers to 1 plus *seconds since the Epoch* for a time when *SS*=59.

35755 Although the **-t** *time* option-argument specifies values in 1969, the access time and modification
35756 time fields are defined in terms of seconds since the Epoch (00:00:00 on 1 January 1970 UTC).
35757 Therefore, depending on the value of *TZ* when *touch* is run, there is never more than a few valid
35758 hours in 1969 and there need not be any valid times in 1969.

35759 One ambiguous situation occurs if **-t** *time* is not specified, **-r** *ref_file* is not specified, and the first
35760 operand is an eight or ten-digit decimal number. A portable script can avoid this problem by
35761 using:

35762 `touch -- file`

35763 or:

35764 `touch ./file`

35765 in this case.

35766 EXAMPLES

35767 None.

35768 RATIONALE

35769 The functionality of *touch* is described almost entirely through references to functions in the System Interfaces volume of IEEE Std 1003.1-2001. In this way, there is no duplication of effort required for describing such side effects as the relationship of user IDs to the user database, permissions, and so on.

35773 There are some significant differences between the *touch* utility in this volume of
35774 IEEE Std 1003.1-2001 and those in System V and BSD systems. They are upwards-compatible for
35775 historical applications from both implementations:

- 35776 1. In System V, an ambiguity exists when a pathname that is a decimal number leads the
35777 operands; it is treated as a time value. In BSD, no *time* value is allowed; files may only be
35778 *touched* to the current time. The **-t time** construct solves these problems for future
35779 conforming applications (note that the **-t** option is not historical practice).
- 35780 2. The inclusion of the century digits, *CC*, is also new. Note that a ten-digit *time* value is
35781 treated as if *YY*, and not *CC*, were specified. The caveat about the range of dates following
35782 the Epoch was included as recognition that some implementations are not able to
35783 represent dates beyond 18 January 2038 because they use **signed int** as a time holder.

35784 The **-r** option was added because several comments requested this capability. This option was
35785 named **-f** in an early proposal, but was changed because the **-f** option is used in the BSD version
35786 of *touch* with a different meaning.

35787 At least one historical implementation of *touch* incremented the exit code if **-c** was specified and
35788 the file did not exist. This volume of IEEE Std 1003.1-2001 requires exit status zero if no errors
35789 occur.

35790 FUTURE DIRECTIONS

35791 Applications should use the **-r** or **-t** options.

35792 SEE ALSO

35793 *date*, the System Interfaces volume of IEEE Std 1003.1-2001, *creat()*, *time()*, *utime()*, the Base
35794 Definitions volume of IEEE Std 1003.1-2001, **<sys/stat.h>**

35795 CHANGE HISTORY

35796 First released in Issue 2.

35797 Issue 6

35798 The obsolescent *date_time* operand is removed.

35799 The Open Group Corrigendum U027/1 is applied. This extends the range of valid time past the
35800 Epoch to at least the time 0 hours, 0 minutes, 0 seconds, January 1, 2038, Coordinated Universal
35801 Time. This is a new requirement on POSIX implementations.

35802 The range for seconds is changed from [00,61] to [00,60] to align with the ISO/IEC 9899:1999
35803 standard, and to allow for positive leap seconds.

35804 NAME

35805 *tput* — change terminal characteristics

35806 SYNOPSIS

35807 UP *tput* [-T *type*] *operand...*

35808

35809 DESCRIPTION

35810 The *tput* utility shall display terminal-dependent information. The manner in which this
35811 information is retrieved is unspecified. The information displayed shall clear the terminal screen,
35812 initialize the user's terminal, or reset the user's terminal, depending on the operand given. The
35813 exact consequences of displaying this information are unspecified.

35814 OPTIONS

35815 The *tput* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section
35816 12.2, Utility Syntax Guidelines.

35817 The following option shall be supported:

35818 **-T *type*** Indicate the type of terminal. If this option is not supplied and the *TERM* variable
35819 is unset or null, an unspecified default terminal type shall be used. The setting of
35820 *type* shall take precedence over the value in *TERM*.

35821 OPERANDS

35822 The following strings shall be supported as operands by the implementation in the POSIX locale:

35823 **clear** Display the clear-screen sequence.

35824 **init** Display the sequence that initializes the user's terminal in an implementation-
35825 defined manner.

35826 **reset** Display the sequence that resets the user's terminal in an implementation-defined
35827 manner.

35828 If a terminal does not support any of the operations described by these operands, this shall not
35829 be considered an error condition.

35830 STDIN

35831 Not used.

35832 INPUT FILES

35833 None.

35834 ENVIRONMENT VARIABLES

35835 The following environment variables shall affect the execution of *tput*:

35836 **LANG** Provide a default value for the internationalization variables that are unset or null.
35837 (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2,
35838 Internationalization Variables for the precedence of internationalization variables
35839 used to determine the values of locale categories.)

35840 **LC_ALL** If set to a non-empty string value, override the values of all the other
35841 internationalization variables.

35842 **LC_CTYPE** Determine the locale for the interpretation of sequences of bytes of text data as
35843 characters (for example, single-byte as opposed to multi-byte characters in
35844 arguments).

35845 **LC_MESSAGES**

35846 Determine the locale that should be used to affect the format and contents of
35847 diagnostic messages written to standard error.

35848 XSI	NLSPATH	Determine the location of message catalogs for the processing of <i>LC_MESSAGES</i> .
35849	TERM	Determine the terminal type. If this variable is unset or null, and if the -T option is not specified, an unspecified default terminal type shall be used.
35850		
35851	ASYNCHRONOUS EVENTS	
35852		Default.
35853	STDOUT	
35854		If standard output is a terminal device, it may be used for writing the appropriate sequence to
35855		clear the screen or reset or initialize the terminal. If standard output is not a terminal device,
35856		undefined results occur.
35857	STDERR	
35858		The standard error shall be used only for diagnostic messages.
35859	OUTPUT FILES	
35860		None.
35861	EXTENDED DESCRIPTION	
35862		None.
35863	EXIT STATUS	
35864		The following exit values shall be returned:
35865	0	The requested string was written successfully.
35866	1	Unspecified.
35867	2	Usage error.
35868	3	No information is available about the specified terminal type.
35869	4	The specified operand is invalid.
35870	>4	An error occurred.
35871	CONSEQUENCES OF ERRORS	
35872		If one of the operands is not available for the terminal, <i>tput</i> continues processing the remaining
35873		operands.
35874	APPLICATION USAGE	
35875		The difference between resetting and initializing a terminal is left unspecified, as they vary
35876		greatly based on hardware types. In general, resetting is a more severe action.
35877		Some terminals use control characters to perform the stated functions, and on such terminals it
35878		might make sense to use <i>tput</i> to store the initialization strings in a file or environment variable
35879		for later use. However, because other terminals might rely on system calls to do this work, the
35880		standard output cannot be used in a portable manner, such as the following non-portable
35881		constructs:
35882		ClearVar='tput clear'
35883		tput reset mailx -s "Wake Up" ddg
35884	EXAMPLES	
35885	1.	Initialize the terminal according to the type of terminal in the environmental variable
35886		TERM . This command can be included in a .profile file.
35887		tput init
35888	2.	Reset a 450 terminal.

35889 `tput -T 450 reset`

35890 RATIONALE

35891 The list of operands was reduced to a minimum for the following reasons:

- 35892 • The only features chosen were those that were likely to be used by human users interacting
35893 with a terminal.
- 35894 • Specifying the full *terminfo* set was not considered desirable, but the standard developers did
35895 not want to select among operands.
- 35896 • This volume of IEEE Std 1003.1-2001 does not attempt to provide applications with
35897 sophisticated terminal handling capabilities, as that falls outside of its assigned scope and
35898 intersects with the responsibilities of other standards bodies.

35899 The difference between resetting and initializing a terminal is left unspecified as this varies
35900 greatly based on hardware types. In general, resetting is a more severe action.

35901 The exit status of 1 is historically reserved for finding out if a Boolean operand is not set.
35902 Although the operands were reduced to a minimum, the exit status of 1 should still be reserved
35903 for the Boolean operands, for those sites that wish to support them.

35904 FUTURE DIRECTIONS

35905 None.

35906 SEE ALSO

35907 `stty`, `tabs`

35908 CHANGE HISTORY

35909 First released in Issue 4.

35910 Issue 6

35911 This utility is marked as part of the User Portability Utilities option.

35912 NAME

35913 tr — translate characters

35914 SYNOPSIS

```
35915       tr [-c | -C][-s] string1 string2
35916       tr -s [-c | -C] string1
35917       tr -d [-c | -C] string1
35918       tr -ds [-c | -C] string1 string2
```

35919 DESCRIPTION

35920 The *tr* utility shall copy the standard input to the standard output with substitution or deletion
35921 of selected characters. The options specified and the *string1* and *string2* operands shall control
35922 translations that occur while copying characters and single-character collating elements.

35923 OPTIONS

35924 The *tr* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section 12.2,
35925 Utility Syntax Guidelines.

35926 The following options shall be supported:

- | | |
|-----------------|--|
| 35927 -c | Complement the set of values specified by <i>string1</i> . See the EXTENDED
35928 DESCRIPTION section. |
| 35929 -C | Complement the set of characters specified by <i>string1</i> . See the EXTENDED
35930 DESCRIPTION section. |
| 35931 -d | Delete all occurrences of input characters that are specified by <i>string1</i> . |
| 35932 -s | Replace instances of repeated characters with a single character, as described in the
35933 EXTENDED DESCRIPTION section. |

35934 OPERANDS

35935 The following operands shall be supported:

35936 *string1, string2*

35937 Translation control strings. Each string shall represent a set of characters to be
35938 converted into an array of characters used for the translation. For a detailed
35939 description of how the strings are interpreted, see the EXTENDED DESCRIPTION
35940 section.

35941 STDIN

35942 The standard input can be any type of file.

35943 INPUT FILES

35944 None.

35945 ENVIRONMENT VARIABLES

35946 The following environment variables shall affect the execution of *tr*:

- | | |
|-------------------------|--|
| 35947 LANG | Provide a default value for the internationalization variables that are unset or null.
(See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2,
35948 Internationalization Variables for the precedence of internationalization variables
35949 used to determine the values of locale categories.) |
| 35951 LC_ALL | If set to a non-empty string value, override the values of all the other
35952 internationalization variables. |
| 35953 LC_COLLATE | Determine the locale for the behavior of range expressions and equivalence classes. |
| 35954 | |

35955	<i>LC_CTYPE</i>	Determine the locale for the interpretation of sequences of bytes of text data as characters (for example, single-byte as opposed to multi-byte characters in arguments) and the behavior of character classes.
35956		
35957		
35958	<i>LC_MESSAGES</i>	Determine the locale that should be used to affect the format and contents of diagnostic messages written to standard error.
35959		
35960		
35961 XSI	<i>NLSPATH</i>	Determine the location of message catalogs for the processing of <i>LC_MESSAGES</i> .
35962	ASYNCHRONOUS EVENTS	
35963		Default.
35964	STDOUT	
35965		The <i>tr</i> output shall be identical to the input, with the exception of the specified transformations.
35966	STDERR	
35967		The standard error shall be used only for diagnostic messages.
35968	OUTPUT FILES	
35969		None.
35970	EXTENDED DESCRIPTION	
35971		The operands <i>string1</i> and <i>string2</i> (if specified) define two arrays of characters. The constructs in
35972		the following list can be used to specify characters or single-character collating elements. If any
35973		of the constructs result in multi-character collating elements, <i>tr</i> shall exclude, without a
35974		diagnostic, those multi-character elements from the resulting array.
35975	<i>character</i>	Any character not described by one of the conventions below shall represent itself.
35976	<i>\octal</i>	Octal sequences can be used to represent characters with specific coded values. An octal sequence shall consist of a backslash followed by the longest sequence of one,
35977		two, or three-octal-digit characters (01234567). The sequence shall cause the value
35978		whose encoding is represented by the one, two, or three-digit octal integer to be
35979		placed into the array. Multi-byte characters require multiple, concatenated escape
35980		sequences of this type, including the leading '\ ' for each byte.
35981		2
35982	<i>\character</i>	The backslash-escape sequences in the Base Definitions volume of
35983		IEEE Std 1003.1-2001, Table 5-1, Escape Sequences and Associated Actions ('\\',
35984		'\a', '\b', '\f', '\n', '\r', '\t', '\v') shall be supported. The results of
35985		using any other character, other than an octal digit, following the backslash are
35986		unspecified.
35987	<i>c–c</i>	In the POSIX locale, this construct shall represent the range of collating elements
35988		between the range endpoints (as long as neither endpoint is an octal sequence of
35989		the form <i>\octal</i>), inclusive, as defined by the collation sequence. The characters or
35990		collating elements in the range shall be placed in the array in ascending collation
35991		sequence. If the second endpoint precedes the starting endpoint in the collation
35992		sequence, it is unspecified whether the range of collating elements is empty, or this
35993		construct is treated as invalid. In locales other than the POSIX locale, this construct
35994		has unspecified behavior.
35995		If either or both of the range endpoints are octal sequences of the form <i>\octal</i> , this
35996		shall represent the range of specific coded values between the two range
35997		endpoints, inclusive.
35998	<i>[:class:]</i>	Represents all characters belonging to the defined character class, as defined by the
35999		current setting of the <i>LC_CTYPE</i> locale category. The following character class
36000		names shall be accepted when specified in <i>string1</i> :

36001		alnum	blank	digit	lower	punct	upper
36002		alpha	cntrl	graph	print	space	xdigit
36003	xsi	In addition, character class expressions of the form [:name:] shall be recognized in those locales where the <i>name</i> keyword has been given a charclass definition in the <i>LC_CTYPE</i> category.					
36004							
36005							
36006		When both the -d and -s options are specified, any of the character class names shall be accepted in <i>string2</i> . Otherwise, only character class names lower or upper are valid in <i>string2</i> and then only if the corresponding character class (upper and lower , respectively) is specified in the same relative position in <i>string1</i> . Such a specification shall be interpreted as a request for case conversion. When [:lower:] appears in <i>string1</i> and [:upper:] appears in <i>string2</i> , the arrays shall contain the characters from the toupper mapping in the <i>LC_CTYPE</i> category of the current locale. When [:upper:] appears in <i>string1</i> and [:lower:] appears in <i>string2</i> , the arrays shall contain the characters from the tolower mapping in the <i>LC_CTYPE</i> category of the current locale. The first character from each mapping pair shall be in the array for <i>string1</i> and the second character from each mapping pair shall be in the array for <i>string2</i> in the same relative position.					
36007							
36008							
36009							
36010							
36011							
36012							
36013							
36014							
36015							
36016							
36017							
36018		Except for case conversion, the characters specified by a character class expression shall be placed in the array in an unspecified order.					
36019							
36020		If the name specified for <i>class</i> does not define a valid character class in the current locale, the behavior is undefined.					
36021							
36022	[=equiv=]	Represents all characters or collating elements belonging to the same equivalence class as <i>equiv</i> , as defined by the current setting of the <i>LC_COLLATE</i> locale category. An equivalence class expression shall be allowed only in <i>string1</i> , or in <i>string2</i> when it is being used by the combined -d and -s options. The characters belonging to the equivalence class shall be placed in the array in an unspecified order.					
36023							
36024							
36025							
36026							
36027							
36028	[x*n]	Represents <i>n</i> repeated occurrences of the character <i>x</i> . Because this expression is used to map multiple characters to one, it is only valid when it occurs in <i>string2</i> . If <i>n</i> is omitted or is zero, it shall be interpreted as large enough to extend the <i>string2</i> -based sequence to the length of the <i>string1</i> -based sequence. If <i>n</i> has a leading zero, it shall be interpreted as an octal value. Otherwise, it shall be interpreted as a decimal value.					
36029							
36030							
36031							
36032							
36033							
36034		When the -d option is not specified:					
36035							
36036							
36037							
36038							
36039							
36040							
36041							
36042							
36043							
36044							
36045							
36046							

36047 When the **-d** option is specified:

- Input characters found in the array specified by *string1* shall be deleted.
- When the **-C** option is specified with **-d**, all characters except those specified by *string1* shall be deleted. The contents of *string2* are ignored, unless the **-s** option is also specified.
- When the **-c** option is specified with **-d**, all values except those specified by *string1* shall be deleted. The contents of *string2* shall be ignored, unless the **-s** option is also specified.
- The same string cannot be used for both the **-d** and the **-s** option; when both options are specified, both *string1* (used for deletion) and *string2* (used for squeezing) shall be required.

36055 When the **-s** option is specified, after any deletions or translations have taken place, repeated
36056 sequences of the same character shall be replaced by one occurrence of the same character, if the
36057 character is found in the array specified by the last operand. If the last operand contains a
36058 character class, such as the following example:

36059 `tr -s '[:space:]'`

36060 the last operand's array shall contain all of the characters in that character class. However, in a
36061 case conversion, as described previously, such as:

36062 `tr -s '[:upper:]' '[:lower:]'`

36063 the last operand's array shall contain only those characters defined as the second characters in
36064 each of the **toupper** or **tolower** character pairs, as appropriate.

36065 An empty string used for *string1* or *string2* produces undefined results.

36066 EXIT STATUS

36067 The following exit values shall be returned:

36068 0 All input was processed successfully.

36069 >0 An error occurred.

36070 CONSEQUENCES OF ERRORS

36071 Default.

36072 APPLICATION USAGE

36073 If necessary, *string1* and *string2* can be quoted to avoid pattern matching by the shell.

36074 If an ordinary digit (representing itself) is to follow an octal sequence, the octal sequence must
36075 use the full three digits to avoid ambiguity.

36076 When *string2* is shorter than *string1*, a difference results between historical System V and BSD
36077 systems. A BSD system pads *string2* with the last character found in *string2*. Thus, it is possible
36078 to do the following:

36079 `tr 0123456789 d`

36080 which would translate all digits to the letter '`d`'. Since this area is specifically unspecified in
36081 this volume of IEEE Std 1003.1-2001, both the BSD and System V behaviors are allowed, but a
36082 conforming application cannot rely on the BSD behavior. It would have to code the example in
36083 the following way:

36084 `tr 0123456789 '[d*]'`

36085 It should be noted that, despite similarities in appearance, the string operands used by *tr* are not
36086 regular expressions.

36087 Unlike some historical implementations, this definition of the *tr* utility correctly processes NUL
36088 characters in its input stream. NUL characters can be stripped by using:

36089 `tr -d '\000'`

36090 EXAMPLES

- 36091 1. The following example creates a list of all words in **file1** one per line in **file2**, where a word
36092 is taken to be a maximal string of letters.

36093 `tr -cs "[[:alpha:]]" "[\n*]" <file1 >file2`

- 36094 2. The next example translates all lowercase characters in **file1** to uppercase and writes the
36095 results to standard output.

36096 `tr "[[:lower:]]" "[[:upper:]]" <file1`

- 36097 3. This example uses an equivalence class to identify accented variants of the base character
36098 'e' in **file1**, which are stripped of diacritical marks and written to **file2**.

36099 `tr "[=e=]" "[e*]" <file1 >file2`

2

36100 RATIONALE

36101 In some early proposals, an explicit option **-n** was added to disable the historical behavior of
36102 stripping NUL characters from the input. It was considered that automatically stripping NUL
36103 characters from the input was not correct functionality. However, the removal of **-n** in a later
36104 proposal does not remove the requirement that *tr* correctly process NUL characters in its input
36105 stream. NUL characters can be stripped by using *tr -d '\000'*.

36106 Historical implementations of *tr* differ widely in syntax and behavior. For example, the BSD
36107 version has not needed the bracket characters for the repetition sequence. The *tr* utility syntax is
36108 based more closely on the System V and XPG3 model while attempting to accommodate
36109 historical BSD implementations. In the case of the short *string2* padding, the decision was to
36110 unspecify the behavior and preserve System V and XPG3 scripts, which might find difficulty
36111 with the BSD method. The assumption was made that BSD users of *tr* have to make
36112 accommodations to meet the syntax defined here. Since it is possible to use the repetition
36113 sequence to duplicate the desired behavior, whereas there is no simple way to achieve the
36114 System V method, this was the correct, if not desirable, approach.

36115 The use of octal values to specify control characters, while having historical precedents, is not
36116 portable. The introduction of escape sequences for control characters should provide the
36117 necessary portability. It is recognized that this may cause some historical scripts to break.

36118 An early proposal included support for multi-character collating elements. It was pointed out
36119 that, while *tr* does employ some syntactical elements from REs, the aim of *tr* is quite different;
36120 ranges, for example, do not have a similar meaning ("any of the chars in the range matches",
36121 versus "translate each character in the range to the output counterpart"). As a result, the
36122 previously included support for multi-character collating elements has been removed. What
36123 remains are ranges in current collation order (to support, for example, accented characters),
36124 character classes, and equivalence classes.

36125 In XPG3 the `[:class:]` and `[=equiv=]` conventions are shown with double brackets, as in RE syntax.
36126 However, *tr* does not implement RE principles; it just borrows part of the syntax. Consequently,
36127 `[:class:]` and `[=equiv=]` should be regarded as syntactical elements on a par with `[x*n]`, which is
36128 not an RE bracket expression.

36129 The standard developers will consider changes to *tr* that allow it to translate characters between
36130 different character encodings, or they will consider providing a new utility to accomplish this.

36131	On historical System V systems, a range expression requires enclosing square-brackets, such as:	
36132	tr '[a-z]' '[A-Z]'	
36133	However, BSD-based systems did not require the brackets, and this convention is used here to	
36134	avoid breaking large numbers of BSD scripts:	
36135	tr a-z A-Z	
36136	The preceding System V script will continue to work because the brackets, treated as regular	
36137	characters, are translated to themselves. However, any System V script that relied on "a-z"	
36138	representing the three characters 'a', '−', and 'z' have to be rewritten as "az−".	
36139	The ISO POSIX-2:1993 standard had a −c option that behaved similarly to the −C option, but did	
36140	not supply functionality equivalent to the −c option specified in IEEE Std 1003.1-2001. This	
36141	meant that historical practice of being able to specify <i>tr −d\200−\377</i> (which would delete all	
36142	bytes with the top bit set) would have no effect because, in the C locale, bytes with the values	
36143	octal 200 to octal 377 are not characters.	
36144	The earlier version also said that octal sequences referred to collating elements and could be	
36145	placed adjacent to each other to specify multi-byte characters. However, it was noted that this	
36146	caused ambiguities because <i>tr</i> would not be able to tell whether adjacent octal sequences were	
36147	intending to specify multi-byte characters or multiple single byte characters.	
36148	IEEE Std 1003.1-2001 specifies that octal sequences always refer to single byte binary values	2
36149	when used to specify an endpoint of a range of collating elements.	2
36150	Previous versions of this standard allowed for implementations with bytes other than eight bits,	2
36151	but this has been modified in this version.	2
36152	FUTURE DIRECTIONS	
36153	None.	
36154	SEE ALSO	
36155	<i>sed</i>	
36156	CHANGE HISTORY	
36157	First released in Issue 2.	
36158	Issue 6	
36159	The −C operand is added, and the description of the −c operand is changed to align with the	
36160	IEEE P1003.2b draft standard.	
36161	The normative text is reworded to avoid use of the term “must” for application requirements.	
36162	IEEE Std 1003.1-2001/Cor 2-2004, item XCU/TC2/D6/31 is applied, removing text describing	2
36163	behavior on systems with bytes consisting of more than eight bits.	2
36164	IEEE Std 1003.1-2001/Cor 2-2004, item XCU/TC2/D6/32 is applied, updating an example in the	2
36165	EXAMPLES section to avoid using unspecified behavior.	2
36166	IEEE Std 1003.1-2001/Cor 2-2004, item XCU/TC2/D6/33 is applied, making a correction to the	2
36167	RATIONALE.	2

36168 NAME

36169 true — return true value

36170 SYNOPSIS

36171 true

36172 DESCRIPTION

36173 The *true* utility shall return with exit code zero.

36174 OPTIONS

36175 None.

36176 OPERANDS

36177 None.

36178 STDIN

36179 Not used.

36180 INPUT FILES

36181 None.

36182 ENVIRONMENT VARIABLES

36183 None.

36184 ASYNCHRONOUS EVENTS

36185 Default.

36186 STDOUT

36187 Not used.

1

36188 STDERR

36189 Not used.

36190 OUTPUT FILES

36191 None.

36192 EXTENDED DESCRIPTION

36193 None.

1

36194 EXIT STATUS

36195 Zero.

36196 CONSEQUENCES OF ERRORS

36197 None.

36198 APPLICATION USAGE

36199 This utility is typically used in shell scripts, as shown in the EXAMPLES section. The special
36200 built-in utility : is sometimes more efficient than *true*.

36201 EXAMPLES

36202 This command is executed forever:

36203 while true

36204 do

36205 command

36206 done

36207 RATIONALE

36208 The *true* utility has been retained in this volume of IEEE Std 1003.1-2001, even though the shell
36209 special built-in : provides similar functionality, because *true* is widely used in historical scripts
36210 and is less cryptic to novice script readers.

36211 FUTURE DIRECTIONS

36212 None.

36213 SEE ALSO

36214 *false*, Section 2.9 (on page 47)

36215 CHANGE HISTORY

36216 First released in Issue 2.

36217 Issue 6

36218 IEEE Std 1003.1-2001/Cor 1-2002, item XCU/TC1/D6/39 is applied, replacing the terms “None” 1
36219 and “Default” from the STDERR and EXIT STATUS sections, respectively, with terms as defined 1
36220 in Section 1.11 (on page 20). 1

36221 NAME

36222 tsort — topological sort

36223 SYNOPSIS

36224 XSI tsort [file]

36225

36226 DESCRIPTION

36227 The *tsort* utility shall write to standard output a totally ordered list of items consistent with a
36228 partial ordering of items contained in the input.

36229 The application shall ensure that the input consists of pairs of items (non-empty strings)
36230 separated by <blank>s. Pairs of different items indicate ordering. Pairs of identical items
36231 indicate presence, but not ordering.

36232 OPTIONS

36233 None.

36234 OPERANDS

36235 The following operand shall be supported:

36236 *file* A pathname of a text file to order. If no *file* operand is given, the standard input
36237 shall be used.

36238 STDIN

36239 The standard input shall be a text file that is used if no *file* operand is given.

36240 INPUT FILES

36241 The input file named by the *file* operand is a text file.

36242 ENVIRONMENT VARIABLES

36243 The following environment variables shall affect the execution of *tsort*:

36244 *LANG* Provide a default value for the internationalization variables that are unset or null.
36245 (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2,
36246 Internationalization Variables for the precedence of internationalization variables
36247 used to determine the values of locale categories.)

36248 *LC_ALL* If set to a non-empty string value, override the values of all the other
36249 internationalization variables.

36250 *LC_CTYPE* Determine the locale for the interpretation of sequences of bytes of text data as
36251 characters (for example, single-byte as opposed to multi-byte characters in
36252 arguments and input files).

36253 *LC_MESSAGES*

36254 Determine the locale that should be used to affect the format and contents of
36255 diagnostic messages written to standard error.

36256 *NLSPATH* Determine the location of message catalogs for the processing of *LC_MESSAGES*.

36257 ASYNCHRONOUS EVENTS

36258 Default.

36259 STDOUT

36260 The standard output shall be a text file consisting of the order list produced from the partially
36261 ordered input.

36262 STDERR

36263 The standard error shall be used only for diagnostic messages.

36264 OUTPUT FILES

36265 None.

36266 EXTENDED DESCRIPTION

36267 None.

36268 EXIT STATUS

36269 The following exit values shall be returned:

36270 0 Successful completion.

36271 >0 An error occurred.

36272 CONSEQUENCES OF ERRORS

36273 Default.

36274 APPLICATION USAGE

36275 The *LC_COLLATE* variable need not affect the actions of *tsort*. The output ordering is not lexicographic, but depends on the pairs of items given as input.

36277 EXAMPLES

36278 The command:

```
36279     tsort <<EOF
36280     a b c c d e
36281     g g
36282     f g e f
36283     h h
36284     EOF
```

36285 produces the output:

```
36286     a
36287     b
36288     c
36289     d
36290     e
36291     f
36292     g
36293     h
```

36294 RATIONALE

36295 None.

36296 FUTURE DIRECTIONS

36297 None.

36298 SEE ALSO

36299 None.

36300 CHANGE HISTORY

36301 First released in Issue 2.

36302 Issue 6

36303 The normative text is reworded to avoid use of the term “must” for application requirements.

36304 NAME

36305 *tty* — return user's terminal name

36306 SYNOPSIS

36307 *tty*

36308 DESCRIPTION

36309 The *tty* utility shall write to the standard output the name of the terminal that is open as
36310 standard input. The name that is used shall be equivalent to the string that would be returned by
36311 the *ttynname()* function defined in the System Interfaces volume of IEEE Std 1003.1-2001.

36312 OPTIONS

36313 The *tty* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section 12.2,
36314 Utility Syntax Guidelines.

36315 OPERANDS

36316 None.

36317 STDIN

36318 While no input is read from standard input, standard input shall be examined to determine
36319 whether or not it is a terminal, and, if so, to determine the name of the terminal.

36320 INPUT FILES

36321 None.

36322 ENVIRONMENT VARIABLES

36323 The following environment variables shall affect the execution of *tty*:

36324 *LANG* Provide a default value for the internationalization variables that are unset or null.
36325 (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2,
36326 Internationalization Variables for the precedence of internationalization variables
36327 used to determine the values of locale categories.)

36328 *LC_ALL* If set to a non-empty string value, override the values of all the other
36329 internationalization variables.

36330 *LC_CTYPE* Determine the locale for the interpretation of sequences of bytes of text data as
36331 characters (for example, single-byte as opposed to multi-byte characters in
36332 arguments).

36333 *LC_MESSAGES*

36334 Determine the locale that should be used to affect the format and contents of
36335 diagnostic messages written to standard error and informative messages written to
36336 standard output.

36337 XSI *NLSPATH* Determine the location of message catalogs for the processing of *LC_MESSAGES*.

36338 ASYNCHRONOUS EVENTS

36339 Default.

36340 STDOUT

36341 If standard input is a terminal device, a pathname of the terminal as specified by the *ttynname()*
36342 function defined in the System Interfaces volume of IEEE Std 1003.1-2001 shall be written in the
36343 following format:

36344 "%s\n", <terminal name>

36345 Otherwise, a message shall be written indicating that standard input is not connected to a
36346 terminal. In the POSIX locale, the *tty* utility shall use the format:

36347 "not a tty\n"

36348 STDERR

36349 The standard error shall be used only for diagnostic messages.

36350 OUTPUT FILES

36351 None.

36352 EXTENDED DESCRIPTION

36353 None.

36354 EXIT STATUS

36355 The following exit values shall be returned:

36356 0 Standard input is a terminal.

36357 1 Standard input is not a terminal.

36358 >1 An error occurred.

36359 CONSEQUENCES OF ERRORS

36360 Default.

36361 APPLICATION USAGE

36362 This utility checks the status of the file open as standard input against that of an implementation-defined set of files. It is possible that no match can be found, or that the match found need not be the same file as that which was opened for standard input (although they are the same device).

36366 EXAMPLES

36367 None.

36368 RATIONALE

36369 None.

36370 FUTURE DIRECTIONS

36371 None.

36372 SEE ALSO

36373 The System Interfaces volume of IEEE Std 1003.1-2001, *isatty()*, *ttynname()*

36374 CHANGE HISTORY

36375 First released in Issue 2.

36376 Issue 5

36377 The SYNOPSIS is changed to indicate two forms of the command, with the second form marked as obsolete. This is a clarification and does not change the functionality published in previous issues.

36380 Issue 6

36381 The obsolescent **-s** option is removed.

36382 **NAME**

36383 type — write a description of command type

36384 **SYNOPSIS**

36385 XSI type name...

36386

36387 **DESCRIPTION**

36388 The *type* utility shall indicate how each argument would be interpreted if used as a command name.

36390 **OPTIONS**

36391 None.

36392 **OPERANDS**

36393 The following operand shall be supported:

36394 name A name to be interpreted.

36395 **STDIN**

36396 Not used.

36397 **INPUT FILES**

36398 None.

36399 **ENVIRONMENT VARIABLES**

36400 The following environment variables shall affect the execution of *type*:

36401 *LANG* Provide a default value for the internationalization variables that are unset or null. (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2, Internationalization Variables for the precedence of internationalization variables used to determine the values of locale categories.)

36405 *LC_ALL* If set to a non-empty string value, override the values of all the other internationalization variables.

36407 *LC_CTYPE* Determine the locale for the interpretation of sequences of bytes of text data as characters (for example, single-byte as opposed to multi-byte characters in arguments).

36410 *LC_MESSAGES*

36411 Determine the locale that should be used to affect the format and contents of diagnostic messages written to standard error.

36413 *NLSPATH* Determine the location of message catalogs for the processing of *LC_MESSAGES*.

36414 *PATH* Determine the location of *name*, as described in the Base Definitions volume of IEEE Std 1003.1-2001, Chapter 8, Environment Variables.

36416 **ASYNCHRONOUS EVENTS**

36417 Default.

36418 **STDOUT**

36419 The standard output of *type* contains information about each operand in an unspecified format. The information provided typically identifies the operand as a shell built-in, function, alias, or keyword, and where applicable, may display the operand's pathname.

36422 STDERR

36423 The standard error shall be used only for diagnostic messages.

36424 OUTPUT FILES

36425 None.

36426 EXTENDED DESCRIPTION

36427 None.

36428 EXIT STATUS

36429 The following exit values shall be returned:

36430 0 Successful completion.

36431 >0 An error occurred.

36432 CONSEQUENCES OF ERRORS

36433 Default.

36434 APPLICATION USAGE

36435 Since *type* must be aware of the contents of the current shell execution environment (such as the lists of commands, functions, and built-ins processed by *hash*), it is always provided as a shell regular built-in. If it is called in a separate utility execution environment, such as one of the following:

36439 nohup type writer

36440 find . -type f | xargs type

36441 it might not produce accurate results.

36442 EXAMPLES

36443 None.

36444 RATIONALE

36445 None.

36446 FUTURE DIRECTIONS

36447 None.

36448 SEE ALSO

36449 *command, hash*

36450 CHANGE HISTORY

36451 First released in Issue 2.

36452 NAME

36453 **ulimit** — set or report file size limit

36454 SYNOPSIS

36455 XSI **ulimit [-f][blocks]**

36456

36457 DESCRIPTION

36458 The **ulimit** utility shall set or report the file-size writing limit imposed on files written by the shell and its child processes (files of any size may be read). Only a process with appropriate privileges can increase the limit.

36461 OPTIONS

36462 The **ulimit** utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section 12.2, Utility Syntax Guidelines.

36464 The following option shall be supported:

36465 **-f** Set (or report, if no *blocks* operand is present), the file size limit in blocks. The **-f** option shall also be the default case.

36467 OPERANDS

36468 The following operand shall be supported:

36469 *blocks* The number of 512-byte blocks to use as the new file size limit.

36470 STDIN

36471 Not used.

36472 INPUT FILES

36473 None.

36474 ENVIRONMENT VARIABLES

36475 The following environment variables shall affect the execution of **ulimit**:

36476 **LANG** Provide a default value for the internationalization variables that are unset or null. (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2, Internationalization Variables for the precedence of internationalization variables used to determine the values of locale categories.)

36480 **LC_ALL** If set to a non-empty string value, override the values of all the other internationalization variables.

36482 **LC_CTYPE** Determine the locale for the interpretation of sequences of bytes of text data as characters (for example, single-byte as opposed to multi-byte characters in arguments).

36485 *LC_MESSAGES*

36486 Determine the locale that should be used to affect the format and contents of diagnostic messages written to standard error.

36488 **NLSPATH** Determine the location of message catalogs for the processing of **LC_MESSAGES**.

36489 ASYNCHRONOUS EVENTS

36490 Default.

36491 STDOUT

36492 The standard output shall be used when no *blocks* operand is present. If the current number of blocks is limited, the number of blocks in the current limit shall be written in the following format:

36495 "%d\n", <number of 512-byte blocks>
36496 If there is no current limit on the number of blocks, in the POSIX locale the following format
36497 shall be used:
36498 "unlimited\n"

36499 STDRERR

36500 The standard error shall be used only for diagnostic messages.

36501 OUTPUT FILES

36502 None.

36503 EXTENDED DESCRIPTION

36504 None.

36505 EXIT STATUS

36506 The following exit values shall be returned:

36507 0 Successful completion.

36508 >0 A request for a higher limit was rejected or an error occurred.

36509 CONSEQUENCES OF ERRORS

36510 Default.

36511 APPLICATION USAGE

36512 Since *ulimit* affects the current shell execution environment, it is always provided as a shell
36513 regular built-in. If it is called in a separate utility execution environment, such as one of the
36514 following:

36515 nohup ulimit -f 10000

36516 env ulimit 10000

36517 it does not affect the file size limit of the caller's environment.

36518 Once a limit has been decreased by a process, it cannot be increased (unless appropriate
36519 privileges are involved), even back to the original system limit.

36520 EXAMPLES

36521 Set the file size limit to 51 200 bytes:

36522 ulimit -f 100

36523 RATIONALE

36524 None.

36525 FUTURE DIRECTIONS

36526 None.

36527 SEE ALSO

36528 The System Interfaces volume of IEEE Std 1003.1-2001, *ulimit()*

36529 CHANGE HISTORY

36530 First released in Issue 2.

36531 NAME

36532 umask — get or set the file mode creation mask

36533 SYNOPSIS

36534 umask [-S][*mask*]

36535 DESCRIPTION

36536 The *umask* utility shall set the file mode creation mask of the current shell execution environment (see Section 2.12 (on page 61)) to the value specified by the *mask* operand. This mask shall affect the initial value of the file permission bits of subsequently created files. If *umask* is called in a subshell or separate utility execution environment, such as one of the following:

36540 (umask 002)
36541 nohup umask ...
36542 find . -exec umask ... \;

36543 it shall not affect the file mode creation mask of the caller's environment.

36544 If the *mask* operand is not specified, the *umask* utility shall write to standard output the value of
36545 the invoking process' file mode creation mask.

36546 OPTIONS

36547 The *umask* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section
36548 12.2, Utility Syntax Guidelines.

36549 The following option shall be supported:

36550 -S Produce symbolic output.

36551 The default output style is unspecified, but shall be recognized on a subsequent invocation of
36552 *umask* on the same system as a *mask* operand to restore the previous file mode creation mask.

36553 OPERANDS

36554 The following operand shall be supported:

36555 *mask* A string specifying the new file mode creation mask. The string is treated in the
36556 same way as the *mode* operand described in the EXTENDED DESCRIPTION
36557 section for *chmod*.

36558 For a *symbolic_mode* value, the new value of the file mode creation mask shall be
36559 the logical complement of the file permission bits portion of the file mode specified
36560 by the *symbolic_mode* string.

36561 In a *symbolic_mode* value, the permissions *op* characters '+' and '-' shall be
36562 interpreted relative to the current file mode creation mask; '+' shall cause the bits
36563 for the indicated permissions to be cleared in the mask; '-' shall cause the bits for
36564 the indicated permissions to be set in the mask.

36565 The interpretation of *mode* values that specify file mode bits other than the file
36566 permission bits is unspecified.

36567 In the octal integer form of *mode*, the specified bits are set in the file mode creation
36568 mask.

36569 The file mode creation mask shall be set to the resulting numeric value.

36570 The default output of a prior invocation of *umask* on the same system with no
36571 operand also shall be recognized as a *mask* operand.

36572 STDIN

36573 Not used.

36574 INPUT FILES

36575 None.

36576 ENVIRONMENT VARIABLES

36577 The following environment variables shall affect the execution of *umask*:

36578 *LANG* Provide a default value for the internationalization variables that are unset or null.
36579 (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2,
36580 Internationalization Variables for the precedence of internationalization variables
36581 used to determine the values of locale categories.)

36582 *LC_ALL* If set to a non-empty string value, override the values of all the other
36583 internationalization variables.

36584 *LC_CTYPE* Determine the locale for the interpretation of sequences of bytes of text data as
36585 characters (for example, single-byte as opposed to multi-byte characters in
36586 arguments).

36587 *LC_MESSAGES*

36588 Determine the locale that should be used to affect the format and contents of
36589 diagnostic messages written to standard error.

36590 XSI *NLSPATH* Determine the location of message catalogs for the processing of *LC_MESSAGES*.

36591 ASYNCHRONOUS EVENTS

36592 Default.

36593 STDOUT

36594 When the *mask* operand is not specified, the *umask* utility shall write a message to standard
36595 output that can later be used as a *umask mask* operand.

36596 If **-S** is specified, the message shall be in the following format:

36597 "u=%s,g=%s,o=%s\n", <owner permissions>, <group permissions>,
36598 <other permissions>

36599 where the three values shall be combinations of letters from the set {r, w, x}; the presence of a
36600 letter shall indicate that the corresponding bit is clear in the file mode creation mask.

36601 If a *mask* operand is specified, there shall be no output written to standard output.

36602 STDERR

36603 The standard error shall be used only for diagnostic messages.

36604 OUTPUT FILES

36605 None.

36606 EXTENDED DESCRIPTION

36607 None.

36608 EXIT STATUS

36609 The following exit values shall be returned:

36610 0 The file mode creation mask was successfully changed, or no *mask* operand was supplied.

36611 >0 An error occurred.

36612 CONSEQUENCES OF ERRORS

36613 Default.

36614 APPLICATION USAGE

36615 Since *umask* affects the current shell execution environment, it is generally provided as a shell
36616 regular built-in.36617 In contrast to the negative permission logic provided by the file mode creation mask and the
36618 octal number form of the *mask* argument, the symbolic form of the *mask* argument specifies those
36619 permissions that are left alone.

36620 EXAMPLES

36621 Either of the commands:

36622 `umask a=rx,ug+w`36623 `umask 002`

36624 sets the mode mask so that subsequently created files have their S_IWOTH bit cleared.

36625 After setting the mode mask with either of the above commands, the *umask* command can be
36626 used to write out the current value of the mode mask:36627 `$ umask`36628 `0002`36629 (The output format is unspecified, but historical implementations use the octal integer mode
36630 format.)36631 `$ umask -S`36632 `u=rwx,g=rwx,o=rx`36633 Either of these outputs can be used as the mask operand to a subsequent invocation of the *umask*
36634 utility.

36635 Assuming the mode mask is set as above, the command:

36636 `umask g-w`36637 sets the mode mask so that subsequently created files have their S_IWGRP and S_IWOTH bits
36638 cleared.

36639 The command:

36640 `umask -- -w`36641 sets the mode mask so that subsequently created files have all their write bits cleared. Note that
36642 *mask* operands `-r`, `-w`, `-x` or anything beginning with a hyphen, must be preceded by `--` to
36643 keep it from being interpreted as an option.

36644 RATIONALE

36645 Since *umask* affects the current shell execution environment, it is generally provided as a shell
36646 regular built-in. If it is called in a subshell or separate utility execution environment, such as one
36647 of the following:36648 `(umask 002)`36649 `nohup umask ...`36650 `find . -exec umask ... \;`

36651 it does not affect the file mode creation mask of the environment of the caller.

36652 The description of the historical utility was modified to allow it to use the symbolic modes of
36653 *chmod*. The `-s` option used in early proposals was changed to `-S` because `-s` could be confused

36654	with a <i>symbolic_mode</i> form of mask referring to the S_ISUID and S_ISGID bits.	
36655	The default output style is unspecified to permit implementors to provide migration to the new symbolic style at the time most appropriate to their users. A -o flag to force octal mode output was omitted because the octal mode may not be sufficient to specify all of the information that may be present in the file mode creation mask when more secure file access permission checks are implemented.	2
36660	It has been suggested that trusted systems developers might appreciate ameliorating the requirement that the mode mask “affects” the file access permissions, since it seems access control lists might replace the mode mask to some degree. The wording has been changed to say that it affects the file permission bits, and it leaves the details of the behavior of how they affect the file access permissions to the description in the System Interfaces volume of IEEE Std 1003.1-2001.	
36666	FUTURE DIRECTIONS	
36667	None.	
36668	SEE ALSO	
36669	Chapter 2 (on page 29), <i>chmod</i> , the System Interfaces volume of IEEE Std 1003.1-2001, <i>umask()</i>	
36670	CHANGE HISTORY	
36671	First released in Issue 2.	
36672	Issue 6	
36673	The following new requirements on POSIX implementations derive from alignment with the Single UNIX Specification:	
36674		
36675	• The octal mode is supported.	
36676	IEEE Std 1003.1-2001/Cor 2-2004, item XCU/TC2/D6/34 is applied, making a correction to the RATIONALE.	2
36677		2

36678 NAME

36679 unalias — remove alias definitions

36680 SYNOPSIS

36681 UP `unalias alias-name...`

36682 `unalias -a`

36683

36684 DESCRIPTION

36685 The *unalias* utility shall remove the definition for each alias name specified. See Section 2.3.1 (on page 32). The aliases shall be removed from the current shell execution environment; see Section 2.12 (on page 61).

36688 OPTIONS

36689 The *unalias* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section 12.2, Utility Syntax Guidelines.

36691 The following option shall be supported:

36692 **-a** Remove all alias definitions from the current shell execution environment.

36693 OPERANDS

36694 The following operand shall be supported:

36695 *alias-name* The name of an alias to be removed.

36696 STDIN

36697 Not used.

36698 INPUT FILES

36699 None.

36700 ENVIRONMENT VARIABLES

36701 The following environment variables shall affect the execution of *unalias*:

36702 *LANG* Provide a default value for the internationalization variables that are unset or null. (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2, Internationalization Variables for the precedence of internationalization variables used to determine the values of locale categories.)

36706 *LC_ALL* If set to a non-empty string value, override the values of all the other internationalization variables.

36708 *LC_CTYPE* Determine the locale for the interpretation of sequences of bytes of text data as characters (for example, single-byte as opposed to multi-byte characters in arguments).

36711 *LC_MESSAGES*

36712 Determine the locale that should be used to affect the format and contents of diagnostic messages written to standard error.

36714 XSI *NLSPATH* Determine the location of message catalogs for the processing of *LC_MESSAGES*.

36715 ASYNCHRONOUS EVENTS

36716 Default.

36717 STDOUT

36718 Not used.

36719 STDERR

36720 The standard error shall be used only for diagnostic messages.

36721 OUTPUT FILES

36722 None.

36723 EXTENDED DESCRIPTION

36724 None.

36725 EXIT STATUS

36726 The following exit values shall be returned:

36727 0 Successful completion.

36728 >0 One of the *alias-name* operands specified did not represent a valid alias definition, or an
36729 error occurred.

36730 CONSEQUENCES OF ERRORS

36731 Default.

36732 APPLICATION USAGE

36733 Since *unalias* affects the current shell execution environment, it is generally provided as a shell
36734 regular built-in.

36735 EXAMPLES

36736 None.

36737 RATIONALE

36738 The *unalias* description is based on that from historical KornShell implementations. Known
36739 differences exist between that and the C shell. The KornShell version was adopted to be
36740 consistent with all the other KornShell features in this volume of IEEE Std 1003.1-2001, such as
36741 command line editing.

36742 The **-a** option is the equivalent of the *unalias ** form of the C shell and is provided to address
36743 security concerns about unknown aliases entering the environment of a user (or application)
36744 through the allowable implementation-defined predefined alias route or as a result of an *ENV*
36745 file. (Although *unalias* could be used to simplify the “secure” shell script shown in the *command*
36746 rationale, it does not obviate the need to quote all command names. An initial call to *unalias -a*
36747 would have to be quoted in case there was an alias for *unalias*.)

36748 FUTURE DIRECTIONS

36749 None.

36750 SEE ALSO

36751 Chapter 2 (on page 29), *alias*

36752 CHANGE HISTORY

36753 First released in Issue 4.

36754 Issue 6

36755 This utility is marked as part of the User Portability Utilities option.

36756 NAME

36757 **uname** — return system name

36758 SYNOPSIS

36759 **uname** [**-snrvma**]

36760 DESCRIPTION

36761 By default, the **uname** utility shall write the operating system name to standard output. When
36762 options are specified, symbols representing one or more system characteristics shall be written
36763 to the standard output. The format and contents of the symbols are implementation-defined. On
36764 systems conforming to the System Interfaces volume of IEEE Std 1003.1-2001, the symbols
36765 written shall be those supported by the **uname()** function as defined in the System Interfaces
36766 volume of IEEE Std 1003.1-2001.

36767 OPTIONS

36768 The **uname** utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section
36769 12.2, Utility Syntax Guidelines.

36770 The following options shall be supported:

- 36771 **-a** Behave as though all of the options **-mnrv** were specified.
- 36772 **-m** Write the name of the hardware type on which the system is running to standard
36773 output.
- 36774 **-n** Write the name of this node within an implementation-defined communications
36775 network.
- 36776 **-r** Write the current release level of the operating system implementation.
- 36777 **-s** Write the name of the implementation of the operating system.
- 36778 **-v** Write the current version level of this release of the operating system
36779 implementation.

36780 If no options are specified, the **uname** utility shall write the operating system name, as if the **-s**
36781 option had been specified.

36782 OPERANDS

36783 None.

36784 STDIN

36785 Not used.

36786 INPUT FILES

36787 None.

36788 ENVIRONMENT VARIABLES

36789 The following environment variables shall affect the execution of **uname**:

- 36790 **LANG** Provide a default value for the internationalization variables that are unset or null.
36791 (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2,
36792 Internationalization Variables for the precedence of internationalization variables
36793 used to determine the values of locale categories.)
- 36794 **LC_ALL** If set to a non-empty string value, override the values of all the other
36795 internationalization variables.
- 36796 **LC_CTYPE** Determine the locale for the interpretation of sequences of bytes of text data as
36797 characters (for example, single-byte as opposed to multi-byte characters in
36798 arguments).

36799	LC_MESSAGES	Determine the locale that should be used to affect the format and contents of diagnostic messages written to standard error.
36800		
36801		
36802 XSI	NLSPATH	Determine the location of message catalogs for the processing of <i>LC_MESSAGES</i> .
36803	ASYNCHRONOUS EVENTS	
36804		Default.
36805	STDOUT	
36806		By default, the output shall be a single line of the following form:
36807		"%s\n", <sysname>
36808		If the -a option is specified, the output shall be a single line of the following form:
36809		"%s %s %s %s\n", <sysname>, <nodename>, <release>,
36810		<version>, <machine>
36811		Additional implementation-defined symbols may be written; all such symbols shall be written at the end of the line of output before the <newline>.
36812		
36813		If options are specified to select different combinations of the symbols, only those symbols shall be written, in the order shown above for the -a option. If a symbol is not selected for writing, its corresponding trailing <blank>s also shall not be written.
36814		
36815		
36816	STDERR	
36817		The standard error shall be used only for diagnostic messages.
36818	OUTPUT FILES	
36819		None.
36820	EXTENDED DESCRIPTION	
36821		None.
36822	EXIT STATUS	
36823		The following exit values shall be returned:
36824		0 The requested information was successfully written.
36825		>0 An error occurred.
36826	CONSEQUENCES OF ERRORS	
36827		Default.
36828	APPLICATION USAGE	
36829		Note that any of the symbols could include embedded <space>s, which may affect parsing
36830		algorithms if multiple options are selected for output.
36831		The node name is typically a name that the system uses to identify itself for inter-system
36832		communication addressing.
36833	EXAMPLES	
36834		The following command:
36835		uname -sr
36836		writes the operating system name and release level, separated by one or more <blank>s.

36837 RATIONALE

36838 It was suggested that this utility cannot be used portably since the format of the symbols is
36839 implementation-defined. The POSIX.1 working group could not achieve consensus on defining
36840 these formats in the underlying *uname()* function, and there was no expectation that this volume
36841 of IEEE Std 1003.1-2001 would be any more successful. Some applications may still find this
36842 historical utility of value. For example, the symbols could be used for system log entries or for
36843 comparison with operator or user input.

36844 FUTURE DIRECTIONS

36845 None.

36846 SEE ALSO

36847 The System Interfaces volume of IEEE Std 1003.1-2001, *uname()*

36848 CHANGE HISTORY

36849 First released in Issue 2.

36850 NAME

36851 *uncompress* — expand compressed data

36852 SYNOPSIS

36853 XSI **uncompress [-cfv][file...]**

36854

36855 DESCRIPTION

36856 The *uncompress* utility shall restore files to their original state after they have been compressed using the *compress* utility. If no files are specified, the standard input shall be uncompressed to the standard output. If the invoking process has appropriate privileges, the ownership, modes, access time, and modification time of the original file shall be preserved.

36860 This utility shall support the uncompressing of any files produced by the *compress* utility on the same implementation. For files produced by *compress* on other systems, *uncompress* supports 9 to 14-bit compression (see *compress*, **-b**); it is implementation-defined whether values of **-b** greater than 14 are supported.

36864 OPTIONS

36865 The *uncompress* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section 12.2, Utility Syntax Guidelines.

36867 The following options shall be supported:

36868 **-c** Write to standard output; no files are changed.

36869 **-f** Do not prompt for overwriting files. Except when run in the background, if **-f** is not given the user shall be prompted as to whether an existing file should be overwritten. If the standard input is not a terminal and **-f** is not given, *uncompress* shall write a diagnostic message to standard error and exit with a status greater than zero.

36874 **-v** Write messages to standard error concerning the expansion of each file.

36875 OPERANDS

36876 The following operand shall be supported:

36877 *file* A pathname of a file. If *file* already has the **.Z** suffix specified, it shall be used as the input file and the output file shall be named *file* with the **.Z** suffix removed. Otherwise, *file* shall be used as the name of the output file and *file* with the **.Z** suffix appended shall be used as the input file.

36881 STDIN

36882 The standard input shall be used only if no *file* operands are specified, or if a *file* operand is '**-**'.

36883 INPUT FILES

36884 Input files shall be in the format produced by the *compress* utility.

36885 ENVIRONMENT VARIABLES

36886 The following environment variables shall affect the execution of *uncompress*:

36887 **LANG** Provide a default value for the internationalization variables that are unset or null. (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2, Internationalization Variables for the precedence of internationalization variables used to determine the values of locale categories.)

36891 **LC_ALL** If set to a non-empty string value, override the values of all the other internationalization variables.

36893	<i>LC_CTYPE</i>	Determine the locale for the interpretation of sequences of bytes of text data as characters (for example, single-byte as opposed to multi-byte characters in arguments).
36896	<i>LC_MESSAGES</i>	Determine the locale that should be used to affect the format and contents of diagnostic messages written to standard error.
36899	<i>NLSPATH</i>	Determine the location of message catalogs for the processing of <i>LC_MESSAGES</i> .
36900	ASYNCHRONOUS EVENTS	
36901		Default.
36902	STDOUT	
36903		When there are no <i>file</i> operands or the -c option is specified, the uncompressed output is written to standard output.
36905	STDERR	
36906		Prompts shall be written to the standard error output under the conditions specified in the DESCRIPTION and OPTIONS sections. The prompts shall contain the <i>file</i> pathname, but their format is otherwise unspecified. Otherwise, the standard error output shall be used only for diagnostic messages.
36910	OUTPUT FILES	
36911		Output files are the same as the respective input files to <i>compress</i> .
36912	EXTENDED DESCRIPTION	
36913		None.
36914	EXIT STATUS	
36915		The following exit values shall be returned:
36916	0	Successful completion.
36917	>0	An error occurred.
36918	CONSEQUENCES OF ERRORS	
36919		The input file remains unmodified.
36920	APPLICATION USAGE	
36921		The limit of 14 on the <i>compress -b bits</i> argument is to achieve portability to all systems (within the restrictions imposed by the lack of an explicit published file format). Some implementations based on 16-bit architectures cannot support 15 or 16-bit uncompression.
36924	EXAMPLES	
36925		None.
36926	RATIONALE	
36927		None.
36928	FUTURE DIRECTIONS	
36929		None.
36930	SEE ALSO	
36931		<i>compress, zcat</i>
36932	CHANGE HISTORY	
36933		First released in Issue 4.

36934 **Issue 6**

36935

The normative text is reworded to avoid use of the term “must” for application requirements.

36936 NAME

36937 unexpand — convert spaces to tabs

36938 SYNOPSIS

36939 UP unexpand [-a | -t tablist][file...]

36940

36941 DESCRIPTION

36942 The *unexpand* utility shall copy files or standard input to standard output, converting <blank>s at the beginning of each line into the maximum number of <tab>s followed by the minimum number of <space>s needed to fill the same column positions originally filled by the translated <blank>s. By default, tabstops shall be set at every eighth column position. Each <backspace> shall be copied to the output, and shall cause the column position count for tab calculations to be decremented; the count shall never be decremented to a value less than one.

36948 OPTIONS

36949 The *unexpand* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section 12.2, Utility Syntax Guidelines.

36951 The following options shall be supported:

36952 **-a** In addition to translating <blank>s at the beginning of each line, translate all sequences of two or more <blank>s immediately preceding a tab stop to the maximum number of <tab>s followed by the minimum number of <space>s needed to fill the same column positions originally filled by the translated <blank>s.

36957 **-t tablist** Specify the tab stops. The application shall ensure that the *tablist* option-argument is a single argument consisting 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.

36963 The application shall ensure that each tab-stop position *N* is an integer value greater than zero, and the list shall be in strictly ascending order. This is 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. When the **-t** option is not specified, the default shall be the equivalent of specifying **-t 8** (except for the interaction with **-a**, described below).

36969 No <space>-to-<tab> conversions shall occur for characters at positions beyond the last of those specified in a multiple tab-stop list.

36971 When **-t** is specified, the presence or absence of the **-a** option shall be ignored; conversion shall not be limited to the processing of leading <blank>s.

36973 OPERANDS

36974 The following operand shall be supported:

36975 *file* A pathname of a text file to be used as input.

36976 STDIN

36977 See the INPUT FILES section.

36978 INPUT FILES

36979 The input files shall be text files.

36980 ENVIRONMENT VARIABLES

36981 The following environment variables shall affect the execution of *unexpand*:

36982 *LANG* Provide a default value for the internationalization variables that are unset or null.
36983 (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2,
36984 Internationalization Variables for the precedence of internationalization variables
36985 used to determine the values of locale categories.)

36986 *LC_ALL* If set to a non-empty string value, override the values of all the other
36987 internationalization variables.

36988 *LC_CTYPE* Determine the locale for the interpretation of sequences of bytes of text data as
36989 characters (for example, single-byte as opposed to multi-byte characters in
36990 arguments and input files), the processing of <tab>s and <space>s, and for the
36991 determination of the width in column positions each character would occupy on
36992 an output device.

36993 *LC_MESSAGES*

36994 Determine the locale that should be used to affect the format and contents of
36995 diagnostic messages written to standard error.

36996 XSI *NLSPATH* Determine the location of message catalogs for the processing of *LC_MESSAGES*.

36997 ASYNCHRONOUS EVENTS

36998 Default.

36999 STDOUT

37000 The standard output shall be equivalent to the input files with the specified <space>-to-<tab>
37001 conversions.

37002 STDERR

37003 The standard error shall be used only for diagnostic messages.

37004 OUTPUT FILES

37005 None.

37006 EXTENDED DESCRIPTION

37007 None.

37008 EXIT STATUS

37009 The following exit values shall be returned:

37010 0 Successful completion.

37011 >0 An error occurred.

37012 CONSEQUENCES OF ERRORS

37013 Default.

37014 APPLICATION USAGE

37015 One non-intuitive aspect of *unexpand* is its restriction to leading spaces when neither **-a** nor **-t** is specified. Users who always want to convert all spaces in a file can easily alias *unexpand* to use the **-a** or **-t 8** option.

37018 EXAMPLES

37019 None.

37020 RATIONALE

37021 On several occasions, consideration was given to adding a **-t** option to the *unexpand* utility to complement the **-t** in *expand* (see *expand*). The historical intent of *unexpand* was to translate multiple <blank>s into tab stops, where tab stops were a multiple of eight column positions on most UNIX systems. An early proposal omitted **-t** because it seemed outside the scope of the User Portability Utilities option; it was not described in any of the base documents. However, hard-coding tab stops every eight columns was not suitable for the international community and broke historical precedents for some vendors in the FORTRAN community, so **-t** was restored in conjunction with the list of valid extension categories considered by the standard developers.

37029 Thus, *unexpand* is now the logical converse of *expand*.

37030 FUTURE DIRECTIONS

37031 None.

37032 SEE ALSO

37033 *expand*, *tabs*

37034 CHANGE HISTORY

37035 First released in Issue 4.

37036 Issue 6

37037 This utility is marked as part of the User Portability Utilities option.

37038 The definition of the *LC_CTYPE* environment variable is changed to align with the IEEE P1003.2b draft standard.

37040 The normative text is reworded to avoid use of the term “must” for application requirements.

37041 NAME

37042 unget — undo a previous get of an SCCS file (**DEVELOPMENT**)

37043 SYNOPSIS

37044 XSI **unget [-ns][-r SID] file...**

37045

37046 DESCRIPTION

37047 The *unget* utility shall reverse the effect of a *get -e* done prior to creating the intended new delta.

37048 OPTIONS

37049 The *unget* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section 12.2, Utility Syntax Guidelines.

37051 The following options shall be supported:

37052 **-r SID** Uniquely identify which delta is no longer intended. (This would have been specified by *get* as the new delta.) The use of this option is necessary only if two or more outstanding *get* commands for editing on the same SCCS file were done by the same person (login name).

37056 **-s** Suppress the writing to standard output of the intended delta's SID.

37057 **-n** Retain the file that was obtained by *get*, which would normally be removed from the current directory.

37059 OPERANDS

37060 The following operands shall be supported:

37061 *file* A pathname of an existing SCCS file or a directory. If *file* is a directory, the *unget* utility shall behave as though each file in the directory were specified as a named file, except that non-SCCS files (last component of the pathname does not begin with s.) and unreadable files shall be silently ignored.

37065 If exactly one *file* operand appears, and it is ‘-’, the standard input shall be read; each line of the standard input shall be taken to be the name of an SCCS file to be processed. Non-SCCS files and unreadable files shall be silently ignored.

37068 STDIN

37069 The standard input shall be a text file used only when the *file* operand is specified as ‘-’. Each line of the text file shall be interpreted as an SCCS pathname.

37071 INPUT FILES

37072 Any SCCS files processed shall be files of an unspecified format.

37073 ENVIRONMENT VARIABLES

37074 The following environment variables shall affect the execution of *unget*:

37075 *LANG* Provide a default value for the internationalization variables that are unset or null. (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2, Internationalization Variables for the precedence of internationalization variables used to determine the values of locale categories.)

37079 *LC_ALL* If set to a non-empty string value, override the values of all the other internationalization variables.

37081 *LC_CTYPE* Determine the locale for the interpretation of sequences of bytes of text data as characters (for example, single-byte as opposed to multi-byte characters in arguments and input files).

- 37084 ***LC_MESSAGES***
37085 Determine the locale that should be used to affect the format and contents of
37086 diagnostic messages written to standard error.
- 37087 **NLSPATH** Determine the location of message catalogs for the processing of *LC_MESSAGES*.
- 37088 **ASYNCHRONOUS EVENTS**
- 37089 Default.
- 37090 **STDOUT**
- 37091 The standard output shall consist of a line for each file, in the following format:
37092 "%s\n", <SID removed from file>
- 37093 If there is more than one named file or if a directory or standard input is named, each pathname
37094 shall be written before each of the preceding lines:
- 37095 "\n%s:\n", <pathname>
- 37096 **STDERR**
- 37097 The standard error shall be used only for diagnostic messages.
- 37098 **OUTPUT FILES**
- 37099 Any SCCS files updated shall be files of an unspecified format. During processing of a *file*, a
37100 locking *z-file*, as described in *get*, and a *q-file* (a working copy of the *p-file*), may be created and
37101 deleted. The *p-file* and *g-file*, as described in *get*, shall be deleted.
- 37102 **EXTENDED DESCRIPTION**
- 37103 None.
- 37104 **EXIT STATUS**
- 37105 The following exit values shall be returned:
37106 0 Successful completion.
37107 >0 An error occurred.
- 37108 **CONSEQUENCES OF ERRORS**
- 37109 Default.
- 37110 **APPLICATION USAGE**
- 37111 None.
- 37112 **EXAMPLES**
- 37113 None.
- 37114 **RATIONALE**
- 37115 None.
- 37116 **FUTURE DIRECTIONS**
- 37117 None.
- 37118 **SEE ALSO**
- 37119 *delta, get, sact*
- 37120 **CHANGE HISTORY**
- 37121 First released in Issue 2.
- 37122 **Issue 6**
- 37123 The normative text is reworded to avoid use of the term “must” for application requirements.

37124 NAME

37125 *uniq* — report or filter out repeated lines in a file

37126 SYNOPSIS

37127 *uniq* [-c|-d|-u][-f *fields*][-s *char*][*input_file* [*output_file*]]

37128 DESCRIPTION

37129 The *uniq* utility shall read an input file comparing adjacent lines, and write one copy of each
37130 input line on the output. The second and succeeding copies of repeated adjacent input lines shall
37131 not be written.

37132 Repeated lines in the input shall not be detected if they are not adjacent.

37133 OPTIONS

37134 The *uniq* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section
37135 12.2, Utility Syntax Guidelines.

37136 The following options shall be supported:

37137 **-c** Precede each output line with a count of the number of times the line occurred in
37138 the input.

37139 **-d** Suppress the writing of lines that are not repeated in the input.

37140 **-f** *fields* Ignore the first *fields* fields on each input line when doing comparisons, where
37141 *fields* is a positive decimal integer. A field is the maximal string matched by the
37142 basic regular expression:

37143 [[:blank:]]*[^[:blank:]]*

37144 If the *fields* option-argument specifies more fields than appear on an input line, a
37145 null string shall be used for comparison.

37146 **-s** *chars* Ignore the first *chars* characters when doing comparisons, where *chars* shall be a
37147 positive decimal integer. If specified in conjunction with the **-f** option, the first
37148 *chars* characters after the first *fields* fields shall be ignored. If the *chars* option-
37149 argument specifies more characters than remain on an input line, a null string shall
37150 be used for comparison.

37151 **-u** Suppress the writing of lines that are repeated in the input.

37152 OPERANDS

37153 The following operands shall be supported:

37154 *input_file* A pathname of the input file. If the *input_file* operand is not specified, or if the
37155 *input_file* is ‘-’, the standard input shall be used.

37156 *output_file* A pathname of the output file. If the *output_file* operand is not specified, the
37157 standard output shall be used. The results are unspecified if the file named by
37158 *output_file* is the file named by *input_file*.

37159 STDIN

37160 The standard input shall be used only if no *input_file* operand is specified or if *input_file* is ‘-’.
37161 See the INPUT FILES section.

37162 INPUT FILES

37163 The input file shall be a text file.

37164 ENVIRONMENT VARIABLES

37165 The following environment variables shall affect the execution of *uniq*:

37166 ***LANG*** Provide a default value for the internationalization variables that are unset or null.
 37167 (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2,
 37168 Internationalization Variables for the precedence of internationalization variables
 37169 used to determine the values of locale categories.)

37170 ***LC_ALL*** If set to a non-empty string value, override the values of all the other
 37171 internationalization variables.

37172 ***LC_COLLATE*** 1
 37173 Determine the locale for ordering rules. 1

37174 ***LC_CTYPE*** Determine the locale for the interpretation of sequences of bytes of text data as
 37175 characters (for example, single-byte as opposed to multi-byte characters in
 37176 arguments and input files) and which characters constitute a <blank> in the
 37177 current locale.

37178 ***LC_MESSAGES***
 37179 Determine the locale that should be used to affect the format and contents of
 37180 diagnostic messages written to standard error.

37181 **XSI** ***NLSPATH*** Determine the location of message catalogs for the processing of *LC_MESSAGES*.

37182 ASYNCHRONOUS EVENTS

37183 Default.

37184 STDOUT

37185 The standard output shall be used only if no *output_file* operand is specified. See the OUTPUT
 37186 FILES section.

37187 STDERR

37188 The standard error shall be used only for diagnostic messages.

37189 OUTPUT FILES

37190 If the **-c** option is specified, the output file shall be empty or each line shall be of the form: 1

37191 " %d %s ", <number of duplicates>, <line>

37192 otherwise, the output file shall be empty or each line shall be of the form: 1

37193 " %s ", <line>

37194 EXTENDED DESCRIPTION

37195 None.

37196 EXIT STATUS

37197 The following exit values shall be returned:

37198 0 The utility executed successfully.

37199 >0 An error occurred.

37200 CONSEQUENCES OF ERRORS

37201 Default.

37202 APPLICATION USAGE

37203 The *sort* utility can be used to cause repeated lines to be adjacent in the input file.

37204 EXAMPLES

37205 The following input file data (but flushed left) was used for a test series on *uniq*:

```
37206 #01 foo0 bar0 fool bar1
37207 #02 bar0 fool bar1 fool
37208 #03 foo0 bar0 fool bar1
37209 #04
37210 #05 foo0 bar0 fool bar1
37211 #06 foo0 bar0 fool bar1
37212 #07 bar0 fool bar1 foo0
```

37213 What follows is a series of test invocations of the *uniq* utility that use a mixture of *uniq* options
37214 against the input file data. These tests verify the meaning of *adjacent*. The *uniq* utility views the
37215 input data as a sequence of strings delimited by '\n'. Accordingly, for the *field*th member of
37216 the sequence, *uniq* interprets unique or repeated adjacent lines strictly relative to the *fields+1*th
37217 member.

- 37218 1. This first example tests the line counting option, comparing each line of the input file data
37219 starting from the second field:

```
37220 uniq -c -f 1 uniq_0I.t
37221      1 #01 foo0 bar0 fool bar1
37222      1 #02 bar0 fool bar1 foo0
37223      1 #03 foo0 bar0 fool bar1
37224      1 #04
37225      2 #05 foo0 bar0 fool bar1
37226      1 #07 bar0 fool bar1 foo0
```

37227 The number '2', prefixing the fifth line of output, signifies that the *uniq* utility detected a
37228 pair of repeated lines. Given the input data, this can only be true when *uniq* is run using
37229 the **-f 1** option (which shall cause *uniq* to ignore the first field on each input line).

- 37230 2. The second example tests the option to suppress unique lines, comparing each line of the
37231 input file data starting from the second field:

```
37232 uniq -d -f 1 uniq_0I.t
37233 #05 foo0 bar0 fool bar1
```

- 37234 3. This test suppresses repeated lines, comparing each line of the input file data starting from
37235 the second field:

```
37236 uniq -u -f 1 uniq_0I.t
37237 #01 foo0 bar0 fool bar1
37238 #02 bar0 fool bar1 fool
37239 #03 foo0 bar0 fool bar1
37240 #04
37241 #07 bar0 fool bar1 foo0
```

- 37242 4. This suppresses unique lines, comparing each line of the input file data starting from the
37243 third character:

```
37244 uniq -d -s 2 uniq_0I.t
```

37245 In the last example, the *uniq* utility found no input matching the above criteria.

37246 RATIONALE

37247 Some historical implementations have limited lines to be 1080 bytes in length, which does not
37248 meet the implied {LINE_MAX} limit.

37249 FUTURE DIRECTIONS

37250 None.

37251 SEE ALSO

37252 *comm, sort*

37253 CHANGE HISTORY

37254 First released in Issue 2.

37255 Issue 6

37256 The obsolescent SYNOPSIS and associated text are removed.

37257 The normative text is reworded to avoid use of the term “must” for application requirements.

37258 IEEE Std 1003.1-2001/Cor 1-2002, item XCU/TC1/D6/40 is applied, adding *LC_COLLATE* to the 1
37259 ENVIRONMENT VARIABLES section, and changing “the application shall ensure that” in the 1
37260 OUTPUT FILES section. 1

37261 NAME

37262 unlink — call the *unlink()* function

37263 SYNOPSIS

37264 XSI `unlink file`

37265

37266 DESCRIPTION

37267 The *unlink* utility shall perform the function call:

37268 `unlink(file);`

37269 A user may need appropriate privilege to invoke the *unlink* utility.

37270 OPTIONS

37271 None.

37272 OPERANDS

37273 The following operands shall be supported:

37274 *file* The pathname of an existing file.

37275 STDIN

37276 Not used.

37277 INPUT FILES

37278 Not used.

37279 ENVIRONMENT VARIABLES

37280 The following environment variables shall affect the execution of *unlink*:

37281 *LANG* Provide a default value for the internationalization variables that are unset or null.
37282 (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2,
37283 Internationalization Variables for the precedence of internationalization variables
37284 used to determine the values of locale categories.)

37285 *LC_ALL* If set to a non-empty string value, override the values of all the other
37286 internationalization variables.

37287 *LC_CTYPE* Determine the locale for the interpretation of sequences of bytes of text data as
37288 characters (for example, single-byte as opposed to multi-byte characters in
37289 arguments).

37290 *LC_MESSAGES*

37291 Determine the locale that should be used to affect the format and contents of
37292 diagnostic messages written to standard error.

37293 *NLSPATH* Determine the location of message catalogs for the processing of *LC_MESSAGES*.

37294 ASYNCHRONOUS EVENTS

37295 Default.

37296 STDOUT

37297 None.

37298 STDERR

37299 The standard error shall be used only for diagnostic messages.

37300 OUTPUT FILES

37301 None.

37302 EXTENDED DESCRIPTION

37303 None.

37304 EXIT STATUS

37305 The following exit values shall be returned:

37306 0 Successful completion.

37307 >0 An error occurred.

37308 CONSEQUENCES OF ERRORS

37309 Default.

37310 APPLICATION USAGE

37311 None.

37312 EXAMPLES

37313 None.

37314 RATIONALE

37315 None.

37316 FUTURE DIRECTIONS

37317 None.

37318 SEE ALSO

37319 *link*, *rm*, the System Interfaces volume of IEEE Std 1003.1-2001, *unlink()*

37320 CHANGE HISTORY

37321 First released in Issue 5.

37322 NAME

37323 uucp — system-to-system copy

37324 SYNOPSIS

37325 XSI uucp [**-cCdfjmr**] [**-n user**] *source-file...* *destination-file*

37326

37327 DESCRIPTION

37328 The *uucp* utility shall copy files named by the *source-file* argument to the *destination-file* argument. The files named can be on local or remote systems.

37330 The *uucp* utility cannot guarantee support for all character encodings in all circumstances. For
37331 example, transmission data may be restricted to 7 bits by the underlying network, 8-bit data and
37332 filenames need not be portable to non-internationalized systems, and so on. Under these
37333 circumstances, it is recommended that only characters defined in the ISO/IEC 646:1991
37334 standard International Reference Version (equivalent to ASCII) 7-bit range of characters be used,
37335 and that only characters defined in the portable filename character set be used for naming files.
37336 The protocol for transfer of files is unspecified by IEEE Std 1003.1-2001.

37337 Typical implementations of this utility require a communications line configured to use the Base
37338 Definitions volume of IEEE Std 1003.1-2001, Chapter 11, General Terminal Interface, but other
37339 communications means may be used. On systems where there are no available communications
37340 means (either temporarily or permanently), this utility shall write an error message describing
37341 the problem and exit with a non-zero exit status.

37342 OPTIONS

37343 The *uucp* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section
37344 12.2, Utility Syntax Guidelines.

37345 The following options shall be supported:

37346 **-c** Do not copy local file to the spool directory for transfer to the remote machine
37347 (default).

37348 **-C** Force the copy of local files to the spool directory for transfer.

37349 **-d** Make all necessary directories for the file copy (default).

37350 **-f** Do not make intermediate directories for the file copy.

37351 **-j** Write the job identification string to standard output. This job identification can be
37352 used by *ustat* to obtain the status or terminate a job.

37353 **-m** Send mail to the requester when the copy is completed.

37354 **-n user** Notify *user* on the remote system that a file was sent.

37355 **-r** Do not start the file transfer; just queue the job.

37356 OPERANDS

37357 The following operands shall be supported:

37358 *destination-file, source-file*

37359 A pathname of a file to be copied to, or from, respectively. Either name can be a
37360 pathname on the local machine, or can have the form:

37361 *system-name ! pathname*

37362 where *system-name* is taken from a list of system names that *uucp* knows about.
37363 The destination *system-name* can also be a list of names such as:

37364 `system-name ! system-name ! . . . ! system-name ! pathname`

37365 in which case, an attempt is made to send the file via the specified route to the destination. Care should be taken to ensure that intermediate nodes in the route are willing to forward information.

37368 The shell pattern matching notation characters '?', '*', and "[. . .]" appearing in *pathname* shall be expanded on the appropriate system.

37370 Pathnames can be one of:

37371 1. An absolute pathname.

37372 2. A pathname preceded by `~user` where *user* is a login name on the specified system and is replaced by that user's login directory. Note that if an invalid login is specified, the default is to the public directory (called *PUBDIR*; the actual location of *PUBDIR* is implementation-defined).

37376 3. A pathname preceded by `~/destination` where *destination* is appended to *PUBDIR*.

37378 **Note:** This destination is treated as a filename unless more than one file is being transferred by this request or the destination is already a directory. To ensure that it is a directory, follow the destination with a '/'. For example, `~/dan/` as the destination makes the directory *PUBDIR/dan* if it does not exist and puts the requested files in that directory.

37383 4. Anything else shall be prefixed by the current directory.

37384 If the result is an erroneous pathname for the remote system, the copy shall fail. If 37385 the *destination-file* is a directory, the last part of the *source-file* name shall be used.

37386 The read, write, and execute permissions given by *uucp* are implementation-defined.

37388 **STDIN**

37389 Not used.

37390 **INPUT FILES**

37391 The files to be copied are regular files.

37392 **ENVIRONMENT VARIABLES**

37393 The following environment variables shall affect the execution of *uucp*:

37394 **LANG** Provide a default value for the internationalization variables that are unset or null. 37395 (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2, 37396 Internationalization Variables for the precedence of internationalization variables 37397 used to determine the values of locale categories.)

37398 **LC_ALL** If set to a non-empty string value, override the values of all the other 37399 internationalization variables.

37400 **LC_COLLATE** Determine the locale for the behavior of ranges, equivalence classes, and multi-character collating elements within bracketed filename patterns.

37403 **LC_CTYPE** Determine the locale for the interpretation of sequences of bytes of text data as 37404 characters (for example, single-byte as opposed to multi-byte characters in 37405 arguments and input files) and the behavior of character classes within bracketed 37406 filename patterns (for example, "' [[:lower:]]*'").

- 37407 ***LC_MESSAGES***
37408 Determine the locale that should be used to affect the format and contents of
37409 diagnostic messages written to standard error, and informative messages written
37410 to standard output.
- 37411 **NLSPATH** Determine the location of message catalogs for the processing of *LC_MESSAGES*.
- 37412 **ASYNCHRONOUS EVENTS**
37413 Default.
- 37414 **STDOUT**
37415 Not used.
- 37416 **STDERR**
37417 The standard error shall be used only for diagnostic messages.
- 37418 **OUTPUT FILES**
37419 The output files (which may be on other systems) are copies of the input files.
37420 If **-m** is used, mail files are modified.
- 37421 **EXTENDED DESCRIPTION**
37422 None.
- 37423 **EXIT STATUS**
37424 The following exit values shall be returned:
37425 0 Successful completion.
37426 >0 An error occurred.
- 37427 **CONSEQUENCES OF ERRORS**
37428 Default.
- 37429 **APPLICATION USAGE**
37430 The domain of remotely accessible files can (and for obvious security reasons usually should) be
37431 severely restricted.

37432 Note that the '!' character in addresses has to be escaped when using *csh* as a command
37433 interpreter because of its history substitution syntax. For *ksh* and *sh* the escape is not necessary,
37434 but may be used.

37435 As noted above, shell metacharacters appearing in pathnames are expanded on the appropriate
37436 system. On an internationalized system, this is done under the control of local settings of
37437 *LC_COLLATE* and *LC_CTYPE*. Thus, care should be taken when using bracketed filename
37438 patterns, as collation and typing rules may vary from one system to another. Also be aware that
37439 certain types of expression (that is, equivalence classes, character classes, and collating symbols)
37440 need not be supported on non-internationalized systems.
- 37441 **EXAMPLES**
37442 None.
- 37443 **RATIONALE**
37444 None.
- 37445 **FUTURE DIRECTIONS**
37446 None.

37447 SEE ALSO

37448 *mailx, uuencode, uustat, uux*

37449 CHANGE HISTORY

37450 First released in Issue 2.

37451 Issue 6

37452 The *LC_TIME* and *TZ* entries are removed from the ENVIRONMENT VARIABLES section.

37453 The UN margin codes and associated shading are removed from the **-C**, **-f**, **-j**, **-n**, and **-r** options in response to The Open Group Base Resolution bwg2001-003.
37454

37455 NAME

37456 uudecode — decode a binary file

37457 SYNOPSIS

37458 UP **uudecode [-o outfile][file]**

37459

37460 DESCRIPTION

37461 The *uudecode* utility shall read a file, or standard input if no file is specified, that includes data
37462 created by the *uuencode* utility. The *uudecode* utility shall scan the input file, searching for data
37463 compatible with one of the formats specified in *uuencode*, and attempt to create or overwrite the
37464 file described by the data (or overridden by the **-o** option). The pathname shall be contained in
37465 the data or specified by the **-o** option. The file access permission bits and contents for the file to
37466 be produced shall be contained in that data. The mode bits of the created file (other than
37467 standard output) shall be set from the file access permission bits contained in the data; that is,
37468 other attributes of the mode, including the file mode creation mask (see *umask*), shall not affect
37469 the file being produced. If either of the *op* characters '+' and '-' (see *chmod*) are specified in
37470 symbolic mode, the initial mode on which those operations are based is unspecified.

2
2

37471 If the pathname of the file to be produced exists, and the user does not have write permission on
37472 that file, *uudecode* shall terminate with an error. If the pathname of the file to be produced exists,
37473 and the user has write permission on that file, the existing file shall be overwritten.

37474 If the input data was produced by *uuencode* on a system with a different number of bits per byte
37475 than on the target system, the results of *uudecode* are unspecified.

37476 OPTIONS

37477 The *uudecode* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001,
37478 Section 12.2, Utility Syntax Guidelines.

37479 The following option shall be supported by the implementation:

37480 **-o outfile** A pathname of a file that shall be used instead of any pathname contained in the
37481 input data. Specifying an *outfile* option-argument of */dev/stdout* shall indicate
37482 standard output.

37483 OPERANDS

37484 The following operand shall be supported:

37485 *file* The pathname of a file containing the output of *uuencode*.

37486 STDIN

37487 See the INPUT FILES section.

37488 INPUT FILES

37489 The input files shall be files containing the output of *uuencode*.

37490 ENVIRONMENT VARIABLES

37491 The following environment variables shall affect the execution of *uudecode*:

37492 *LANG* Provide a default value for the internationalization variables that are unset or null.
37493 (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2,
37494 Internationalization Variables for the precedence of internationalization variables
37495 used to determine the values of locale categories.)

37496 *LC_ALL* If set to a non-empty string value, override the values of all the other
37497 internationalization variables.

37498 *LC_CTYPE* Determine the locale for the interpretation of sequences of bytes of text data as
37499 characters (for example, single-byte as opposed to multi-byte characters in

37500		arguments and input files).
37501	LC_MESSAGES	Determine the locale that should be used to affect the format and contents of diagnostic messages written to standard error.
37502		
37503		
37504 XSI	NLSPATH	Determine the location of message catalogs for the processing of <i>LC_MESSAGES</i> .
37505	ASYNCHRONOUS EVENTS	
37506		Default.
37507	STDOUT	
37508		If the file data header encoded by <i>uuencode</i> is – or /dev/stdout, or the –o /dev/stdout option overrides the file data, the standard output shall be in the same format as the file originally encoded by <i>uuencode</i> . Otherwise, the standard output shall not be used.
37509		
37510		
37511	STDERR	
37512		The standard error shall be used only for diagnostic messages.
37513	OUTPUT FILES	
37514		The output file shall be in the same format as the file originally encoded by <i>uuencode</i> .
37515	EXTENDED DESCRIPTION	
37516		None.
37517	EXIT STATUS	
37518		The following exit values shall be returned:
37519	0	Successful completion.
37520	>0	An error occurred.
37521	CONSEQUENCES OF ERRORS	
37522		Default.
37523	APPLICATION USAGE	
37524		The user who is invoking <i>uudecode</i> must have write permission on any file being created.
37525		
37526		
37527		
37528		
		The output of <i>uuencode</i> is essentially an encoded bit stream that is not cognizant of byte boundaries. It is possible that a 9-bit byte target machine can process input from an 8-bit source, if it is aware of the requirement, but the reverse is unlikely to be satisfying. Of course, the only data that is meaningful for such a transfer between architectures is generally character data.
37529	EXAMPLES	
37530		None.
37531	RATIONALE	
37532		Input files are not necessarily text files, as stated by an early proposal. Although the <i>uuencode</i> output is a text file, that output could have been wrapped within another file or mail message that is not a text file.
37533		
37534		
37535		
37536		The –o option is not historical practice, but was added at the request of WG15 so that the user could override the target pathname without having to edit the input data itself.
37537		
37538		
37539		
37540		
37541		
		In early drafts, the [–o <i>outfile</i>] option-argument allowed the use of – to mean standard output. The symbol – has only been used previously in IEEE Std 1003.1-2001 as a standard input indicator. The developers of the standard did not wish to overload the meaning of – in this manner. The /dev/stdout concept exists on most modern systems. The /dev/stdout syntax does not refer to a new special file. It is just a magic cookie to specify standard output.

37542 FUTURE DIRECTIONS

37543 None.

37544 SEE ALSO

37545 *chmod, umask, uuencode*

2

37546 CHANGE HISTORY

37547 First released in Issue 4.

37548 Issue 6

37549 This utility is marked as part of the User Portability Utilities option.

37550 The **-o** *outfile* option is added, as specified in the IEEE P1003.2b draft standard.

37551 The normative text is reworded to avoid use of the term “must” for application requirements.

37552 IEEE Std 1003.1-2001/Cor 2-2004, item XCU/TC2/D6/35 is applied, clarifying in the 2

37553 DESCRIPTION that the initial mode used if either of the *op* characters is '+' or '-' is 2

37554 unspecified. 2

37555 NAME

37556 *uuencode* — encode a binary file

37557 SYNOPSIS

37558 UP *uuencode [-m][file] decode.pathname*

37559

37560 DESCRIPTION

37561 The *uuencode* utility shall write an encoded version of the named input file, or standard input if no *file* is specified, to standard output. The output shall be encoded using one of the algorithms described in the STDOUT section and shall include the file access permission bits (in *chmod* octal or symbolic notation) of the input file and the *decode.pathname*, for re-creation of the file on another system that conforms to this volume of IEEE Std 1003.1-2001.

37566 OPTIONS

37567 The *uuencode* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section 12.2, Utility Syntax Guidelines.

37569 The following option shall be supported by the implementation:

37570 **-m** Encode the output using the MIME Base64 algorithm described in STDOUT. If **-m** is not specified, the historical algorithm described in STDOUT shall be used.

37572 OPERANDS

37573 The following operands shall be supported:

37574 *decode.pathname*

37575 The pathname of the file into which the *uudecode* utility shall place the decoded file. Specifying a *decode.pathname* operand of */dev/stdout* shall indicate that *uudecode* is to use standard output. If there are characters in *decode.pathname* that are not in the portable filename character set the results are unspecified.

37579 *file* A pathname of the file to be encoded.

37580 STDIN

37581 See the INPUT FILES section.

37582 INPUT FILES

37583 Input files can be files of any type.

37584 ENVIRONMENT VARIABLES

37585 The following environment variables shall affect the execution of *uuencode*:

37586 *LANG* Provide a default value for the internationalization variables that are unset or null. (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2, Internationalization Variables for the precedence of internationalization variables used to determine the values of locale categories.)

37590 *LC_ALL* If set to a non-empty string value, override the values of all the other internationalization variables.

37592 *LC_CTYPE* Determine the locale for the interpretation of sequences of bytes of text data as characters (for example, single-byte as opposed to multi-byte characters in arguments and input files).

37595 *LC_MESSAGES*

37596 Determine the locale that should be used to affect the format and contents of diagnostic messages written to standard error.

37598 XSI **NLSPATH** Determine the location of message catalogs for the processing of *LC_MESSAGES*.

37599 **ASYNCHRONOUS EVENTS**

37600 Default.

37601 **STDOUT**

37602 **uuencode Base64 Algorithm**

37603 The standard output shall be a text file (encoded in the character set of the current locale) that
37604 begins with the line:

37605 "begin-base64Δ%sΔ%s\n" , <mode> , <decode_pathname>

37606 and ends with the line:

37607 "====\n"

37608 In both cases, the lines shall have no preceding or trailing <blank>s.

37609 The encoding process represents 24-bit groups of input bits as output strings of four encoded
37610 characters. Proceeding from left to right, a 24-bit input group shall be formed by concatenating
37611 three 8-bit input groups. Each 24-bit input group then shall be treated as four concatenated 6-bit
37612 groups, each of which shall be translated into a single digit in the Base64 alphabet. When
37613 encoding a bit stream via the Base64 encoding, the bit stream shall be presumed to be ordered
37614 with the most-significant bit first. That is, the first bit in the stream shall be the high-order bit in
37615 the first byte, and the eighth bit shall be the low-order bit in the first byte, and so on. Each 6-bit
37616 group is used as an index into an array of 64 printable characters, as shown in Table 4-21.

37617 **Table 4-21 uuencode Base64 Values**

Value	Encoding	Value	Encoding	Value	Encoding	Value	Encoding
0	A	17	R	34	i	51	z
1	B	18	S	35	j	52	0
2	C	19	T	36	k	53	1
3	D	20	U	37	l	54	2
4	E	21	V	38	m	55	3
5	F	22	W	39	n	56	4
6	G	23	X	40	o	57	5
7	H	24	Y	41	p	58	6
8	I	25	Z	42	q	59	7
9	J	26	a	43	r	60	8
10	K	27	b	44	s	61	9
11	L	28	c	45	t	62	+
12	M	29	d	46	u	63	/
13	N	30	e	47	v		
14	O	31	f	48	w	(pad)	=
15	P	32	g	49	x		
16	Q	33	h	50	y		

37636 The character referenced by the index shall be placed in the output string.

37637 The output stream (encoded bytes) shall be represented in lines of no more than 76 characters
37638 each. All line breaks or other characters not found in the table shall be ignored by decoding
37639 software (see *uudecode*).

37640 Special processing shall be performed if fewer than 24 bits are available at the end of a message
37641 or encapsulated part of a message. A full encoding quantum shall always be completed at the

37642 end of a message. When fewer than 24 input bits are available in an input group, zero bits shall
 37643 be added (on the right) to form an integral number of 6-bit groups. Output character positions
 37644 that are not required to represent actual input data shall be set to the character '='. Since all
 37645 Base64 input is an integral number of octets, only the following cases can arise:

- 37646 1. The final quantum of encoding input is an integral multiple of 24 bits; here, the final unit of
 37647 encoded output shall be an integral multiple of 4 characters with no '=' padding.
- 37648 2. The final quantum of encoding input is exactly 16 bits; here, the final unit of encoded
 37649 output shall be three characters followed by one '=' padding character.
- 37650 3. The final quantum of encoding input is exactly 8 bits; here, the final unit of encoded output
 37651 shall be two characters followed by two '=' padding characters.

37652 A terminating "====" evaluates to nothing and denotes the end of the encoded data.

37653 uuencode Historical Algorithm

37654 The standard output shall be a text file (encoded in the character set of the current locale) that
 37655 begins with the line:

37656 "beginΔ%sΔ%s\n" <mode>, <decode_pathname>

37657 and ends with the line:

37658 "end\n"

37659 In both cases, the lines shall have no preceding or trailing <blank>s.

37660 The algorithm that shall be used for lines in between **begin** and **end** takes three octets as input
 37661 and writes four characters of output by splitting the input at six-bit intervals into four octets,
 37662 containing data in the lower six bits only. These octets shall be converted to characters by adding
 37663 a value of 0x20 to each octet, so that each octet is in the range [0x20,0x5f], and then it shall be
 37664 assumed to represent a printable character in the ISO/IEC 646:1991 standard encoded character
 37665 set. It then shall be translated into the corresponding character codes for the codeset in use in the
 37666 current locale. (For example, the octet 0x41, representing 'A', would be translated to 'A' in the
 37667 current codeset, such as 0xc1 if it were EBCDIC.)

37668 Where the bits of two octets are combined, the least significant bits of the first octet shall be
 37669 shifted left and combined with the most significant bits of the second octet shifted right. Thus
 3770 the three octets *A*, *B*, *C* shall be converted into the four octets:

```
37671 0x20 + (( A >> 2 ) & 0x3F)
37672 0x20 + (((A << 4) | ((B >> 4) & 0xF)) & 0x3F)
37673 0x20 + (((B << 2) | ((C >> 6) & 0x3)) & 0x3F)
37674 0x20 + (( C ) & 0x3F)
```

37675 These octets then shall be translated into the local character set.

37676 Each encoded line contains a length character, equal to the number of characters to be decoded
 37677 plus 0x20 translated to the local character set as described above, followed by the encoded
 37678 characters. The maximum number of octets to be encoded on each line shall be 45.

37679 STDRERR

37680 The standard error shall be used only for diagnostic messages.

37681 OUTPUT FILES

37682 None.

37683 EXTENDED DESCRIPTION

37684 None.

37685 EXIT STATUS

37686 The following exit values shall be returned:

37687 0 Successful completion.

37688 >0 An error occurred.

37689 CONSEQUENCES OF ERRORS

37690 Default.

37691 APPLICATION USAGE

37692 The file is expanded by 35 percent (each three octets become four, plus control information) causing it to take longer to transmit.

37693 Since this utility is intended to create files to be used for data interchange between systems with possibly different codesets, and to represent binary data as a text file, the ISO/IEC 646:1991 standard was chosen for a midpoint in the algorithm as a known reference point. The output from *uuencode* is a text file on the local system. If the output were in the ISO/IEC 646:1991 standard codeset, it might not be a text file (at least because the <newline>s might not match), and the goal of creating a text file would be defeated. If this text file was then carried to another machine with the same codeset, it would be perfectly compatible with that system's *uudecode*. If it was transmitted over a mail system or sent to a machine with a different codeset, it is assumed that, as for every other text file, some translation mechanism would convert it (by the time it reached a user on the other system) into an appropriate codeset. This translation only makes sense from the local codeset, not if the file has been put into a ISO/IEC 646:1991 standard representation first. Similarly, files processed by *uuencode* can be placed in *pax* archives, intermixed with other text files in the same codeset.

37707 EXAMPLES

37708 None.

37709 RATIONALE

37710 A new algorithm was added at the request of the international community to parallel work in RFC 2045 (MIME). As with the historical *uuencode* format, the Base64 Content-Transfer-Encoding is designed to represent arbitrary sequences of octets in a form that is not humanly readable. A 65-character subset of the ISO/IEC 646:1991 standard is used, enabling 6 bits to be represented per printable character. (The extra 65th character, '=' , is used to signify a special processing function.)

37716 This subset has the important property that it is represented identically in all versions of the ISO/IEC 646:1991 standard, including US ASCII, and all characters in the subset are also represented identically in all versions of EBCDIC. The historical *uuencode* algorithm does not share this property, which is the reason that a second algorithm was added to the ISO POSIX-2 standard.

37721 The string "====" was used for the termination instead of the end used in the original format because the latter is a string that could be valid encoded input.

37723 In an early draft, the **-m** option was named **-b** (for Base64), but it was renamed to reflect its relationship to the RFC 2045. A **-u** was also present to invoke the default algorithm, but since this was not historical practice, it was omitted as being unnecessary.

37726 See the RATIONALE section in *uudecode* for the derivation of the **/dev/stdout** symbol.

37727 FUTURE DIRECTIONS

37728 None.

37729 SEE ALSO

37730 *chmod, mailx, uudecode*

37731 CHANGE HISTORY

37732 First released in Issue 4.

37733 Issue 6

37734 This utility is marked as part of the User Portability Utilities option.

37735 The Base64 algorithm and the ability to output to **/dev/stdout** are added as specified in the IEEE P1003.2b draft standard.
37736

37737 NAME

37738 *uustat* — uucp status inquiry and job control

37739 SYNOPSIS

37740 XSI **uustat [-q | -k *jobid* | -r *jobid*]**

37741 **uustat [-s *system*] [-u *user*]**

37742

37743 DESCRIPTION

37744 The *uustat* utility shall display the status of, or cancel, previously specified *uucp* requests, or
37745 provide general status on *uucp* connections to other systems.

37746 When no options are given, *uustat* shall write to standard output the status of all *uucp* requests
37747 issued by the current user.

37748 Typical implementations of this utility require a communications line configured to use the Base
37749 Definitions volume of IEEE Std 1003.1-2001, Chapter 11, General Terminal Interface, but other
37750 communications means may be used. On systems where there are no available communications
37751 means (either temporarily or permanently), this utility shall write an error message describing
37752 the problem and exit with a non-zero exit status.

37753 OPTIONS

37754 The *uustat* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section
37755 12.2, Utility Syntax Guidelines.

37756 The following options shall be supported:

37757 **-q** Write the jobs queued for each machine.

37758 **-k *jobid*** Kill the *uucp* request whose job identification is *jobid*. The application shall ensure
37759 that the killed *uucp* request belongs to the person invoking *uustat* unless that user
37760 has appropriate privileges.

37761 **-r *jobid*** Rejuvenate *jobid*. The files associated with *jobid* are touched so that their
37762 modification time is set to the current time. This prevents the cleanup program
37763 from deleting the job until the jobs modification time reaches the limit imposed by
37764 the program.

37765 **-s *system*** Write the status of all *uucp* requests for remote system *system*.

37766 **-u *user*** Write the status of all *uucp* requests issued by *user*.

37767 OPERANDS

37768 None.

37769 STDIN

37770 Not used.

37771 INPUT FILES

37772 None.

37773 ENVIRONMENT VARIABLES

37774 The following environment variables shall affect the execution of *uustat*:

37775 **LANG** Provide a default value for the internationalization variables that are unset or null.
37776 (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2,
37777 Internationalization Variables for the precedence of internationalization variables
37778 used to determine the values of locale categories.)

37779	<i>LC_ALL</i>	If set to a non-empty string value, override the values of all the other internationalization variables.
37781	<i>LC_CTYPE</i>	Determine the locale for the interpretation of sequences of bytes of text data as characters (for example, single-byte as opposed to multi-byte characters in arguments).
37784	<i>LC_MESSAGES</i>	Determine the locale that should be used to affect the format and contents of diagnostic messages written to standard error, and informative messages written to standard output.
37788	<i>NLSPATH</i>	Determine the location of message catalogs for the processing of <i>LC_MESSAGES</i> .
37789	ASYNCHRONOUS EVENTS	
37790		Default.
37791	STDOUT	
37792		The standard output shall consist of information about each job selected, in an unspecified format. The information shall include at least the job ID, the user ID or name, and the remote system name.
37795	STDERR	
37796		The standard error shall be used only for diagnostic messages.
37797	OUTPUT FILES	
37798		None.
37799	EXTENDED DESCRIPTION	
37800		None.
37801	EXIT STATUS	
37802		The following exit values shall be returned:
37803		0 Successful completion.
37804		>0 An error occurred.
37805	CONSEQUENCES OF ERRORS	
37806		Default.
37807	APPLICATION USAGE	
37808		None.
37809	EXAMPLES	
37810		None.
37811	RATIONALE	
37812		None.
37813	FUTURE DIRECTIONS	
37814		None.
37815	SEE ALSO	
37816		<i>uucp</i>
37817	CHANGE HISTORY	
37818		First released in Issue 2.

37819 Issue 6

- 37820 The normative text is reworded to avoid use of the term “must” for application requirements.
- 37821 The *LC_TIME* and *TZ* entries are removed from the ENVIRONMENT VARIABLES section.
- 37822 The UN margin code and associated shading are removed from the **-q** option in response to The
37823 Open Group Base Resolution bwg2001-003.

37824 NAME

37825 uux — remote command execution

37826 SYNOPSIS

37827 XSI `uux [-np] command-string`37828 `uux [-jnp] command-string`

37829

37830 DESCRIPTION

37831 The *uux* utility shall gather zero or more files from various systems, execute a shell pipeline (see
37832 Section 2.9 (on page 47)) on a specified system, and then send the standard output of the
37833 command to a file on a specified system. Only the first command of a pipeline can have a
37834 *system-name!* prefix. All other commands in the pipeline shall be executed on the system of the
37835 first command.

37836 The following restrictions are applicable to the shell pipeline processed by *uux*:

- In gathering files from different systems, pathname expansion shall not be performed by *uux*.
Thus, a request such as:

37839 `uux "c99 remsys!~/* .c"`

37840 would attempt to copy the file named literally `*.c` to the local system.

- The redirection operators "`>>`", "`<<`", "`>|`", and "`>&`" shall not be accepted. Any use of
these redirection operators shall cause this utility to write an error message describing the
problem and exit with a non-zero exit status.

- The reserved word `!` cannot be used at the head of the pipeline to modify the exit status. (See
the *command-string* operand description below.)

- Alias substitution shall not be performed.

37847 A filename can be specified as for *uucp*; it can be an absolute pathname, a pathname preceded by
37848 `-name` (which is replaced by the corresponding login directory), a pathname specified as `~/dest`
37849 (*dest* is prefixed by the public directory called *PUBDIR*; the actual location of *PUBDIR* is
37850 implementation-defined), or a simple filename (which is prefixed by *uux* with the current
37851 directory). See *uucp* for the details.

37852 The execution of commands on remote systems shall take place in an execution directory known
37853 to the *uucp* system. All files required for the execution shall be put into this directory unless they
37854 already reside on that machine. Therefore, the application shall ensure that non-local filenames
37855 (without path or machine reference) are unique within the *uux* request.

37856 The *uux* utility shall attempt to get all files to the execution system. For files that are output files,
37857 the application shall ensure that the filename is escaped using parentheses.

37858 The remote system shall notify the user by mail if the requested command on the remote system
37859 was disallowed or the files were not accessible. This notification can be turned off by the `-n`
37860 option.

37861 Typical implementations of this utility require a communications line configured to use the Base
37862 Definitions volume of IEEE Std 1003.1-2001, Chapter 11, General Terminal Interface, but other
37863 communications means may be used. On systems where there are no available communications
37864 means (either temporarily or permanently), this utility shall write an error message describing
37865 the problem and exit with a non-zero exit status.

37866 The *uux* utility cannot guarantee support for all character encodings in all circumstances. For
37867 example, transmission data may be restricted to 7 bits by the underlying network, 8-bit data and

37868 filenames need not be portable to non-internationalized systems, and so on. Under these
37869 circumstances, it is recommended that only characters defined in the ISO/IEC 646:1991
37870 standard International Reference Version (equivalent to ASCII) 7-bit range of characters be used
37871 and that only characters defined in the portable filename character set be used for naming files.

37872 OPTIONS

37873 The *uux* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section
37874 12.2, Utility Syntax Guidelines.

37875 The following options shall be supported:

- 37876 **-p** Make the standard input to *uux* the standard input to the *command-string*.
- 37877 **-j** Write the job identification string to standard output. This job identification can be
37878 used by *ustat* to obtain the status or terminate a job.
- 37879 **-n** Do not notify the user if the command fails.

37880 OPERANDS

37881 The following operand shall be supported:

37882 *command-string*
37883 A string made up of one or more arguments that are similar to normal command
37884 arguments, except that the command and any filenames can be prefixed by
37885 *system-name!*. A null *system-name* shall be interpreted as the local system.

37886 STDIN

37887 The standard input shall not be used unless the '**-**' or **-p** option is specified; in those cases, the
37888 standard input shall be made the standard input of the *command-string*.

37889 INPUT FILES

37890 Input files shall be selected according to the contents of *command-string*.

37891 ENVIRONMENT VARIABLES

37892 The following environment variables shall affect the execution of *uux*:

- 37893 **LANG** Provide a default value for the internationalization variables that are unset or null.
(See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2,
37894 Internationalization Variables for the precedence of internationalization variables
37895 used to determine the values of locale categories.)
- 37897 **LC_ALL** If set to a non-empty string value, override the values of all the other
37898 internationalization variables.
- 37899 **LC_CTYPE** Determine the locale for the interpretation of sequences of bytes of text data as
37900 characters (for example, single-byte as opposed to multi-byte characters in
37901 arguments).
- 37902 **LC_MESSAGES**
37903 Determine the locale that should be used to affect the format and contents of
37904 diagnostic messages written to standard error.
- 37905 **NLSPATH** Determine the location of message catalogs for the processing of *LC_MESSAGES*.

37906 ASYNCHRONOUS EVENTS

37907 Default.

37908 STDOUT

37909 The standard output shall not be used unless the **-j** option is specified; in that case, the job
37910 identification string shall be written to standard output in the following format:

37911 "**%s\n**" , <*jobid*>

37912 STDERR

37913 The standard error shall be used only for diagnostic messages.

37914 OUTPUT FILES

37915 Output files shall be created or written, or both, according to the contents of *command-string*.

37916 If **-n** is not used, mail files shall be modified following any command or file-access failures on
37917 the remote system.

37918 EXTENDED DESCRIPTION

37919 None.

37920 EXIT STATUS

37921 The following exit values shall be returned:

37922 0 Successful completion.

37923 >0 An error occurred.

37924 CONSEQUENCES OF ERRORS

37925 Default.

37926 APPLICATION USAGE

37927 Note that, for security reasons, many installations limit the list of commands executable on
37928 behalf of an incoming request from *uux*. Many sites permit little more than the receipt of mail
37929 via *uux*.

37930 Any characters special to the command interpreter should be quoted either by quoting the entire
37931 *command-string* or quoting the special characters as individual arguments.

37932 As noted in *uucp*, shell pattern matching notation characters appearing in pathnames are
37933 expanded on the appropriate local system. This is done under the control of local settings of
37934 *LC_COLLATE* and *LC_CTYPE*. Thus, care should be taken when using bracketed filename
37935 patterns, as collation and typing rules may vary from one system to another. Also be aware that
37936 certain types of expression (that is, equivalence classes, character classes, and collating symbols)
37937 need not be supported on non-internationalized systems.

37938 EXAMPLES

- 37939 1. The following command gets **file1** from system **a** and **file2** from system **b**, executes *diff* on
37940 the local system, and puts the results in **file.diff** in the local *PUBDIR* directory. (*PUBDIR* is
37941 the *uucp* public directory on the local system.)

37942 uux " !diff a!/usr/file1 b!/a4/file2 >!~/file.diff "

- 37943 2. The following command fails because *uux* places all files copied to a system in the same
37944 working directory. Although the files **xyz** are from two different systems, their filenames
37945 are the same and conflict.

37946 uux " !diff a!/usr1/xyz b!/usr2/xyz >!~/xyz.diff "

- 37947 3. The following command succeeds (assuming *diff* is permitted on system **a**) because the file
37948 local to system **a** is not copied to the working directory, and hence does not conflict with
37949 the file from system **c**.

37950 uux "a!diff a!/usr/xyz c!/usr/xyz >!~/xyz.diff"

37951 RATIONALE

37952 None.

37953 FUTURE DIRECTIONS

37954 None.

37955 SEE ALSO

37956 Chapter 2 (on page 29), *uucp*, *uuencode*, *uustat*

37957 CHANGE HISTORY

37958 First released in Issue 2.

37959 Issue 6

37960 The obsolescent SYNOPSIS is removed.

37961 The normative text is reworded to avoid use of the term “must” for application requirements.

37962 The UN margin code and associated shading are removed from the **-j** option in response to The
37963 Open Group Base Resolution bwg2001-003.

37964 NAME

37965 val — validate SCCS files (**DEVELOPMENT**)

37966 SYNOPSIS

37967 XSI val -

37968 val [-s][-m name][-r SID][-y type] file...

37969

37970 DESCRIPTION

37971 The *val* utility shall determine whether the specified *file* is an SCCS file meeting the
37972 characteristics specified by the options.

37973 OPTIONS

37974 The *val* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section 12.2,
37975 Utility Syntax Guidelines, except that the usage of the '-' operand is not strictly as intended by
37976 the guidelines (that is, reading options and operands from standard input).

37977 The following options shall be supported:

- 37978 **-m name** Specify a *name*, which is compared with the SCCS %M% keyword in *file*; see *get*.
- 37979 **-r SID** Specify a *SID* (SCCS Identification String), an SCCS delta number. A check shall be
37980 made to determine whether the *SID* is ambiguous (for example, **-r 1** is ambiguous
37981 because it physically does not exist but implies 1.1, 1.2, and so on, which may
37982 exist) or invalid (for example, **-r 1.0** or **-r 1.1.0** are invalid because neither case can
37983 exist as a valid delta number). If the *SID* is valid and not ambiguous, a check shall
37984 be made to determine whether it actually exists.
- 37985 **-s** Silence the diagnostic message normally written to standard output for any error
37986 that is detected while processing each named file on a given command line.
- 37987 **-y type** Specify a *type*, which shall be compared with the SCCS %Y% keyword in *file*; see
37988 *get*.

37989 OPERANDS

37990 The following operands shall be supported:

- 37991 *file* A pathname of an existing SCCS file. If exactly one *file* operand appears, and it is
37992 '-', the standard input shall be read: each line shall be independently processed
37993 as if it were a command line argument list. (However, the line is not subjected to
37994 any of the shell word expansions, such as parameter expansion or quote removal.)

37995 STDIN

37996 The standard input shall be a text file used only when the *file* operand is specified as '-'.

37997 INPUT FILES

37998 Any SCCS files processed shall be files of an unspecified format.

37999 ENVIRONMENT VARIABLES

38000 The following environment variables shall affect the execution of *val*:

- 38001 **LANG** Provide a default value for the internationalization variables that are unset or null.
38002 (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2,
38003 Internationalization Variables for the precedence of internationalization variables
38004 used to determine the values of locale categories.)
- 38005 **LC_ALL** If set to a non-empty string value, override the values of all the other
38006 internationalization variables.

38007 *LC_CTYPE* Determine the locale for the interpretation of sequences of bytes of text data as characters (for example, single-byte as opposed to multi-byte characters in arguments and input files).
38008
38009

38010 *LC_MESSAGES* Determine the locale that should be used to affect the format and contents of diagnostic messages written to standard error, and informative messages written to standard output.

38014 NLSPATH Determine the location of message catalogs for the processing of *LC_MESSAGES*.

38015 ASYNCHRONOUS EVENTS

38016 Default.

38017 STDOUT

38018 The standard output shall consist of informative messages about either:

1. Each file processed
 2. Each command line read from standard input

If the standard input is not used, for each *file* operand yielding a discrepancy, the output line shall have the following format:

38023 "%s: %s\n", <pathname>, <unspecified string>

If standard input is used, a line of input shall be written before each of the preceding lines for files containing discrepancies:

38026 "%s:\n", <input_line>

38027 STDERR

38028 Not used.

38029 OUTPUT FILES

38030 None.

38031 EXTENDED DESCRIPTION

38032 None.

38033 EXIT STATUS

38034 The 8-bit code returned by *val* shall be a disjunction of the possible errors; that is, it can be
38035 interpreted as a bit string where set bits are interpreted as follows:

38036	0x80	=	Missing file argument.
38037	0x40	=	Unknown or duplicate option.
38038	0x20	=	Corrupted SCCS file.
38039	0x10	=	Cannot open file or file not SCCS.
38040	0x08	=	<i>SID</i> is invalid or ambiguous.
38041	0x04	=	<i>SID</i> does not exist.
38042	0x02	=	%Y%, -y mismatch.
38043	0x01	=	%M%, -m mismatch.

38044 Note that `val` can process two or more files on a given command line and can process multiple
38045 command lines (when reading the standard input). In these cases an aggregate code shall be
38046 returned: a logical OR of the codes generated for each command line and file processed.

38047 CONSEQUENCES OF ERRORS

38048 Default.

38049 APPLICATION USAGE

38050 Since the *val* exit status sets the 0x80 bit, shell applications checking "\$?" cannot tell if it
38051 terminated due to a missing file argument or receipt of a signal.

38052 EXAMPLES

38053 In a directory with three SCCS files—**s.x** (of **t** type “text”), **s.y**, and **s.z** (a corrupted file)—the
38054 following command could produce the output shown:

```
38055     val - <<EOF
38056     -y source s.x
38057     -m y s.y
38058     s.z
38059     EOF
38060
38061         -y source s.x
38062             s.x: %Y%, -y mismatch
38063             s.z
38064             s.z: corrupted SCCS file
```

38064 RATIONALE

38065 None.

38066 FUTURE DIRECTIONS

38067 None.

38068 SEE ALSO

38069 *admin, delta, get, prs*

38070 CHANGE HISTORY

38071 First released in Issue 2.

38072 Issue 6

38073 The Open Group Corrigendum U025/4 is applied, correcting a typographical error in the EXIT
38074 STATUS.

38075 NAME

38076 *vi* — screen-oriented (visual) display editor

38077 SYNOPSIS

38078 UP *vi* [-rR][-c *command*][-t *tagstring*][-w *size*][*file* ...]

38079

38080 DESCRIPTION

38081 This utility shall be provided on systems that both support the User Portability Utilities option
38082 and define the POSIX2_CHAR_TERM symbol. On other systems it is optional.

38083 The *vi* (visual) utility is a screen-oriented text editor. Only the open and visual modes of the
38084 editor are described in IEEE Std 1003.1-2001; see the line editor *ex* for additional editing
38085 capabilities used in *vi*. The user can switch back and forth between *vi* and *ex* and execute *ex*
38086 commands from within *vi*.

38087 This reference page uses the term *edit buffer* to describe the current working text. No specific
38088 implementation is implied by this term. All editing changes are performed on the edit buffer,
38089 and no changes to it shall affect any file until an editor command writes the file.

38090 When using *vi*, the terminal screen acts as a window into the editing buffer. Changes made to
38091 the editing buffer shall be reflected in the screen display; the position of the cursor on the screen
38092 shall indicate the position within the editing buffer.

38093 Certain terminals do not have all the capabilities necessary to support the complete *vi* definition.
38094 When these commands cannot be supported on such terminals, this condition shall not produce
38095 an error message such as “not an editor command” or report a syntax error. The implementation
38096 may either accept the commands and produce results on the screen that are the result of an
38097 unsuccessful attempt to meet the requirements of this volume of IEEE Std 1003.1-2001 or report
38098 an error describing the terminal-related deficiency.

38099 OPTIONS

38100 The *vi* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section 12.2,
38101 Utility Syntax Guidelines.

38102 The following options shall be supported:

38103 **-c *command*** See the *ex* command description of the **-c** option.

38104 **-r** See the *ex* command description of the **-r** option.

38105 **-R** See the *ex* command description of the **-R** option.

38106 **-t *tagstring*** See the *ex* command description of the **-t** option.

38107 **-w *size*** See the *ex* command description of the **-w** option.

38108 OPERANDS

38109 See the OPERANDS section of the *ex* command for a description of the operands supported by
38110 the *vi* command.

38111 STDIN

38112 If standard input is not a terminal device, the results are undefined. The standard input consists
38113 of a series of commands and input text, as described in the EXTENDED DESCRIPTION section.

38114 If a read from the standard input returns an error, or if the editor detects an end-of-file condition
38115 from the standard input, it shall be equivalent to a SIGHUP asynchronous event.

38116 INPUT FILES

38117 See the INPUT FILES section of the *ex* command for a description of the input files supported by
38118 the *vi* command.

38119 ENVIRONMENT VARIABLES

38120 See the ENVIRONMENT VARIABLES section of the *ex* command for the environment variables
38121 that affect the execution of the *vi* command.

38122 ASYNCHRONOUS EVENTS

38123 See the ASYNCHRONOUS EVENTS section of the *ex* for the asynchronous events that affect the
38124 execution of the *vi* command.

38125 STDOUT

38126 If standard output is not a terminal device, undefined results occur.

38127 Standard output may be used for writing prompts to the user, for informational messages, and
38128 for writing lines from the file.

38129 STDERR

38130 If standard output is not a terminal device, undefined results occur.

38131 The standard error shall be used only for diagnostic messages.

38132 OUTPUT FILES

38133 See the OUTPUT FILES section of the *ex* command for a description of the output files
38134 supported by the *vi* command.

38135 EXTENDED DESCRIPTION

38136 If the terminal does not have the capabilities necessary to support an unspecified portion of the
38137 *vi* definition, implementations shall start initially in *ex* mode or open mode. Otherwise, after
38138 initialization, *vi* shall be in command mode; text input mode can be entered by one of several
38139 commands used to insert or change text. In text input mode, <ESC> can be used to return to
38140 command mode; other uses of <ESC> are described later in this section; see **Terminate**
38141 **Command or Input Mode** (on page 997).

38142 Initialization in ex and vi

38143 See **Initialization in ex and vi** (on page 358) for a description of *ex* and *vi* initialization for the *vi*
38144 utility.

38145 Command Descriptions in vi

38146 The following symbols are used in this reference page to represent arguments to commands.

38147 *buffer* See the description of *buffer* in the EXTENDED DESCRIPTION section of the *ex* utility;
38148 see **Command Descriptions in ex** (on page 368).

38149 In open and visual mode, when a command synopsis shows both [*buffer*] and [*count*]
38150 preceding the command name, they can be specified in either order.

38151 *count* A positive integer used as an optional argument to most commands, either to give a
38152 repeat count or as a size. This argument is optional and shall default to 1 unless
38153 otherwise specified.

38154 The Synopsis lines for the *vi* commands <control>-G, <control>-L, <control>-R,
38155 <control>-], %, &, ^, D, m, M, Q, u, U, and ZZ do not have *count* as an optional
38156 argument. Regardless, it shall not be an error to specify a *count* to these commands, and
38157 any specified *count* shall be ignored.

38158 ***motion*** An optional trailing argument used by the **!**, **<**, **>**, **c**, **d**, and **y** commands, which is used
 38159 to indicate the region of text that shall be affected by the command. The motion can be
 38160 either one of the command characters repeated or one of several other *vi* commands
 38161 (listed in the following table). Each of the applicable commands specifies the region of
 38162 text matched by repeating the command; each command that can be used as a motion
 38163 command specifies the region of text it affects.

38164 Commands that take *motion* arguments operate on either lines or characters, depending
 38165 on the circumstances. When operating on lines, all lines that fall partially or wholly
 38166 within the text region specified for the command shall be affected. When operating on
 38167 characters, only the exact characters in the specified text region shall be affected. Each
 38168 motion command specifies this individually.

38169 When commands that may be motion commands are not used as motion commands,
 38170 they shall set the current position to the current line and column as specified.

38171 The following commands shall be valid cursor motion commands:

38172 <apostrophe>	(-	j	H
38173 <carriage-return>)	\$	k	L
38174 <comma>	[[%	l	M
38175 <control>-H]]	_	n	N
38176 <control>-N	{	;	t	T
38177 <control>-P	}	?	w	W
38178 <grave accent>	^	b	B	
38179 <newline>	+	e	E	
38180 <space>		f	F	
38181 <zero>	/	h	G	

38182 Any *count* that is specified to a command that has an associated motion command shall
 38183 be applied to the motion command. If a *count* is applied to both the command and its
 38184 associated motion command, the effect shall be multiplicative.

38185 The following symbols are used in this section to specify locations in the edit buffer:

38186 ***current character***

38187 The character that is currently indicated by the cursor.

38188 ***end of a line***

38189 The point located between the last non-<newline> (if any) and the terminating
 38190 <newline> of a line. For an empty line, this location coincides with the beginning of the
 38191 line.

38192 ***end of the edit buffer***

38193 The location corresponding to the end of the last line in the edit buffer.

38194 The following symbols are used in this section to specify command actions:

38195 ***bigword*** In the POSIX locale, *vi* shall recognize four kinds of *bigwords*:

- 38196 1. A maximal sequence of non-<blank>s preceded and followed by <blank>s or the
 38197 beginning or end of a line or the edit buffer
- 38198 2. One or more sequential blank lines
- 38199 3. The first character in the edit buffer
- 38200 4. The last non-<newline> in the edit buffer

- 38201 *word* In the POSIX locale, *vi* shall recognize five kinds of words:
- 38202 1. A maximal sequence of letters, digits, and underscores, delimited at both ends by:
 - 38203 — Characters other than letters, digits, or underscores
 - 38204 — The beginning or end of a line
 - 38205 — The beginning or end of the edit buffer
 - 38206 2. A maximal sequence of characters other than letters, digits, underscores, or
38207 <blank>s, delimited at both ends by:
 - 38208 — A letter, digit, underscore
 - 38209 — <blank>s
 - 38210 — The beginning or end of a line
 - 38211 — The beginning or end of the edit buffer
 - 38212 3. One or more sequential blank lines
 - 38213 4. The first character in the edit buffer
 - 38214 5. The last non-<newline> in the edit buffer

38215 *section boundary*

38216 A *section boundary* is one of the following:

- 38217 1. A line whose first character is a <form-feed>
- 38218 2. A line whose first character is an open curly brace (' { ')
- 38219 3. A line whose first character is a period and whose second and third characters
38220 match a two-character pair in the **sections** edit option (see *ed*)
- 38221 4. A line whose first character is a period and whose only other character matches
38222 the first character of a two-character pair in the **sections** edit option, where the
38223 second character of the two-character pair is a <space>
- 38224 5. The first line of the edit buffer
- 38225 6. The last line of the edit buffer if the last line of the edit buffer is empty or if it is a
38226]] or } command; otherwise, the last non-<newline> of the last line of the edit
38227 buffer

38228 *paragraph boundary*

38229 A *paragraph boundary* is one of the following:

- 38230 1. A section boundary
- 38231 2. A line whose first character is a period and whose second and third characters
38232 match a two-character pair in the **paragraphs** edit option (see *ed*)
- 38233 3. A line whose first character is a period and whose only other character matches
38234 the first character of a two-character pair in the **paragraphs** edit option, where the
38235 second character of the two-character pair is a <space>
- 38236 4. One or more sequential blank lines

38237 *remembered search direction*

38238 See the description of *remembered search direction* in *ed*.

38239 *sentence boundary*

38240 A *sentence boundary* is one of the following:

- 38241 1. A paragraph boundary
- 38242 2. The first non-<blank> that occurs after a paragraph boundary
- 38243 3. The first non-<blank> that occurs after a period ('.'), exclamation mark ('!'),
38244 or question mark ('?'), followed by two <space>s or the end of a line; any
38245 number of closing parenthesis (')'), closing brackets ('[]'), double quote ('"'),
38246 or single quote ('''') characters can appear between the punctuation mark and
38247 the two <space>s or end-of-line

38248 In the remainder of the description of the *vi* utility, the term "buffer line" refers to a line in the
38249 edit buffer and the term "display line" refers to the line or lines on the display screen used to
38250 display one buffer line. The term "current line" refers to a specific "buffer line".

38251 If there are display lines on the screen for which there are no corresponding buffer lines because
38252 they correspond to lines that would be after the end of the file, they shall be displayed as a single
38253 tilde ('~') character, plus the terminating <newline>.

38254 The last line of the screen shall be used to report errors or display informational messages. It
38255 shall also be used to display the input for "line-oriented commands" (/, ?, :, and !). When a line-
38256 oriented command is executed, the editor shall enter text input mode on the last line on the
38257 screen, using the respective command characters as prompt characters. (In the case of the !
38258 command, the associated motion shall be entered by the user before the editor enters text input
38259 mode.) The line entered by the user shall be terminated by a <newline>, a non-<control>-V-
38260 escaped <carriage-return>, or unescaped <ESC>. It is unspecified if more characters than
38261 require a display width minus one column number of screen columns can be entered.

38262 If any command is executed that overwrites a portion of the screen other than the last line of the
38263 screen (for example, the *ex suspend* or ! commands), other than the *ex shell* command, the user
38264 shall be prompted for a character before the screen is refreshed and the edit session continued.

38265 <tab>s shall take up the number of columns on the screen set by the **tabstop** edit option (see *ed*),
38266 unless there are less than that number of columns before the display margin that will cause the
38267 displayed line to be folded; in this case, they shall only take up the number of columns up to that
38268 boundary.

38269 The cursor shall be placed on the current line and relative to the current column as specified by
38270 each command described in the following sections.

38271 In open mode, if the current line is not already displayed, then it shall be displayed.

38272 In visual mode, if the current line is not displayed, then the lines that are displayed shall be
38273 expanded, scrolled, or redrawn to cause an unspecified portion of the current line to be
38274 displayed. If the screen is redrawn, no more than the number of display lines specified by the
38275 value of the **window** edit option shall be displayed (unless the current line cannot be completely
38276 displayed in the number of display lines specified by the **window** edit option) and the current
38277 line shall be positioned as close to the center of the displayed lines as possible (within the
38278 constraints imposed by the distance of the line from the beginning or end of the edit buffer). If
38279 the current line is before the first line in the display and the screen is scrolled, an unspecified
38280 portion of the current line shall be placed on the first line of the display. If the current line is after
38281 the last line in the display and the screen is scrolled, an unspecified portion of the current line
38282 shall be placed on the last line of the display.

38283 In visual mode, if a line from the edit buffer (other than the current line) does not entirely fit into
38284 the lines at the bottom of the display that are available for its presentation, the editor may

choose not to display any portion of the line. The lines of the display that do not contain text from the edit buffer for this reason shall each consist of a single '@' character.

In visual mode, the editor may choose for unspecified reasons to not update lines in the display to correspond to the underlying edit buffer text. The lines of the display that do not correctly correspond to text from the edit buffer for this reason shall consist of a single '@' character (plus the terminating <newline>), and the <control>-R command shall cause the editor to update the screen to correctly represent the edit buffer.

Open and visual mode commands that set the current column set it to a column position in the display, and not a character position in the line. In this case, however, the column position in the display shall be calculated for an infinite width display; for example, the column related to a character that is part of a line that has been folded onto additional screen lines will be offset from the display line column where the buffer line begins, not from the beginning of a particular display line.

The display cursor column in the display is based on the value of the current column, as follows, with each rule applied in turn:

1. If the current column is after the last display line column used by the displayed line, the display cursor column shall be set to the last display line column occupied by the last non-<newline> in the current line; otherwise, the display cursor column shall be set to the current column.
2. If the character of which some portion is displayed in the display line column specified by the display cursor column requires more than a single display line column:
 - a. If in text input mode, the display cursor column shall be adjusted to the first display line column in which any portion of that character is displayed.
 - b. Otherwise, the display cursor column shall be adjusted to the last display line column in which any portion of that character is displayed.

The current column shall not be changed by these adjustments to the display cursor column.

If an error occurs during the parsing or execution of a vi command:

- The terminal shall be alerted. Execution of the vi command shall stop, and the cursor (for example, the current line and column) shall not be further modified.
- Unless otherwise specified by the following command sections, it is unspecified whether an informational message shall be displayed.
- Any partially entered vi command shall be discarded.
- If the vi command resulted from a map expansion, all characters from that map expansion shall be discarded, except as otherwise specified by the map command (see ed).
- If the vi command resulted from the execution of a buffer, no further commands caused by the execution of the buffer shall be executed.

38321 **Page Backwards**

38322 *Synopsis:* [*count*] <control>-B

38323 If in open mode, the <control>-B command shall behave identically to the z command.
38324 Otherwise, if the current line is the first line of the edit buffer, it shall be an error.

38325 If the **window** edit option is less than 3, display a screen where the last line of the display shall
38326 be some portion of:

38327 (*current first line*) -1

38328 otherwise, display a screen where the first line of the display shall be some portion of:

38329 (*current first line*) - *count* x ((**window** edit option) -2)

38330 If this calculation would result in a line that is before the first line of the edit buffer, the first line
38331 of the display shall display some portion of the first line of the edit buffer.

38332 *Current line*: If no lines from the previous display remain on the screen, set to the last line of the
38333 display; otherwise, set to (*line* - the number of new lines displayed on this screen).

38334 *Current column*: Set to non-<blank>.

38335 **Scroll Forward**

38336 *Synopsis:* [*count*] <control>-D

38337 If the current line is the last line of the edit buffer, it shall be an error.

38338 If no *count* is specified, *count* shall default to the *count* associated with the previous <control>-D
38339 or <control>-U command. If there was no previous <control>-D or <control>-U command, *count*
38340 shall default to the value of the **scroll** edit option.

38341 If in open mode, write lines starting with the line after the current line, until *count* lines or the
38342 last line of the file have been written.

38343 *Current line*: If the current line + *count* is past the last line of the edit buffer, set to the last line of
38344 the edit buffer; otherwise, set to the current line + *count*.

38345 *Current column*: Set to non-<blank>.

38346 **Scroll Forward by Line**

38347 *Synopsis:* [*count*] <control>-E

38348 Display the line *count* lines after the last line currently displayed.

38349 If the last line of the edit buffer is displayed, it shall be an error. If there is no line *count* lines
38350 after the last line currently displayed, the last line of the display shall display some portion of
38351 the last line of the edit buffer.

38352 *Current line*: Unchanged if the previous current character is displayed; otherwise, set to the first
38353 line displayed.

38354 *Current column*: Unchanged.

38355 **Page Forward**

38356 *Synopsis:* [*count*] <control>-F

38357 If in open mode, the <control>-F command shall behave identically to the z command.
38358 Otherwise, if the current line is the last line of the edit buffer, it shall be an error.

38359 If the **window** edit option is less than 3, display a screen where the first line of the display shall
38360 be some portion of:

38361 (*current last line*) +1

38362 otherwise, display a screen where the first line of the display shall be some portion of:

38363 (*current first line*) + *count* x ((**window** edit option) -2)

38364 If this calculation would result in a line that is after the last line of the edit buffer, the last line of
38365 the display shall display some portion of the last line of the edit buffer.

38366 *Current line*: If no lines from the previous display remain on the screen, set to the first line of the
38367 display; otherwise, set to (*line* + the number of new lines displayed on this screen).

38368 *Current column*: Set to non-<blank>.

38369 **Display Information**

38370 *Synopsis:* <control>-G

38371 This command shall be equivalent to the **ex file** command.

38372 **Move Cursor Backwards**

38373 *Synopsis:* [*count*] <control>-H

38374 [*count*] h

38375 the current erase character (see stty)

38376 If there are no characters before the current character on the current line, it shall be an error. If
38377 there are less than *count* previous characters on the current line, *count* shall be adjusted to the
38378 number of previous characters on the line.

38379 If used as a motion command:

38380 1. The text region shall be from the character before the starting cursor up to and including
38381 the *count*th character before the starting cursor.

38382 2. Any text copied to a buffer shall be in character mode.

38383 If not used as a motion command:

38384 *Current line*: Unchanged.

38385 *Current column*: Set to (*column* – the number of columns occupied by *count* characters ending
38386 with the previous current column).

38387

Move Down

38388 *Synopsis:* [count] <newline>
38389 [count] <control>-J
38390 [count] <control>-M
38391 [count] <control>-N
38392 [count] j
38393 [count] <carriage-return>
38394 [count] +

38395 If there are less than *count* lines after the current line in the edit buffer, it shall be an error.

38396 If used as a motion command:

- 38397 1. The text region shall include the starting line and the next *count* – 1 lines.
- 38398 2. Any text copied to a buffer shall be in line mode.

38399 If not used as a motion command:

38400 *Current line:* Set to *current line*+ *count*.

38401 *Current column:* Set to non-<blank> for the <carriage-return>, <control>-M, and + commands;
38402 otherwise, unchanged.

Clear and Redisplay

38404 *Synopsis:* <control>-L

38405 If in open mode, clear the screen and redisplay the current line. Otherwise, clear and redisplay
38406 the screen.

38407 *Current line:* Unchanged.

38408 *Current column:* Unchanged.

Move Up

38410 *Synopsis:* [count] <control>-P
38411 [count] k
38412 [count] –

38413 If there are less than *count* lines before the current line in the edit buffer, it shall be an error.

38414 If used as a motion command:

- 38415 1. The text region shall include the starting line and the previous *count* lines.
- 38416 2. Any text copied to a buffer shall be in line mode.

38417 If not used as a motion command:

38418 *Current line:* Set to *current line* – *count*.

38419 *Current column:* Set to non-<blank> for the – command; otherwise, unchanged.

38420 **Redraw Screen**

38421 *Synopsis:* <control>-R

38422 If any lines have been deleted from the display screen and flagged as deleted on the terminal
38423 using the @ convention (see the beginning of the EXTENDED DESCRIPTION section), they shall
38424 be redisplayed to match the contents of the edit buffer.

38425 It is unspecified whether lines flagged with @ because they do not fit on the terminal display
38426 shall be affected.

38427 *Current line:* Unchanged.

38428 *Current column:* Unchanged.

38429 **Scroll Backward**

38430 *Synopsis:* [*count*] <control>-U

38431 If the current line is the first line of the edit buffer, it shall be an error.

38432 If no *count* is specified, *count* shall default to the *count* associated with the previous <control>-D
38433 or <control>-U command. If there was no previous <control>-D or <control>-U command, *count*
38434 shall default to the value of the **scroll** edit option.

38435 *Current line:* If *count* is greater than the current line, set to 1; otherwise, set to the current line –
38436 *count*.

38437 *Current column:* Set to non-<blank>.

38438 **Scroll Backward by Line**

38439 *Synopsis:* [*count*] <control>-Y

38440 Display the line *count* lines before the first line currently displayed.

38441 If the current line is the first line of the edit buffer, it shall be an error. If this calculation would
38442 result in a line that is before the first line of the edit buffer, the first line of the display shall
38443 display some portion of the first line of the edit buffer.

38444 *Current line:* Unchanged if the previous current character is displayed; otherwise, set to the first
38445 line displayed.

38446 *Current column:* Unchanged.

38447 **Edit the Alternate File**

38448 *Synopsis:* <control>-^

38449 This command shall be equivalent to the ex **edit** command, with the alternate pathname as its
38450 argument.

38451 **Terminate Command or Input Mode**

38452 *Synopsis:* <ESC>

38453 If a partial vi command (as defined by at least one, non-*count* character) has been entered,
38454 discard the *count* and the command character(s).

38455 Otherwise, if no command characters have been entered, and the <ESC> was the result of a map
38456 expansion, the terminal shall be alerted and the <ESC> character shall be discarded, but it shall
38457 not be an error.

38458 Otherwise, it shall be an error.

38459 *Current line*: Unchanged.

38460 *Current column*: Unchanged.

38461 **Search for tagstring**

38462 *Synopsis*: <control>-]

38463 If the current character is not a word or <blank>, it shall be an error.

38464 This command shall be equivalent to the **ex tag** command, with the argument to that command
38465 defined as follows.

38466 If the current character is a <blank>:

38467 1. Skip all <blank>s after the cursor up to the end of the line.

38468 2. If the end of the line is reached, it shall be an error.

38469 Then, the argument to the **ex tag** command shall be the current character and all subsequent
38470 characters, up to the first non-word character or the end of the line.

38471 **Move Cursor Forward**

38472 *Synopsis*: [count] <space>

38473 [count] l (ell)

38474 If there are less than *count* non-<newline>s after the cursor on the current line, *count* shall be
38475 adjusted to the number of non-<newline>s after the cursor on the line.

38476 If used as a motion command:

38477 1. If the current or *countth* character after the cursor is the last non-<newline> in the line, the
38478 text region shall be comprised of the current character up to and including the last non-
38479 <newline> in the line. Otherwise, the text region shall be from the current character up to,
38480 but not including, the *countth* character after the cursor.

38481 2. Any text copied to a buffer shall be in character mode.

38482 If not used as a motion command:

38483 If there are no non-<newline>s after the current character on the current line, it shall be an error.

38484 *Current line*: Unchanged.

38485 *Current column*: Set to the last column that displays any portion of the *countth* character after the
38486 current character.

38487 **Replace Text with Results from Shell Command**

38488 *Synopsis*: [count] ! motion shell-commands <newline>

38489 If the motion command is the ! command repeated:

38490 1. If the edit buffer is empty and no *count* was supplied, the command shall be the equivalent
38491 of the **ex :read !** command, with the text input, and no text shall be copied to any buffer.

38492 2. Otherwise:

38493 a. If there are less than *count* -1 lines after the current line in the edit buffer, it shall be
38494 an error.

- 38495 b. The text region shall be from the current line up to and including the next *count* – 1
38496 lines.

38497 Otherwise, the text region shall be the lines in which any character of the text region specified by
38498 the motion command appear.

38499 Any text copied to a buffer shall be in line mode.

38500 This command shall be equivalent to the *ex!* command for the specified lines.

38501 **Move Cursor to End-of-Line**

38502 *Synopsis:* [*count*] \$

38503 It shall be an error if there are less than (*count* – 1) lines after the current line in the edit buffer.

38504 If used as a motion command:

- 38505 1. If *count* is 1:

38506 a. It shall be an error if the line is empty.

38507 b. Otherwise, the text region shall consist of all characters from the starting cursor to
38508 the last non-<newline> in the line, inclusive, and any text copied to a buffer shall be
38509 in character mode.

38510 2. Otherwise, if the starting cursor position is at or before the first non-<blank> in the line,
38511 the text region shall consist of the current and the next *count* – 1 lines, and any text saved to
38512 a buffer shall be in line mode.

38513 3. Otherwise, the text region shall consist of all characters from the starting cursor to the last
38514 non-<newline> in the line that is *count* – 1 lines forward from the current line, and any text
38515 copied to a buffer shall be in character mode.

38516 If not used as a motion command:

38517 *Current line:* Set to the *current line* + *count* – 1.

38518 *Current column:* The current column is set to the last display line column of the last non-
38519 <newline> in the line, or column position 1 if the line is empty.

38520 The current column shall be adjusted to be on the last display line column of the last non-
38521 <newline> of the current line as subsequent commands change the current line, until a
38522 command changes the current column.

38523 **Move to Matching Character**

38524 *Synopsis:* %

38525 If the character at the current position is not a parenthesis, bracket, or curly brace, search
38526 forward in the line to the first one of those characters. If no such character is found, it shall be an
38527 error.

38528 The matching character shall be the parenthesis, bracket, or curly brace matching the
38529 parenthesis, bracket, or curly brace, respectively, that was at the current position or that was
38530 found on the current line.

38531 Matching shall be determined as follows, for an open parenthesis:

- 38532 1. Set a counter to 1.
38533 2. Search forwards until a parenthesis is found or the end of the edit buffer is reached.

- 38534 3. If the end of the edit buffer is reached, it shall be an error.
- 38535 4. If an open parenthesis is found, increment the counter by 1.
- 38536 5. If a close parenthesis is found, decrement the counter by 1.
- 38537 6. If the counter is zero, the current character is the matching character.
- 38538 Matching for a close parenthesis shall be equivalent, except that the search shall be backwards,
- 38539 from the starting character to the beginning of the buffer, a close parenthesis shall increment the
- 38540 counter by 1, and an open parenthesis shall decrement the counter by 1.
- 38541 Matching for brackets and curly braces shall be equivalent, except that searching shall be done
- 38542 for open and close brackets or open and close curly braces. It is implementation-defined whether
- 38543 other characters are searched for and matched as well.
- 38544 If used as a motion command:
- 38545 1. If the matching cursor was after the starting cursor in the edit buffer, and the starting
 - 38546 cursor position was at or before the first non-<blank> non-<newline> in the starting line,
 - 38547 and the matching cursor position was at or after the last non-<blank> non-<newline> in the
 - 38548 matching line, the text region shall consist of the current line to the matching line,
 - 38549 inclusive, and any text copied to a buffer shall be in line mode.
 - 38550 2. If the matching cursor was before the starting cursor in the edit buffer, and the starting
 - 38551 cursor position was at or after the last non-<blank> non-<newline> in the starting line, and
 - 38552 the matching cursor position was at or before the first non-<blank> non-<newline> in the
 - 38553 matching line, the text region shall consist of the current line to the matching line,
 - 38554 inclusive, and any text copied to a buffer shall be in line mode.
 - 38555 3. Otherwise, the text region shall consist of the starting character to the matching character,
 - 38556 inclusive, and any text copied to a buffer shall be in character mode.
- 38557 If not used as a motion command:
- 38558 *Current line*: Set to the line where the matching character is located.
- 38559 *Current column*: Set to the last column where any portion of the matching character is displayed.
- 38560 **Repeat Substitution**
- 38561 *Synopsis*: &
- 38562 Repeat the previous substitution command. This command shall be equivalent to the *ex &*
- 38563 command with the current line as its addresses, and without *options*, *count*, or *flags*.
- 38564 **Return to Previous Context at Beginning of Line**
- 38565 *Synopsis*: ' *character*
- 38566 It shall be an error if there is no line in the edit buffer marked by *character*.
- 38567 If used as a motion command:
- 38568 1. If the starting cursor is after the marked cursor, then the locations of the starting cursor
 - 38569 and the marked cursor in the edit buffer shall be logically swapped.
 - 38570 2. The text region shall consist of the starting line up to and including the marked line, and
 - 38571 any text copied to a buffer shall be in line mode.
- 38572 If not used as a motion command:

38573 *Current line*: Set to the line referenced by the mark.

38574 *Current column*: Set to non-<blank>.

38575 **Return to Previous Context**

38576 *Synopsis*: ` character

38577 It shall be an error if the marked line is no longer in the edit buffer. If the marked line no longer contains a character in the saved numbered character position, it shall be as if the marked position is the first non-<blank>.

38580 If used as a motion command:

1. It shall be an error if the marked cursor references the same character in the edit buffer as the starting cursor.

2. If the starting cursor is after the marked cursor, then the locations of the starting cursor and the marked cursor in the edit buffer shall be logically swapped.

3. If the starting line is empty or the starting cursor is at or before the first non-<blank> non-<newline> of the starting line, and the marked cursor line is empty or the marked cursor references the first character of the marked cursor line, the text region shall consist of all lines containing characters from the starting cursor to the line before the marked cursor line, inclusive, and any text copied to a buffer shall be in line mode.

4. Otherwise, if the marked cursor line is empty or the marked cursor references a character at or before the first non-<blank> non-<newline> of the marked cursor line, the region of text shall be from the starting cursor to the last non-<newline> of the line before the marked cursor line, inclusive, and any text copied to a buffer shall be in character mode.

5. Otherwise, the region of text shall be from the starting cursor (inclusive), to the marked cursor (exclusive), and any text copied to a buffer shall be in character mode.

38596 If not used as a motion command:

38597 *Current line*: Set to the line referenced by the mark.

38598 *Current column*: Set to the last column in which any portion of the character referenced by the mark is displayed.

38600 **Return to Previous Section**

38601 *Synopsis*: [count] [[

1

38602 Move the cursor backward through the edit buffer to the first character of the previous section boundary, *count* times.

38604 If used as a motion command:

1. If the starting cursor was at the first character of the starting line or the starting line was empty, and the first character of the boundary was the first character of the boundary line, the text region shall consist of the current line up to and including the line where the *count*th next boundary starts, and any text copied to a buffer shall be in line mode.

2. If the boundary was the last line of the edit buffer or the last non-<newline> of the last line of the edit buffer, the text region shall consist of the last character in the edit buffer up to and including the starting character, and any text saved to a buffer shall be in character mode.

- 38613 3. Otherwise, the text region shall consist of the starting character up to but not including the
38614 first character in the *count*th next boundary, and any text copied to a buffer shall be in
38615 character mode.

38616 If not used as a motion command:

38617 *Current line*: Set to the line where the *count*th next boundary in the edit buffer starts.

38618 *Current column*: Set to the last column in which any portion of the first character of the *count*th
38619 next boundary is displayed, or column position 1 if the line is empty.

38620 Move to Next Section

38621 *Synopsis*: [*count*]]]

1

38622 Move the cursor forward through the edit buffer to the first character of the next section
38623 boundary, *count* times.

38624 If used as a motion command:

- 38625 1. If the starting cursor was at the first character of the starting line or the starting line was
38626 empty, and the first character of the boundary was the first character of the boundary line,
38627 the text region shall consist of the current line up to and including the line where the
38628 *count*th previous boundary starts, and any text copied to a buffer shall be in line mode.
- 38629 2. If the boundary was the first line of the edit buffer, the text region shall consist of the first
38630 character in the edit buffer up to but not including the starting character, and any text
38631 copied to a buffer shall be in character mode.
- 38632 3. Otherwise, the text region shall consist of the first character in the *count*th previous section
38633 boundary up to but not including the starting character, and any text copied to a buffer
38634 shall be in character mode.

38635 If not used as a motion command:

38636 *Current line*: Set to the line where the *count*th previous boundary in the edit buffer starts.

38637 *Current column*: Set to the last column in which any portion of the first character of the *count*th
38638 previous boundary is displayed, or column position 1 if the line is empty.

38639 Move to First Non-<blank> Position on Current Line

38640 *Synopsis*: ^

38641 If used as a motion command:

- 38642 1. If the line has no non-<blank> non-<newline>s, or if the cursor is at the first non-<blank>
38643 non-<newline> of the line, it shall be an error.
- 38644 2. If the cursor is before the first non-<blank> non-<newline> of the line, the text region shall
38645 be comprised of the current character, up to, but not including, the first non-<blank> non-
38646 <newline> of the line.
- 38647 3. If the cursor is after the first non-<blank> non-<newline> of the line, the text region shall
38648 be from the character before the starting cursor up to and including the first non-<blank>
38649 non-<newline> of the line.
- 38650 4. Any text copied to a buffer shall be in character mode.

38651 If not used as a motion command:

38652 *Current line*: Unchanged.

38653 *Current column*: Set to non-<blank>.

38654 **Current and Line Above**

38655 *Synopsis*: [*count*] _

38656 If there are less than *count* – 1 lines after the current line in the edit buffer, it shall be an error.

38657 If used as a motion command:

38658 1. If *count* is less than 2, the text region shall be the current line.

38659 2. Otherwise, the text region shall include the starting line and the next *count* – 1 lines.

38660 3. Any text copied to a buffer shall be in line mode.

38661 If not used as a motion command:

38662 *Current line*: Set to current line + *count* – 1.

38663 *Current column*: Set to non-<blank>.

38664 **Move Back to Beginning of Sentence**

38665 *Synopsis*: [*count*] (

38666 Move backward to the beginning of a sentence. This command shall be equivalent to the [[command, with the exception that sentence boundaries shall be used instead of section boundaries.

38669 **Move Forward to Beginning of Sentence**

38670 *Synopsis*: [*count*])

38671 Move forward to the beginning of a sentence. This command shall be equivalent to the]] command, with the exception that sentence boundaries shall be used instead of section boundaries.

38674 **Move Back to Preceding Paragraph**

38675 *Synopsis*: [*count*] {

38676 Move back to the beginning of the preceding paragraph. This command shall be equivalent to the [[command, with the exception that paragraph boundaries shall be used instead of section boundaries.

38679 **Move Forward to Next Paragraph**

38680 *Synopsis*: [*count*] }

38681 Move forward to the beginning of the next paragraph. This command shall be equivalent to the]] command, with the exception that paragraph boundaries shall be used instead of section boundaries.

38684 **Move to Specific Column Position**38685 *Synopsis:* [count] |

38686 For the purposes of this command, lines that are too long for the current display and that have
 38687 been folded shall be treated as having a single, 1-based, number of columns.

38688 If there are less than *count* columns in which characters from the current line are displayed on
 38689 the screen, *count* shall be adjusted to be the last column in which any portion of the line is
 38690 displayed on the screen.

38691 If used as a motion command:

- 38692 1. If the line is empty, or the cursor character is the same as the character on the *count*th
 38693 column of the line, it shall be an error.
- 38694 2. If the cursor is before the *count*th column of the line, the text region shall be comprised of
 38695 the current character, up to but not including the character on the *count*th column of the
 38696 line.
- 38697 3. If the cursor is after the *count*th column of the line, the text region shall be from the
 38698 character before the starting cursor up to and including the character on the *count*th
 38699 column of the line.

38700 4. Any text copied to a buffer shall be in character mode.

38701 If not used as a motion command:

38702 *Current line:* Unchanged.

38703 *Current column:* Set to the last column in which any portion of the character that is displayed in
 38704 the *count* column of the line is displayed.

38705 **Reverse Find Character**38706 *Synopsis:* [count] ,

38707 If the last F, f, T, or t command was F, f, T, or t, this command shall be equivalent to an f, F, t, or
 38708 T command, respectively, with the specified *count* and the same search character.

38709 If there was no previous F, f, T, or t command, it shall be an error.

38710 **Repeat**38711 *Synopsis:* [count] .

38712 Repeat the last !, <, >, A, C, D, I, J, O, P, R, S, X, Y, a, c, d, i, o, p, r, s, x, y, or ~ command. It shall
 38713 be an error if none of these commands have been executed. Commands (other than commands
 38714 that enter text input mode) executed as a result of map expansions, shall not change the value of
 38715 the last repeatable command.

38716 Repeated commands with associated motion commands shall repeat the motion command as
 38717 well; however, any specified *count* shall replace the *count*(s) that were originally specified to the
 38718 repeated command or its associated motion command.

38719 If the motion component of the repeated command is f, F, t, or T, the repeated command shall
 38720 not set the remembered search character for the ; and , commands.

38721 If the repeated command is p or P, and the buffer associated with that command was a numeric
 38722 buffer named with a number less than 9, the buffer associated with the repeated command shall
 38723 be set to be the buffer named by the name of the previous buffer logically incremented by 1.

38724 If the repeated character is a text input command, the input text associated with that command
38725 is repeated literally:

- 38726 • Input characters are neither macro or abbreviation-expanded.
- 38727 • Input characters are not interpreted in any special way with the exception that <newline>,
38728 <carriage-return>, and <control>-T behave as described in **Input Mode Commands in vi** (on
38729 page 1023).

38730 *Current line*: Set as described for the repeated command.

38731 *Current column*: Set as described for the repeated command.

38732 **Find Regular Expression**

38733 *Synopsis*: /

38734 If the input line contains no non-<newline>s, it shall be equivalent to a line containing only the
38735 last regular expression encountered. The enhanced regular expressions supported by vi are
38736 described in **Regular Expressions in ex** (on page 391).

38737 Otherwise, the line shall be interpreted as one or more regular expressions, optionally followed
38738 by an address offset or a vi z command.

38739 If the regular expression is not the last regular expression on the line, or if a line offset or z
38740 command is specified, the regular expression shall be terminated by an unescaped ' /'
38741 character, which shall not be used as part of the regular expression. If the regular expression is
38742 not the first regular expression on the line, it shall be preceded by zero or more <blank>s, a
38743 semicolon, zero or more <blank>s, and a leading ' /' character, which shall not be interpreted as
38744 part of the regular expression. It shall be an error to precede any regular expression with any
38745 characters other than these.

38746 Each search shall begin from the character after the first character of the last match (or, if it is the
38747 first search, after the cursor). If the **wrapscan** edit option is set, the search shall continue to the
38748 character before the starting cursor character; otherwise, to the end of the edit buffer. It shall be
38749 an error if any search fails to find a match, and an informational message to this effect shall be
38750 displayed.

38751 An optional address offset (see **Addressing in ex** (on page 361)) can be specified after the last
38752 regular expression by including a trailing ' /' character after the regular expression and
38753 specifying the address offset. This offset will be from the line containing the match for the last
38754 regular expression specified. It shall be an error if the line offset would indicate a line address
38755 less than 1 or greater than the last line in the edit buffer. An address offset of zero shall be
38756 supported. It shall be an error to follow the address offset with any other characters than
38757 <blank>s.

38758 If not used as a motion command, an optional z command (see **Redraw Window** (on page 1022))
38759 can be specified after the last regular expression by including a trailing ' /' character after the
38760 regular expression, zero or more <blank>s, a 'z', zero or more <blank>s, an optional new
38761 **window** edit option value, zero or more <blank>s, and a location character. The effect shall be as
38762 if the z command was executed after the / command. It shall be an error to follow the z
38763 command with any other characters than <blank>s.

38764 The remembered search direction shall be set to forward.

38765 If used as a motion command:

- 38766 1. It shall be an error if the last match references the same character in the edit buffer as the
38767 starting cursor.

- 38768 2. If any address offset is specified, the last match shall be adjusted by the specified offset as
38769 described previously.
- 38770 3. If the starting cursor is after the last match, then the locations of the starting cursor and the
38771 last match in the edit buffer shall be logically swapped.
- 38772 4. If any address offset is specified, the text region shall consist of all lines containing
38773 characters from the starting cursor to the last match line, inclusive, and any text copied to a
38774 buffer shall be in line mode.
- 38775 5. Otherwise, if the starting line is empty or the starting cursor is at or before the first non-
38776 <blank> non-<newline> of the starting line, and the last match line is empty or the last
38777 match starts at the first character of the last match line, the text region shall consist of all
38778 lines containing characters from the starting cursor to the line before the last match line,
38779 inclusive, and any text copied to a buffer shall be in line mode.
- 38780 6. Otherwise, if the last match line is empty or the last match begins at a character at or
38781 before the first non-<blank> non-<newline> of the last match line, the region of text shall
38782 be from the current cursor to the last non-<newline> of the line before the last match line,
38783 inclusive, and any text copied to a buffer shall be in character mode.
- 38784 7. Otherwise, the region of text shall be from the current cursor (inclusive), to the first
38785 character of the last match (exclusive), and any text copied to a buffer shall be in character
38786 mode.

38787 If not used as a motion command:

38788 *Current line*: If a match is found, set to the last matched line plus the address offset, if any;
38789 otherwise, unchanged.

38790 *Current column*: Set to the last column on which any portion of the first character in the last
38791 matched string is displayed, if a match is found; otherwise, unchanged.

38792 Move to First Character in Line

38793 *Synopsis*: 0 (zero)

38794 Move to the first character on the current line. The character '0' shall not be interpreted as a
38795 command if it is immediately preceded by a digit.

38796 If used as a motion command:

- 38797 1. If the cursor character is the first character in the line, it shall be an error.
- 38798 2. The text region shall be from the character before the cursor character up to and including
38799 the first character in the line.
- 38800 3. Any text copied to a buffer shall be in character mode.

38801 If not used as a motion command:

38802 *Current line*: Unchanged.

38803 *Current column*: The last column in which any portion of the first character in the line is
38804 displayed, or if the line is empty, unchanged.

38805 **Execute an ex Command**38806 *Synopsis:* :38807 Execute one or more *ex* commands.38808 If any portion of the screen other than the last line of the screen was overwritten by any *ex* command (except **shell**), *vi* shall display a message indicating that it is waiting for an input from 38809 the user, and shall then read a character. This action may also be taken for other, unspecified 38810 reasons.38811 If the next character entered is a ' : ', another *ex* command shall be accepted and executed. Any 38812 other character shall cause the screen to be refreshed and *vi* shall return to command mode.38813 *Current line:* As specified for the *ex* command.38814 *Current column:* As specified for the *ex* command.38815 **Repeat Find**38816 *Synopsis:* [*count*] ;38817 This command shall be equivalent to the last **F**, **f**, **T**, or **t** command, with the specified *count*, and 38818 with the same search character used for the last **F**, **f**, **T**, or **t** command. If there was no previous **F**, 38819 **f**, **T**, or **t** command, it shall be an error.38820 **Shift Left**38821 *Synopsis:* [*count*] < *motion*

38822 If the motion command is the < command repeated:

- 38823 1. If there are less than *count* - 1 lines after the current line in the edit buffer, it shall be an 38824 error.
- 38825 2. The text region shall be from the current line, up to and including the next *count* - 1 lines.

38826 Shift any line in the text region specified by the *count* and motion command one shiftwidth (see 38827 the *ex shiftwidth* option) toward the start of the line, as described by the *ex <* command. The 38828 unshifted lines shall be copied to the unnamed buffer in line mode.38829 *Current line:* If the motion was from the current cursor position toward the end of the edit 38830 buffer, unchanged. Otherwise, set to the first line in the edit buffer that is part of the text region 38831 specified by the motion command.38832 *Current column:* Set to non-<blank>.38833 **Shift Right**38834 *Synopsis:* [*count*] > *motion*

38835 If the motion command is the > command repeated:

- 38836 1. If there are less than *count* - 1 lines after the current line in the edit buffer, it shall be an 38837 error.
- 38838 2. The text region shall be from the current line, up to and including the next *count* - 1 lines.

38839 Shift any line with characters in the text region specified by the *count* and motion command one 38840 shiftwidth (see the *ex shiftwidth* option) away from the start of the line, as described by the *ex >* 38841 command. The unshifted lines shall be copied into the unnamed buffer in line mode.

38843 *Current line*: If the motion was from the current cursor position toward the end of the edit
38844 buffer, unchanged. Otherwise, set to the first line in the edit buffer that is part of the text region
38845 specified by the motion command.

38846 *Current column*: Set to non-<blank>.

38847 **Scan Backwards for Regular Expression**

38848 *Synopsis*: ?

38849 Scan backwards; the ? command shall be equivalent to the / command (see **Find Regular**
38850 **Expression** (on page 1005)) with the following exceptions:

- 38851 1. The input prompt shall be a '?'.
38852 2. Each search shall begin from the character before the first character of the last match (or, if
38853 it is the first search, the character before the cursor character).
38854 3. The search direction shall be from the cursor toward the beginning of the edit buffer, and
38855 the **wrapscan** edit option shall affect whether the search wraps to the end of the edit buffer
38856 and continues.
38857 4. The remembered search direction shall be set to backward.

38858 **Execute**

38859 *Synopsis*: @*buffer*

38860 If the *buffer* is specified as @, the last buffer executed shall be used. If no previous buffer has been
38861 executed, it shall be an error.

38862 Behave as if the contents of the named buffer were entered as standard input. After each line of a
38863 line-mode buffer, and all but the last line of a character mode buffer, behave as if a <newline>
38864 were entered as standard input.

38865 If an error occurs during this process, an error message shall be written, and no more characters
38866 resulting from the execution of this command shall be processed.

38867 If a *count* is specified, behave as if that count were entered as user input before the characters
38868 from the @ buffer were entered.

38869 *Current line*: As specified for the individual commands.

38870 *Current column*: As specified for the individual commands.

38871 **Reverse Case**

38872 *Synopsis*: [*count*] ~

38873 Reverse the case of the current character and the next *count* -1 characters, such that lowercase
38874 characters that have uppercase counterparts shall be changed to uppercase characters, and
38875 uppercase characters that have lowercase counterparts shall be changed to lowercase characters,
38876 as prescribed by the current locale. No other characters shall be affected by this command.

38877 If there are less than *count* -1 characters after the cursor in the edit buffer, *count* shall be adjusted
38878 to the number of characters after the cursor in the edit buffer minus 1.

38879 For the purposes of this command, the next character after the last non-<newline> on the line
38880 shall be the next character in the edit buffer.

38881 *Current line*: Set to the line including the (*count*-1)th character after the cursor.

38882 *Current column*: Set to the last column in which any portion of the (*count*-1)th character after the
38883 cursor is displayed.

38884 **Append**

38885 *Synopsis:* [*count*] a

38886 Enter text input mode after the current cursor position. No characters already in the edit buffer
38887 shall be affected by this command. A *count* shall cause the input text to be appended *count* -1
38888 more times to the end of the input.

38889 *Current line/column*: As specified for the text input commands (see **Input Mode Commands in vi**
38890 (on page 1023)).

38891 **Append at End-of-Line**

38892 *Synopsis:* [*count*] A

38893 This command shall be equivalent to the vi command:

38894 \$ [*count*] a

38895 (see **Append**).

38896 **Move Backward to Preceding Word**

38897 *Synopsis:* [*count*] b

38898 With the exception that words are used as the delimiter instead of bigwords, this command shall
38899 be equivalent to the B command.

38900 **Move Backward to Preceding Bigword**

38901 *Synopsis:* [*count*] B

38902 If the edit buffer is empty or the cursor is on the first character of the edit buffer, it shall be an
38903 error. If less than *count* bigwords begin between the cursor and the start of the edit buffer, *count*
38904 shall be adjusted to the number of bigword beginnings between the cursor and the start of the
38905 edit buffer.

38906 If used as a motion command:

38907 1. The text region shall be from the first character of the *count*th previous bigword beginning
38908 up to but not including the cursor character.

38909 2. Any text copied to a buffer shall be in character mode.

38910 If not used as a motion command:

38911 *Current line*: Set to the line containing the *current column*.

38912 *Current column*: Set to the last column upon which any part of the first character of the *count*th
38913 previous bigword is displayed.

38914 **Change**

38915 *Synopsis:* `[buffer][count] c motion`

38916 If the motion command is the **c** command repeated:

- 38917 1. The buffer text shall be in line mode.
- 38918 2. If there are less than *count* – 1 lines after the current line in the edit buffer, it shall be an error.
- 38920 3. The text region shall be from the current line up to and including the next *count* – 1 lines.

38921 Otherwise, the buffer text mode and text region shall be as specified by the motion command.

38922 The replaced text shall be copied into *buffer*, if specified, and into the unnamed buffer. If the text to be replaced contains characters from more than a single line, or the buffer text is in line mode, the replaced text shall be copied into the numeric buffers as well.

38925 If the buffer text is in line mode:

- 38926 1. Any lines that contain characters in the region shall be deleted, and the editor shall enter text input mode at the beginning of a new line which shall replace the first line deleted.
- 38928 2. If the **autoindent** edit option is set, **autoindent** characters equal to the **autoindent** characters on the first line deleted shall be inserted as if entered by the user.

38930 Otherwise, if characters from more than one line are in the region of text:

- 38931 1. The text shall be deleted.
- 38932 2. Any text remaining in the last line in the text region shall be appended to the first line in the region, and the last line in the region shall be deleted.
- 38934 3. The editor shall enter text input mode after the last character not deleted from the first line in the text region, if any; otherwise, on the first column of the first line in the region.

38936 Otherwise:

- 38937 1. If the glyph for '\$' is smaller than the region, the end of the region shall be marked with a '\$'.
- 38939 2. The editor shall enter text input mode, overwriting the region of text.

38940 *Current line/column:* As specified for the text input commands (see **Input Mode Commands in vi** (on page 1023)).

38942 **Change to End-of-Line**

38943 *Synopsis:* `[buffer][count] C`

38944 This command shall be equivalent to the **vi** command:

38945 `[buffer][count] c$`

38946 See the **c** command.

38947 **Delete**

38948 *Synopsis:* `[buffer][count] d motion`

38949 If the motion command is the **d** command repeated:

38950 1. The buffer text shall be in line mode.

38951 2. If there are less than *count* – 1 lines after the current line in the edit buffer, it shall be an
38952 error.

38953 3. The text region shall be from the current line up to and including the next *count* – 1 lines.

38954 Otherwise, the buffer text mode and text region shall be as specified by the motion command.

38955 If in open mode, and the current line is deleted, and the line remains on the display, an '@'
38956 character shall be displayed as the first glyph of that line.

38957 Delete the region of text into *buffer*, if specified, and into the unnamed buffer. If the text to be
38958 deleted contains characters from more than a single line, or the buffer text is in line mode, the
38959 deleted text shall be copied into the numeric buffers, as well.

38960 *Current line:* Set to the first text region line that appears in the edit buffer, unless that line has
38961 been deleted, in which case it shall be set to the last line in the edit buffer, or line 1 if the edit
38962 buffer is empty.

38963 *Current column:*

38964 1. If the line is empty, set to column position 1.

38965 2. Otherwise, if the buffer text is in line mode or the motion was from the cursor toward the
38966 end of the edit buffer:

38967 a. If a character from the current line is displayed in the current column, set to the last
38968 column that displays any portion of that character.

38969 b. Otherwise, set to the last column in which any portion of any character in the line is
38970 displayed.

38971 3. Otherwise, if a character is displayed in the column that began the text region, set to the
38972 last column that displays any portion of that character.

38973 4. Otherwise, set to the last column in which any portion of any character in the line is
38974 displayed.

38975 **Delete to End-of-Line**

38976 *Synopsis:* `[buffer] D`

38977 Delete the text from the current position to the end of the current line; equivalent to the vi
38978 command:

38979 `[buffer] d$`

38980 **Move to End-of-Word**

38981 *Synopsis:* [*count*] e

38982 With the exception that words are used instead of bigwords as the delimiter, this command shall
38983 be equivalent to the E command.

38984 **Move to End-of-Bigword**

38985 *Synopsis:* [*count*] E

38986 If the edit buffer is empty it shall be an error. If less than *count* bigwords end between the cursor
38987 and the end of the edit buffer, *count* shall be adjusted to the number of bigword endings between
38988 the cursor and the end of the edit buffer.

38989 If used as a motion command:

1. The text region shall be from the last character of the *count*th next bigword up to and including the cursor character.

2. Any text copied to a buffer shall be in character mode.

38993 If not used as a motion command:

38994 *Current line:* Set to the line containing the current column.

38995 *Current column:* Set to the last column upon which any part of the last character of the *count*th
38996 next bigword is displayed.

39000 **Find Character in Current Line (Forward)**

39001 *Synopsis:* [*count*] f *character*

39002 It shall be an error if *count* occurrences of the character do not occur after the cursor in the line.

39003 If used as a motion command:

1. The text range shall be from the cursor character up to and including the *count*th occurrence of the specified character after the cursor.

2. Any text copied to a buffer shall be in character mode.

39004 If not used as a motion command:

39005 *Current line:* Unchanged.

39006 *Current column:* Set to the last column in which any portion of the *count*th occurrence of the
39007 specified character after the cursor appears in the line.

39008 **Find Character in Current Line (Reverse)**

39009 *Synopsis:* [*count*] F *character*

39010 It shall be an error if *count* occurrences of the character do not occur before the cursor in the line.

39011 If used as a motion command:

1. The text region shall be from the *count*th occurrence of the specified character before the cursor, up to, but not including the cursor character.

2. Any text copied to a buffer shall be in character mode.

39015 If not used as a motion command:

- 39016 *Current line*: Unchanged.
- 39017 *Current column*: Set to the last column in which any portion of the *count*th occurrence of the specified character before the cursor appears in the line.
- 39019 **Move to Line**
- 39020 *Synopsis:* [*count*] G
- 39021 If *count* is not specified, it shall default to the last line of the edit buffer. If *count* is greater than the last line of the edit buffer, it shall be an error.
- 39023 If used as a motion command:
- 39024 1. The text region shall be from the cursor line up to and including the specified line.
 - 39025 2. Any text copied to a buffer shall be in line mode.
- 39026 If not used as a motion command:
- 39027 *Current line*: Set to *count* if *count* is specified; otherwise, the last line.
- 39028 *Current column*: Set to non-<blank>.
- 39029 **Move to Top of Screen**
- 39030 *Synopsis:* [*count*] H
- 39031 If the beginning of the line *count* greater than the first line of which any portion appears on the display does not exist, it shall be an error.
- 39033 If used as a motion command:
- 39034 1. If in open mode, the text region shall be the current line.
 - 39035 2. Otherwise, the text region shall be from the starting line up to and including (the first line of the display + *count* - 1).
 - 39037 3. Any text copied to a buffer shall be in line mode.
- 39038 If not used as a motion command:
- 39039 If in open mode, this command shall set the current column to non-<blank> and do nothing else.
- 39040 Otherwise, it shall set the current line and current column as follows.
- 39041 *Current line*: Set to (the first line of the display + *count* - 1).
- 39042 *Current column*: Set to non-<blank>.
- 39043 **Insert Before Cursor**
- 39044 *Synopsis:* [*count*] i
- 39045 Enter text input mode before the current cursor position. No characters already in the edit buffer shall be affected by this command. A *count* shall cause the input text to be appended *count* - 1 more times to the end of the input.
- 39048 *Current line/column*: As specified for the text input commands (see **Input Mode Commands in vi** (on page 1023)).

39050 **Insert at Beginning of Line**

39051 *Synopsis:* [*count*] *I*

39052 This command shall be equivalent to the *vi* command *^[count]i*.

39053 **Join**

39054 *Synopsis:* [*count*] *J*

39055 If the current line is the last line in the edit buffer, it shall be an error.

39056 This command shall be equivalent to the *ex join* command with no addresses, and an *ex* command *count* value of 1 if *count* was not specified or if a *count* of 1 was specified, and an *ex* command *count* value of *count* -1 for any other value of *count*, except that the current line and column shall be set as follows.

39060 *Current line:* Unchanged.

39061 *Current column:* The last column in which any portion of the character following the last character in the initial line is displayed, or the last non-<newline> in the line if no characters were appended.

39064 **Move to Bottom of Screen**

39065 *Synopsis:* [*count*] *L*

39066 If the beginning of the line *count* less than the last line of which any portion appears on the display does not exist, it shall be an error.

39068 If used as a motion command:

1. If in open mode, the text region shall be the current line.
2. Otherwise, the text region shall include all lines from the starting cursor line to (the last line of the display -(*count* -1)).
3. Any text copied to a buffer shall be in line mode.

39073 If not used as a motion command:

1. If in open mode, this command shall set the current column to non-<blank> and do nothing else.
2. Otherwise, it shall set the current line and current column as follows.

39077 *Current line:* Set to (the last line of the display -(*count* -1)).

39078 *Current column:* Set to non-<blank>.

39079 **Mark Position**

39080 *Synopsis:* m *letter*

39081 This command shall be equivalent to the *ex mark* command with the specified character as an argument.

39083 **Move to Middle of Screen**39084 *Synopsis:* M

39085 The middle line of the display shall be calculated as follows:

39086 (the top line of the display) + (((number of lines displayed) +1) /2) -1

39087 If used as a motion command:

- 39088 1. If in open mode, the text region shall be the current line.
- 39089 2. Otherwise, the text region shall include all lines from the starting cursor line up to and
39090 including the middle line of the display.
- 39091 3. Any text copied to a buffer shall be in line mode.

39092 If not used as a motion command:

39093 If in open mode, this command shall set the current column to non-<blank> and do nothing else.

39094 Otherwise, it shall set the current line and current column as follows.

39095 *Current line:* Set to the middle line of the display.39096 *Current column:* Set to non-<blank>.39097 **Repeat Regular Expression Find (Forward)**39098 *Synopsis:* n39099 If the remembered search direction was forward, the n command shall be equivalent to the vi /
39100 command with no characters entered by the user. Otherwise, it shall be equivalent to the vi ?
39101 command with no characters entered by the user.39102 If the n command is used as a motion command for the ! command, the editor shall not enter
39103 text input mode on the last line on the screen, and shall behave as if the user entered a single '!'
39104 character as the text input.39105 **Repeat Regular Expression Find (Reverse)**39106 *Synopsis:* N39107 Scan for the next match of the last pattern given to / or ?, but in the reverse direction; this is the
39108 reverse of n.39109 If the remembered search direction was forward, the N command shall be equivalent to the vi ?
39110 command with no characters entered by the user. Otherwise, it shall be equivalent to the vi /
39111 command with no characters entered by the user. If the N command is used as a motion
39112 command for the ! command, the editor shall not enter text input mode on the last line on the
39113 screen, and shall behave as if the user entered a single ! character as the text input.39114 **Insert Empty Line Below**39115 *Synopsis:* o39116 Enter text input mode in a new line appended after the current line. A count shall cause the input
39117 text to be appended count -1 more times to the end of the already added text, each time starting
39118 on a new, appended line.39119 *Current line/column:* As specified for the text input commands (see **Input Mode Commands in vi**
39120 (on page 1023)).

39121 **Insert Empty Line Above**

39122 *Synopsis:* \circ

39123 Enter text input mode in a new line inserted before the current line. A *count* shall cause the input
39124 text to be appended *count* –1 more times to the end of the already added text, each time starting
39125 on a new, appended line.

39126 *Current line/column:* As specified for the text input commands (see **Input Mode Commands in vi**
39127 (on page 1023)).

39128 **Put from Buffer Following**

39129 *Synopsis:* $[buffer] \ p$

39130 If no *buffer* is specified, the unnamed buffer shall be used.

39131 If the buffer text is in line mode, the text shall be appended below the current line, and each line
39132 of the buffer shall become a new line in the edit buffer. A *count* shall cause the buffer text to be
39133 appended *count* –1 more times to the end of the already added text, each time starting on a new,
39134 appended line.

39135 If the buffer text is in character mode, the text shall be appended into the current line after the
39136 cursor, and each line of the buffer other than the first and last shall become a new line in the edit
39137 buffer. A *count* shall cause the buffer text to be appended *count* –1 more times to the end of the
39138 already added text, each time starting after the last added character.

39139 *Current line:* If the buffer text is in line mode, set the line to line +1; otherwise, unchanged.

39140 *Current column:* If the buffer text is in line mode:

- 39141 1. If there is a non-<blank> in the first line of the buffer, set to the last column on which any
39142 portion of the first non-<blank> in the line is displayed.
- 39143 2. If there is no non-<blank> in the first line of the buffer, set to the last column on which any portion
39144 of the last non-<newline> in the first line of the buffer is displayed.

39145 If the buffer text is in character mode:

- 39146 1. If the text in the buffer is from more than a single line, then set to the last column on which
39147 any portion of the first character from the buffer is displayed.
- 39148 2. Otherwise, if the buffer is the unnamed buffer, set to the last column on which any portion
39149 of the last character from the buffer is displayed.
- 39150 3. Otherwise, set to the first column on which any portion of the first character from the
39151 buffer is displayed.

39152 **Put from Buffer Before**

39153 *Synopsis:* $[buffer] \ P$

39154 If no *buffer* is specified, the unnamed buffer shall be used.

39155 If the buffer text is in line mode, the text shall be inserted above the current line, and each line of
39156 the buffer shall become a new line in the edit buffer. A *count* shall cause the buffer text to be
39157 appended *count* –1 more times to the end of the already added text, each time starting on a new,
39158 appended line.

39159 If the buffer text is in character mode, the text shall be inserted into the current line before the
39160 cursor, and each line of the buffer other than the first and last shall become a new line in the edit
39161 buffer. A *count* shall cause the buffer text to be appended *count* –1 more times to the end of the

39162 already added text, each time starting after the last added character.
39163 *Current line*: Unchanged.
39164 *Current column*: If the buffer text is in line mode:
39165 1. If there is a non-<blank> in the first line of the buffer, set to the last column on which any
39166 portion of that character is displayed.
39167 2. If there is no non-<blank> in the first line of the buffer, set to the last column on which any
39168 portion of the last non-<newline> in the first line of the buffer is displayed.
39169 If the buffer text is in character mode:
39170 1. If the buffer is the unnamed buffer, set to the last column on which any portion of the last
39171 character from the buffer is displayed.
39172 2. Otherwise, set to the first column on which any portion of the first character from the
39173 buffer is displayed.

39174 **Enter ex Mode**

39175 *Synopsis*: Q
39176 Leave visual or open mode and enter *ex* command mode.
39177 *Current line*: Unchanged.
39178 *Current column*: Unchanged.

39179 **Replace Character**

39180 *Synopsis*: [*count*] r *character*
39181 Replace the *count* characters at and after the cursor with the specified character. If there are less
39182 than *count* non-<newline>s at and after the cursor on the line, it shall be an error.
39183 If character is <control>-V, any next character other than the <newline> shall be stripped of any
39184 special meaning and used as a literal character.
39185 If character is <ESC>, no replacement shall be made and the current line and current column
39186 shall be unchanged.
39187 If character is <carriage-return> or <newline>, *count* new lines shall be appended to the current
39188 line. All but the last of these lines shall be empty. *count* characters at and after the cursor shall be
39189 discarded, and any remaining characters after the cursor in the current line shall be moved to the
39190 last of the new lines. If the **autoindent** edit option is set, they shall be preceded by the same
39191 number of **autoindent** characters found on the line from which the command was executed.
39192 *Current line*: Unchanged unless the replacement character is a <carriage-return> or <newline>,
39193 in which case it shall be set to line + *count*.
39194 *Current column*: Set to the last column position on which a portion of the last replaced character
39195 is displayed, or if the replacement character caused new lines to be created, set to non-<blank>.

39196 **Replace Characters**

39197 *Synopsis:* R

39198 Enter text input mode at the current cursor position possibly replacing text on the current line. A
39199 *count* shall cause the input text to be appended *count* – 1 more times to the end of the input.

39200 *Current line/column:* As specified for the text input commands (see **Input Mode Commands in vi**
39201 (on page 1023)).

39202 **Substitute Character**

39203 *Synopsis:* [buffer][count] s

39204 This command shall be equivalent to the vi command:

39205 [buffer][count] c<space>

39206 **Substitute Lines**

39207 *Synopsis:* [buffer][count] S

39208 This command shall be equivalent to the vi command:

39209 [buffer][count] c_

39210 **Move Cursor to Before Character (Forward)**

39211 *Synopsis:* [count] t character

39212 It shall be an error if *count* occurrences of the character do not occur after the cursor in the line.

39213 If used as a motion command:

- 39214 1. The text region shall be from the cursor up to but not including the *count*th occurrence of
39215 the specified character after the cursor.
- 39216 2. Any text copied to a buffer shall be in character mode.

39217 If not used as a motion command:

39218 *Current line:* Unchanged.

39219 *Current column:* Set to the last column in which any portion of the character before the *count*th
39220 occurrence of the specified character after the cursor appears in the line.

39221 **Move Cursor to After Character (Reverse)**

39222 *Synopsis:* [count] T character

39223 It shall be an error if *count* occurrences of the character do not occur before the cursor in the line.

39224 If used as a motion command:

- 39225 1. If the character before the cursor is the specified character, it shall be an error.
- 39226 2. The text region shall be from the character before the cursor up to but not including the
39227 *count*th occurrence of the specified character before the cursor.
- 39228 3. Any text copied to a buffer shall be in character mode.

39229 If not used as a motion command:

39230 *Current line*: Unchanged.
39231 *Current column*: Set to the last column in which any portion of the character after the *count*th
39232 occurrence of the specified character before the cursor appears in the line.

39233 Undo

39234 *Synopsis:* u

39235 This command shall be equivalent to the **ex undo** command except that the current line and
39236 current column shall be set as follows:

39237 *Current line*: Set to the first line added or changed if any; otherwise, move to the line preceding
39238 any deleted text if one exists; otherwise, move to line 1.

39239 *Current column*: If undoing an **ex** command, set to the first non-<blank>.

39240 Otherwise, if undoing a text input command:

- 39241 1. If the command was a **C**, **c**, **O**, **o**, **R**, **S**, or **s** command, the current column shall be set to the
39242 value it held when the text input command was entered.
- 39243 2. Otherwise, set to the last column in which any portion of the first character after the
39244 deleted text is displayed, or, if no non-<newline>s follow the text deleted from this line, set
39245 to the last column in which any portion of the last non-<newline> in the line is displayed,
39246 or 1 if the line is empty.

39247 Otherwise, if a single line was modified (that is, not added or deleted) by the **u** command:

- 39248 1. If text was added or changed, set to the last column in which any portion of the first
39249 character added or changed is displayed.
- 39250 2. If text was deleted, set to the last column in which any portion of the first character after
39251 the deleted text is displayed, or, if no non-<newline>s follow the deleted text, set to the last
39252 column in which any portion of the last non-<newline> in the line is displayed, or 1 if the
39253 line is empty.

39254 Otherwise, set to non-<blank>.

39255 Undo Current Line

39256 *Synopsis:* U

39257 Restore the current line to its state immediately before the most recent time that it became the
39258 current line.

39259 *Current line*: Unchanged.

39260 *Current column*: Set to the first column in the line in which any portion of the first character in
39261 the line is displayed.

39262 Move to Beginning of Word

39263 *Synopsis:* [*count*] w

39264 With the exception that words are used as the delimiter instead of bigwords, this command shall
39265 be equivalent to the **W** command.

39266 **Move to Beginning of Bigword**39267 *Synopsis:* [count] w

39268 If the edit buffer is empty, it shall be an error. If there are less than *count* bigwords between the
39269 cursor and the end of the edit buffer, *count* shall be adjusted to move the cursor to the last
39270 bigword in the edit buffer.

39271 If used as a motion command:

- 39272 1. If the associated command is c, *count* is 1, and the cursor is on a <blank>, the region of text
39273 shall be the current character and no further action shall be taken.
- 39274 2. If there are less than *count* bigwords between the cursor and the end of the edit buffer, then
39275 the command shall succeed, and the region of text shall include the last character of the
39276 edit buffer.
- 39277 3. If there are <blank>s or an end-of-line that precede the *count*th bigword, and the associated
39278 command is c, the region of text shall be up to and including the last character before the
39279 preceding <blank>s or end-of-line.
- 39280 4. If there are <blank>s or an end-of-line that precede the bigword, and the associated
39281 command is d or y, the region of text shall be up to and including the last <blank> before
39282 the start of the bigword or end-of-line.
- 39283 5. Any text copied to a buffer shall be in character mode.

39284 If not used as a motion command:

- 39285 1. If the cursor is on the last character of the edit buffer, it shall be an error.

39286 *Current line:* Set to the line containing the current column.

39287 *Current column:* Set to the last column in which any part of the first character of the *count*th next
39288 bigword is displayed.

39289 **Delete Character at Cursor**39290 *Synopsis:* [buffer][count] x

39291 Delete the *count* characters at and after the current character into *buffer*, if specified, and into the
39292 unnamed buffer.

39293 If the line is empty, it shall be an error. If there are less than *count* non-<newline>s at and after
39294 the cursor on the current line, *count* shall be adjusted to the number of non-<newline>s at and
39295 after the cursor.

39296 *Current line:* Unchanged.

39297 *Current column:* If the line is empty, set to column position 1. Otherwise, if there were *count* or
39298 less non-<newline>s at and after the cursor on the current line, set to the last column that
39299 displays any part of the last non-<newline> of the line. Otherwise, unchanged.

39300 **Delete Character Before Cursor**

39301 *Synopsis:* `[buffer][count] X`

39302 Delete the *count* characters before the current character into *buffer*, if specified, and into the
39303 unnamed buffer.

39304 If there are no characters before the current character on the current line, it shall be an error. If
39305 there are less than *count* previous characters on the current line, *count* shall be adjusted to the
39306 number of previous characters on the line.

39307 *Current line:* Unchanged.

39308 *Current column:* Set to (current column – the width of the deleted characters).

39309 **Yank**

39310 *Synopsis:* `[buffer][count] y motion`

39311 Copy (yank) the region of text into *buffer*, if specified, and into the unnamed buffer.

39312 If the motion command is the *y* command repeated:

39313 1. The buffer shall be in line mode.

39314 2. If there are less than *count* – 1 lines after the current line in the edit buffer, it shall be an
39315 error.

39316 3. The text region shall be from the current line up to and including the next *count* – 1 lines.

39317 Otherwise, the buffer text mode and text region shall be as specified by the motion command.

39318 *Current line:* If the motion was from the current cursor position toward the end of the edit
39319 buffer, unchanged. Otherwise, set to the first line in the edit buffer that is part of the text region
39320 specified by the motion command.

39321 *Current column:*

39322 1. If the motion was from the current cursor position toward the end of the edit buffer,
39323 unchanged.

39324 2. Otherwise, if the current line is empty, set to column position 1.

39325 3. Otherwise, set to the last column that displays any part of the first character in the file that
39326 is part of the text region specified by the motion command.

39327 **Yank Current Line**

39328 *Synopsis:* `[buffer][count] Y`

39329 This command shall be equivalent to the *vi* command:

39330 `[buffer][count] y_`

39331 **Redraw Window**

39332 If in open mode, the **z** command shall have the Synopsis:

39333 *Synopsis:* [*count*] **z**

39334 If *count* is not specified, it shall default to the **window** edit option -1. The **z** command shall be equivalent to the **ex z** command, with a type character of = and a *count* of *count*-2, except that the current line and current column shall be set as follows, and the **window** edit option shall not be affected. If the calculation for the *count* argument would result in a negative number, the *count* argument to the **ex z** command shall be zero. A blank line shall be written after the last line is written.

39340 *Current line:* Unchanged.

39341 *Current column:* Unchanged.

39342 If not in open mode, the **z** command shall have the following Synopsis:

39343 *Synopsis:* [*line*] **z** [*count*] *character*

39344 If *line* is not specified, it shall default to the current line. If *line* is specified, but is greater than the number of lines in the edit buffer, it shall default to the number of lines in the edit buffer.

39346 If *count* is specified, the value of the **window** edit option shall be set to *count* (as described in the **ex window** command), and the screen shall be redrawn.

39348 *line* shall be placed as specified by the following characters:

39349 <newline>, <carriage-return>

39350 Place the beginning of the line on the first line of the display.

39351 · Place the beginning of the line in the center of the display. The middle line of the display shall be calculated as described for the **M** command.

39353 – Place an unspecified portion of the line on the last line of the display.

39354 + If *line* was specified, equivalent to the <newline> case. If *line* was not specified, display a screen where the first line of the display shall be (current last line) +1. If there are no lines after the last line in the display, it shall be an error.

39357 ^ If *line* was specified, display a screen where the last line of the display shall contain an unspecified portion of the first line of a display that had an unspecified portion of the specified line on the last line of the display. If this calculation results in a line before the beginning of the edit buffer, display the first screen of the edit buffer.

39361 Otherwise, display a screen where the last line of the display shall contain an unspecified portion of (current first line -1). If this calculation results in a line before the beginning of the edit buffer, it shall be an error.

39364 *Current line:* If *line* and the '^' character were specified:

39365 1. If the first screen was displayed as a result of the command attempting to display lines before the beginning of the edit buffer: if the first screen was already displayed, unchanged; otherwise, set to (current first line -1).

39368 2. Otherwise, set to the last line of the display.

39369 If *line* and the '+' character were specified, set to the first line of the display.

39370 Otherwise, if *line* was specified, set to *line*.

39371 Otherwise, unchanged.

39372 *Current column:* Set to non-<blank>.

39373 **Exit**

39374 *Synopsis:* ZZ

39375 This command shall be equivalent to the **ex xit** command with no addresses, trailing !, or
39376 filename (see the **ex xit** command).

39377 **Input Mode Commands in vi**

39378 In text input mode, the current line shall consist of zero or more of the following categories, plus
39379 the terminating <newline>:

- 39380 1. Characters preceding the text input entry point

39381 Characters in this category shall not be modified during text input mode.

- 39382 2. **autoindent** characters

39383 **autoindent** characters shall be automatically inserted into each line that is created in text
39384 input mode, either as a result of entering a <newline> or <carriage-return> while in text
39385 input mode, or as an effect of the command itself; for example, **O** or **o** (see the **ex**
39386 **autoindent** command), as if entered by the user.

39387 It shall be possible to erase **autoindent** characters with the <control>-D command; it is
39388 unspecified whether they can be erased by <control>-H, <control>-U, and <control>-W
39389 characters. Erasing any **autoindent** character turns the glyph into erase-columns and
39390 deletes the character from the edit buffer, but does not change its representation on the
39391 screen.

- 39392 3. Text input characters

39393 Text input characters are the characters entered by the user. Erasing any text input
39394 character turns the glyph into erase-columns and deletes the character from the edit buffer,
39395 but does not change its representation on the screen.

39396 Each text input character entered by the user (that does not have a special meaning) shall
39397 be treated as follows:

- 39398 a. The text input character shall be appended to the last character in the edit buffer
39399 from the first, second, or third categories.

- 39400 b. If there are no erase-columns on the screen, the text input command was the **R**
39401 command, and characters in the fifth category from the original line follow the
39402 cursor, the next such character shall be deleted from the edit buffer. If the **slowopen**
39403 edit option is not set, the corresponding glyph on the screen shall become erase-
39404 columns.

- 39405 c. If there are erase-columns on the screen, as many columns as they occupy, or as are
39406 necessary, shall be overwritten to display the text input character. (If only part of a
39407 multi-column glyph is overwritten, the remainder shall be left on the screen, and
39408 continue to be treated as erase-columns; it is unspecified whether the remainder of
39409 the glyph is modified in any way.)

- 39410 d. If additional display line columns are needed to display the text input character:

- 39411 1. If the **slowopen** edit option is set, the text input characters shall be displayed
39412 on subsequent display line columns, overwriting any characters displayed in

- 39413 those columns.
- 39414 2. Otherwise, any characters currently displayed on or after the column on the
39415 display line where the text input character is to be displayed shall be pushed
39416 ahead the number of display line columns necessary to display the rest of the
39417 text input character.
- 39418 4. Erase-columns
- 39419 Erase-columns are not logically part of the edit buffer, appearing only on the screen, and
39420 may be overwritten on the screen by subsequent text input characters. When text input
39421 mode ends, all erase-columns shall no longer appear on the screen.
- 39422 Erase-columns are initially the region of text specified by the **c** command (see **Change** (on
39423 page 1010)); however, erasing **autoindent** or text input characters causes the glyphs of the
39424 erased characters to be treated as erase-columns.
- 39425 5. Characters following the text region for the **c** command, or the text input entry point for all
39426 other commands
- 39427 Characters in this category shall not be modified during text input mode, except as
39428 specified in category 3.b. for the **R** text input command, or as <blank>s deleted when a
39429 <newline> or <carriage-return> is entered.
- 39430 It is unspecified whether it is an error to attempt to erase past the beginning of a line that was
39431 created by the entry of a <newline> or <carriage-return> during text input mode. If it is not an
39432 error, the editor shall behave as if the erasing character was entered immediately after the last
39433 text input character entered on the previous line, and all of the non-<newline>s on the current
39434 line shall be treated as erase-columns.
- 39435 When text input mode is entered, or after a text input mode character is entered (except as
39436 specified for the special characters below), the cursor shall be positioned as follows:
- 39437 1. On the first column that displays any part of the first erase-column, if one exists
- 39438 2. Otherwise, if the **slowopen** edit option is set, on the first display line column after the last
39439 character in the first, second, or third categories, if one exists
- 39440 3. Otherwise, the first column that displays any part of the first character in the fifth category,
39441 if one exists
- 39442 4. Otherwise, the display line column after the last character in the first, second, or third
39443 categories, if one exists
- 39444 5. Otherwise, on column position 1
- 39445 The characters that are updated on the screen during text input mode are unspecified, other than
39446 that the last text input character shall always be updated, and, if the **slowopen** edit option is not
39447 set, the current cursor character shall always be updated.
- 39448 The following specifications are for command characters entered during text input mode.

39449 **NUL**

39450 *Synopsis:* NUL

39451 If the first character of the text input is a NUL, the most recently input text shall be input as if
39452 entered by the user, and then text input mode shall be exited. The text shall be input literally;
39453 that is, characters are neither macro or abbreviation expanded, nor are any characters interpreted
39454 in any special manner. It is unspecified whether implementations shall support more than 256
39455 bytes of remembered input text.

39456 **<control>-D**

39457 *Synopsis:* <control>-D

39458 The <control>-D character shall have no special meaning when in text input mode for a line-
39459 oriented command (see **Command Descriptions in vi** (on page 989)).

39460 This command need not be supported on block-mode terminals.

39461 If the cursor does not follow an **autoindent** character, or an **autoindent** character and a '0' or
39462 '^' character:

- 39463 1. If the cursor is in column position 1, the <control>-D character shall be discarded and no
39464 further action taken.
- 39465 2. Otherwise, the <control>-D character shall have no special meaning.

39466 If the last input character was a '0', the cursor shall be moved to column position 1.

39467 Otherwise, if the last input character was a '^', the cursor shall be moved to column position 1.
39468 In addition, the **autoindent** level for the next input line shall be derived from the same line from
39469 which the **autoindent** level for the current input line was derived.

39470 Otherwise, the cursor shall be moved back to the column after the previous shiftwidth (see the
39471 **ex shiftwidth** command) boundary.

39472 All of the glyphs on columns between the starting cursor position and (inclusively) the ending
39473 cursor position shall become erase-columns as described in **Input Mode Commands in vi** (on
39474 page 1023).

39475 *Current line:* Unchanged.

39476 *Current column:* Set to 1 if the <control>-D was preceded by a '^' or '0'; otherwise, set to
39477 (column -1) -((column -2) % **shiftwidth**).

39478 **<control>-H**

39479 *Synopsis:* <control>-H

39480 If in text input mode for a line-oriented command, and there are no characters to erase, text
39481 input mode shall be terminated, no further action shall be done for this command, and the
39482 current line and column shall be unchanged.

39483 If there are characters other than **autoindent** characters that have been input on the current line
39484 before the cursor, the cursor shall move back one character.

39485 Otherwise, if there are **autoindent** characters on the current line before the cursor, it is
39486 implementation-defined whether the <control>-H command is an error or if the cursor moves
39487 back one **autoindent** character.

39488 Otherwise, if the cursor is in column position 1 and there are previous lines that have been input,
39489 it is implementation-defined whether the <control>-H command is an error or if it is equivalent

39490 to entering <control>-H after the last input character on the previous input line.
39491 Otherwise, it shall be an error.

39492 All of the glyphs on columns between the starting cursor position and (inclusively) the ending
39493 cursor position shall become erase-columns as described in **Input Mode Commands in vi** (on
39494 page 1023).

39495 The current erase character (see *stty*) shall cause an equivalent action to the <control>-H
39496 command, unless the previously inserted character was a backslash, in which case it shall be as
39497 if the literal current erase character had been inserted instead of the backslash.

39498 *Current line*: Unchanged, unless previously input lines are erased, in which case it shall be set to
39499 line -1.

39500 *Current column*: Set to the first column that displays any portion of the character backed up
39501 over.

39502 <newline>

39503 *Synopsis*: <newline>
39504 <carriage-return>
39505 <control>-J
39506 <control>-M

39507 If input was part of a line-oriented command, text input mode shall be terminated and the
39508 command shall continue execution with the input provided.

39509 Otherwise, terminate the current line. If there are no characters other than **autoindent** characters
39510 on the line, all characters on the line shall be discarded. Otherwise, it is unspecified whether the
39511 **autoindent** characters in the line are modified by entering these characters.

39512 Continue text input mode on a new line appended after the current line. If the **slowopen** edit
39513 option is set, the lines on the screen below the current line shall not be pushed down, but the
39514 first of them shall be cleared and shall appear to be overwritten. Otherwise, the lines of the
39515 screen below the current line shall be pushed down.

39516 If the **autoindent** edit option is set, an appropriate number of **autoindent** characters shall be
39517 added as a prefix to the line as described by the *ex autoindent* edit option.

39518 All columns after the cursor that are erase-columns (as described in **Input Mode Commands in**
39519 **vi** (on page 1023)) shall be discarded.

39520 If the **autoindent** edit option is set, all <blank>s immediately following the cursor shall be
39521 discarded.

39522 All remaining characters after the cursor shall be transferred to the new line, positioned after any
39523 **autoindent** characters.

39524 *Current line*: Set to current line +1.

39525 *Current column*: Set to the first column that displays any portion of the first character after the
39526 **autoindent** characters on the new line, if any, or the first column position after the last
39527 **autoindent** character, if any, or column position 1.

39528 **<control>-T**

39529 *Synopsis:* **<control>-T**

39530 The **<control>-T** character shall have no special meaning when in text input mode for a line-oriented command (see **Command Descriptions in vi** (on page 989)).

39532 This command need not be supported on block-mode terminals.

39533 Behave as if the user entered the minimum number of **<blank>**s necessary to move the cursor forward to the column position after the next **shiftwidth** (see the **ex shiftwidth** command) boundary.

39536 *Current line:* Unchanged.

39537 *Current column:* Set to *column* + **shiftwidth** – ((*column* – 1) % **shiftwidth**).

39538 **<control>-U**

39539 *Synopsis:* **<control>-U**

39540 If there are characters other than **autoindent** characters that have been input on the current line before the cursor, the cursor shall move to the first character input after the **autoindent** characters.

39543 Otherwise, if there are **autoindent** characters on the current line before the cursor, it is implementation-defined whether the **<control>-U** command is an error or if the cursor moves to the first column position on the line.

39546 Otherwise, if the cursor is in column position 1 and there are previous lines that have been input, it is implementation-defined whether the **<control>-U** command is an error or if it is equivalent to entering **<control>-U** after the last input character on the previous input line.

39549 Otherwise, it shall be an error.

39550 All of the glyphs on columns between the starting cursor position and (inclusively) the ending cursor position shall become erase-columns as described in **Input Mode Commands in vi** (on page 1023).

39553 The current *kill* character (see **stty**) shall cause an equivalent action to the **<control>-U** command, unless the previously inserted character was a backslash, in which case it shall be as if the literal current *kill* character had been inserted instead of the backslash.

39556 *Current line:* Unchanged, unless previously input lines are erased, in which case it shall be set to line –1.

39558 *Current column:* Set to the first column that displays any portion of the last character backed up over.

39560 **<control>-V**

39561 *Synopsis:* **<control>-V**
 <control>-Q

39563 Allow the entry of any subsequent character, other than **<control>-J** or the **<newline>**, as a literal character, removing any special meaning that it may have to the editor in text input mode. If a **<control>-V** or **<control>-Q** is entered before a **<control>-J** or **<newline>**, the **<control>-V** or **<control>-Q** character shall be discarded, and the **<control>-J** or **<newline>** shall behave as described in the **<newline>** command character during input mode.

39568 For purposes of the display only, the editor shall behave as if a '^' character was entered, and
39569 the cursor shall be positioned as if overwriting the '^' character. When a subsequent character
39570 is entered, the editor shall behave as if that character was entered instead of the original
39571 <control>-V or <control>-Q character.

39572 *Current line:* Unchanged.

39573 *Current column:* Unchanged.

39574 <control>-W

39575 *Synopsis:* <control>-W

39576 If there are characters other than **autoindent** characters that have been input on the current line
39577 before the cursor, the cursor shall move back over the last word preceding the cursor (including
39578 any <blank>s between the end of the last word and the current cursor); the cursor shall not
39579 move to before the first character after the end of any **autoindent** characters.

39580 Otherwise, if there are **autoindent** characters on the current line before the cursor, it is
39581 implementation-defined whether the <control>-W command is an error or if the cursor moves to
39582 the first column position on the line.

39583 Otherwise, if the cursor is in column position 1 and there are previous lines that have been input,
39584 it is implementation-defined whether the <control>-W command is an error or if it is equivalent
39585 to entering <control>-W after the last input character on the previous input line.

39586 Otherwise, it shall be an error.

39587 All of the glyphs on columns between the starting cursor position and (inclusively) the ending
39588 cursor position shall become erase-columns as described in **Input Mode Commands in vi** (on
39589 page 1023).

39590 *Current line:* Unchanged, unless previously input lines are erased, in which case it shall be set to
39591 line -1.

39592 *Current column:* Set to the first column that displays any portion of the last character backed up
39593 over.

39594 <ESC>

39595 *Synopsis:* <ESC>

39596 If input was part of a line-oriented command:

- 39597 1. If *interrupt* was entered, text input mode shall be terminated and the editor shall return to
39598 command mode. The terminal shall be alerted.
- 39599 2. If <ESC> was entered, text input mode shall be terminated and the command shall
39600 continue execution with the input provided.

39601 Otherwise, terminate text input mode and return to command mode.

39602 Any **autoindent** characters entered on newly created lines that have no other non-<newline>s
39603 shall be deleted.

39604 Any leading **autoindent** and <blank>s on newly created lines shall be rewritten to be the
39605 minimum number of <blank>s possible.

39606 The screen shall be redisplayed as necessary to match the contents of the edit buffer.

39607 *Current line:* Unchanged.

- 39608 **Current column:**
- 39609 1. If there are text input characters on the current line, the column shall be set to the last
39610 column where any portion of the last text input character is displayed.
 - 39611 2. Otherwise, if a character is displayed in the current column, unchanged.
 - 39612 3. Otherwise, set to column position 1.
- 39613 **EXIT STATUS**
- 39614 The following exit values shall be returned:
- 39615 0 Successful completion.
- 39616 >0 An error occurred.
- 39617 **CONSEQUENCES OF ERRORS**
- 39618 When any error is encountered and the standard input is not a terminal device file, *vi* shall not
39619 write the file or return to command or text input mode, and shall terminate with a non-zero exit
39620 status.
- 39621 Otherwise, when an unrecoverable error is encountered it shall be equivalent to a SIGHUP
39622 asynchronous event.
- 39623 Otherwise, when an error is encountered, the editor shall behave as specified in **Command**
39624 **Descriptions in vi** (on page 989).
- 39625 **APPLICATION USAGE**
- 39626 None.
- 39627 **EXAMPLES**
- 39628 None.
- 39629 **RATIONALE**
- 39630 See the RATIONALE for *ex* for more information on *vi*. Major portions of the *vi* utility
39631 specification point to *ex* to avoid inadvertent divergence. While *ex* and *vi* have historically been
39632 implemented as a single utility, this is not required by IEEE Std 1003.1-2001.
- 39633 It is recognized that portions of *vi* would be difficult, if not impossible, to implement
39634 satisfactorily on a block-mode terminal, or a terminal without any form of cursor addressing,
39635 thus it is not a mandatory requirement that such features should work on all terminals. It is the
39636 intention, however, that a *vi* implementation should provide the full set of capabilities on all
39637 terminals capable of supporting them.
- 39638 Historically, *vi* exited immediately if the standard input was not a terminal. IEEE Std 1003.1-2001
39639 permits, but does not require, this behavior. An end-of-file condition is not equivalent to an
39640 end-of-file character. A common end-of-file character, <control>-D, is historically a *vi* command.
- 39641 The text in the STDOOUT section reflects the usage of the verb *display* in this section; some
39642 implementations of *vi* use standard output to write to the terminal, but IEEE Std 1003.1-2001
39643 does not require that to be the case.
- 39644 Historically, implementations reverted to open mode if the terminal was incapable of
39645 supporting full visual mode. IEEE Std 1003.1-2001 requires this behavior. Historically, the open
39646 mode of *vi* behaved roughly equivalently to the visual mode, with the exception that only a
39647 single line from the edit buffer (one “buffer line”) was kept current at any time. This line was
39648 normally displayed on the next-to-last line of a terminal with cursor addressing (and the last line
39649 performed its normal visual functions for line-oriented commands and messages). In addition,
39650 some few commands behaved differently in open mode than in visual mode.
39651 IEEE Std 1003.1-2001 requires conformance to historical practice.

39652 Historically, *ex* and *vi* implementations have expected text to proceed in the usual
39653 European/Latin order of left to right, top to bottom. There is no requirement in
39654 IEEE Std 1003.1-2001 that this be the case. The specification was deliberately written using
39655 words like “before”, “after”, “first”, and “last” in order to permit implementations to support
39656 the natural text order of the language.

39657 Historically, lines past the end of the edit buffer were marked with single tilde ('~') characters;
39658 that is, if the one-based display was 20 lines in length, and the last line of the file was on line one,
39659 then lines 2-20 would contain only a single '~' character.

39660 Historically, the *vi* editor attempted to display only complete lines at the bottom of the screen (it
39661 did display partial lines at the top of the screen). If a line was too long to fit in its entirety at the
39662 bottom of the screen, the screen lines where the line would have been displayed were displayed
39663 as single '@' characters, instead of displaying part of the line. IEEE Std 1003.1-2001 permits, but
39664 does not require, this behavior. Implementations are encouraged to attempt always to display a
39665 complete line at the bottom of the screen when doing scrolling or screen positioning by buffer
39666 lines.

39667 Historically, lines marked with '@' were also used to minimize output to dumb terminals over
39668 slow lines; that is, changes local to the cursor were updated, but changes to lines on the screen
39669 that were not close to the cursor were simply marked with an '@' sign instead of being updated
39670 to match the current text. IEEE Std 1003.1-2001 permits, but does not require this feature because
39671 it is used ever less frequently as terminals become smarter and connections are faster.

39672 Initialization in *ex* and *vi*

39673 Historically, *vi* always had a line in the edit buffer, even if the edit buffer was “empty”. For
39674 example:

- 39675 1. The *ex* command = executed from visual mode wrote “1” when the buffer was empty.
- 39676 2. Writes from visual mode of an empty edit buffer wrote files of a single character (a
39677 <newline>), while writes from *ex* mode of an empty edit buffer wrote empty files.
- 39678 3. Put and read commands into an empty edit buffer left an empty line at the top of the edit
39679 buffer.

39680 For consistency, IEEE Std 1003.1-2001 does not permit any of these behaviors.

39681 Historically, *vi* did not always return the terminal to its original modes; for example, ICRNL was
39682 modified if it was not originally set. IEEE Std 1003.1-2001 does not permit this behavior.

39683 Command Descriptions in *vi*

39684 Motion commands are among the most complicated aspects of *vi* to describe. With some
39685 exceptions, the text region and buffer type effect of a motion command on a *vi* command are
39686 described on a case-by-case basis. The descriptions of text regions in IEEE Std 1003.1-2001 are
39687 not intended to imply direction; that is, an inclusive region from line *n* to line *n+5* is identical to
39688 a region from line *n+5* to line *n*. This is of more than academic interest—movements to marks
39689 can be in either direction, and, if the **wrapscan** option is set, so can movements to search points.
39690 Historically, lines are always stored into buffers in text order; that is, from the start of the edit
39691 buffer to the end. IEEE Std 1003.1-2001 requires conformance to historical practice.

39692 Historically, command counts were applied to any associated motion, and were multiplicative
39693 to any supplied motion count. For example, **2cw** is the same as **c2w**, and **2c3w** is the same as
39694 **c6w**. IEEE Std 1003.1-2001 requires this behavior. Historically, *vi* commands that used bigwords,
39695 words, paragraphs, and sentences as objects treated groups of empty lines, or lines that
39696 contained only <blank>s, inconsistently. Some commands treated them as a single entity, while

39697 others treated each line separately. For example, the **w**, **W**, and **B** commands treated groups of
39698 empty lines as individual words; that is, the command would move the cursor to each new
39699 empty line. The **e** and **E** commands treated groups of empty lines as a single word; that is, the
39700 first use would move past the group of lines. The **b** command would just beep at the user, or if
39701 done from the start of the line as a motion command, fail in unexpected ways. If the lines
39702 contained only (or ended with) <blank>s, the **w** and **W** commands would just beep at the user,
39703 the **E** and **e** commands would treat the group as a single word, and the **B** and **b** commands
39704 would treat the lines as individual words. For consistency and simplicity of specification,
39705 IEEE Std 1003.1-2001 requires that all *vi* commands treat groups of empty or blank lines as a
39706 single entity, and that movement through lines ending with <blank>s be consistent with other
39707 movements.

39708 Historically, *vi* documentation indicated that any number of double quotes were skipped after
39709 punctuation marks at sentence boundaries; however, implementations only skipped single
39710 quotes. IEEE Std 1003.1-2001 requires both to be skipped.

39711 Historically, the first and last characters in the edit buffer were word boundaries. This historical
39712 practice is required by IEEE Std 1003.1-2001.

39713 Historically, *vi* attempted to update the minimum number of columns on the screen possible,
39714 which could lead to misleading information being displayed. IEEE Std 1003.1-2001 makes no
39715 requirements other than that the current character being entered is displayed correctly, leaving
39716 all other decisions in this area up to the implementation.

39717 Historically, lines were arbitrarily folded between columns of any characters that required
39718 multiple column positions on the screen, with the exception of tabs, which terminated at the
39719 right-hand margin. IEEE Std 1003.1-2001 permits the former and requires the latter.
39720 Implementations that do not arbitrarily break lines between columns of characters that occupy
39721 multiple column positions should not permit the cursor to rest on a column that does not
39722 contain any part of a character.

39723 The historical *vi* had a problem in that all movements were by buffer lines, not by display or
39724 screen lines. This is often the right thing to do; for example, single line movements, such as **j** or
39725 **k**, should work on buffer lines. Commands like **dj**, or **j.**, where . is a change command, only
39726 make sense for buffer lines. It is not, however, the right thing to do for screen motion or scrolling
39727 commands like <control>-D, <control>-F, and **H**. If the window is fairly small, using buffer lines
39728 in these cases can result in completely random motion; for example, **1<control>-D** can result in a
39729 completely changed screen, without any overlap. This is clearly not what the user wanted. The
39730 problem is even worse in the case of the **H**, **L**, and **M** commands—as they position the cursor at
39731 the first non-<blank> of the line, they may all refer to the same location in large lines, and will
39732 result in no movement at all.

39733 In addition, if the line is larger than the screen, using buffer lines can make it impossible to
39734 display parts of the line—there are not any commands that do not display the beginning of the
39735 line in historical *vi*, and if both the beginning and end of the line cannot be on the screen at the
39736 same time, the user suffers. Finally, the page and half-page scrolling commands historically
39737 moved to the first non-<blank> in the new line. If the line is approximately the same size as the
39738 screen, this is inadequate because the cursor before and after a <control>-D command will refer
39739 to the same location on the screen.

39740 Implementations of *ex* and *vi* exist that do not have these problems because the relevant
39741 commands (<control>-B, <control>-D, <control>-F, <control>-U, <control>-Y, <control>-E, **H**, **L**,
39742 and **M**) operate on display (screen) lines, not (edit) buffer lines.

39743 IEEE Std 1003.1-2001 does not permit this behavior by default because the standard developers
39744 believed that users would find it too confusing. However, historical practice has been relaxed.

39745 For example, **ex** and **vi** historically attempted, albeit sometimes unsuccessfully, to never put part
39746 of a line on the last lines of a screen; for example, if a line would not fit in its entirety, no part of
39747 the line was displayed, and the screen lines corresponding to the line contained single '@'
39748 characters. This behavior is permitted, but not required by IEEE Std 1003.1-2001, so that it is
39749 possible for implementations to support long lines in small screens more reasonably without
39750 changing the commands to be oriented to the display (instead of oriented to the buffer).
39751 IEEE Std 1003.1-2001 also permits implementations to refuse to edit any edit buffer containing a
39752 line that will not fit on the screen in its entirety.

39753 The display area (for example, the value of the **window** edit option) has historically been
39754 “grown”, or expanded, to display new text when local movements are done in displays where
39755 the number of lines displayed is less than the maximum possible. Expansion has historically
39756 been the first choice, when the target line is less than the maximum possible expansion value
39757 away. Scrolling has historically been the next choice, done when the target line is less than half a
39758 display away, and otherwise, the screen was redrawn. There were exceptions, however, in that
39759 **ex** commands generally always caused the screen to be redrawn. IEEE Std 1003.1-2001 does not
39760 specify a standard behavior because there may be external issues, such as connection speed, the
39761 number of characters necessary to redraw as opposed to scroll, or terminal capabilities that
39762 implementations will have to accommodate.

39763 The current line in IEEE Std 1003.1-2001 maps one-to-one to a buffer line in the file. The current
39764 column does not. There are two different column values that are described by
39765 IEEE Std 1003.1-2001. The first is the current column value as set by many of the **vi** commands.
39766 This value is remembered for the lifetime of the editor. The second column value is the actual
39767 position on the screen where the cursor rests. The two are not always the same. For example,
39768 when the cursor is backed by a multi-column character, the actual cursor position on the screen
39769 has historically been the last column of the character in command mode, and the first column of
39770 the character in input mode.

39771 Commands that set the current line, but that do not set the current cursor value (for example, **j**
39772 and **k**) attempt to get as close as possible to the remembered column position, so that the cursor
39773 tends to restrict itself to a vertical column as the user moves around in the edit buffer.
39774 IEEE Std 1003.1-2001 requires conformance to historical practice, requiring that the display
39775 location of the cursor on the display line be adjusted from the current column value as necessary
39776 to support this historical behavior.

39777 Historically, only a single line (and for some terminals, a single line minus 1 column) of
39778 characters could be entered by the user for the line-oriented commands; that is, ;, !, /, or ?.
39779 IEEE Std 1003.1-2001 permits, but does not require, this limitation.

39780 Historically, “soft” errors in **vi** caused the terminal to be alerted, but no error message was
39781 displayed. As a general rule, no error message was displayed for errors in command execution
39782 in **vi**, when the error resulted from the user attempting an invalid or impossible action, or when
39783 a searched-for object was not found. Examples of soft errors included **h** at the left margin,
39784 <control>-B or **[[** at the beginning of the file, **2G** at the end of the file, and so on. In addition,
39785 errors such as **%**, **]]**, **},**, **N**, **n**, **f**, **F**, **t**, and **T** failing to find the searched-for object were soft as well.
39786 Less consistently, **/** and **?** displayed an error message if the pattern was not found, **/**, **?**, **N**, and **n**
39787 displayed an error message if no previous regular expression had been specified, and **;** did not
39788 display an error message if no previous **f**, **F**, **t**, or **T** command had occurred. Also, behavior in
39789 this area might reasonably be based on a runtime evaluation of the speed of a network
39790 connection. Finally, some implementations have provided error messages for soft errors in
39791 order to assist naive users, based on the value of a verbose edit option. IEEE Std 1003.1-2001
39792 does not list specific errors for which an error message shall be displayed. Implementations
39793 should conform to historical practice in the absence of any strong reason to diverge.

39794

Page Backwards

39795

The <control>-B and <control>-F commands historically considered it an error to attempt to page past the beginning or end of the file, whereas the <control>-D and <control>-U commands simply moved to the beginning or end of the file. For consistency, IEEE Std 1003.1-2001 requires the latter behavior for all four commands. All four commands still consider it an error if the current line is at the beginning (<control>-B, <control>-U) or end (<control>-F, <control>-D) of the file. Historically, the <control>-B and <control>-F commands skip two lines in order to include overlapping lines when a single command is entered. This makes less sense in the presence of a *count*, as there will be, by definition, no overlapping lines. The actual calculation used by historical implementations of the vi editor for <control>-B was:

39796

```
((current first line) - count x (window edit option)) +2
```

39797

and for <control>-F was:

39798

```
((current first line) + count x (window edit option)) -2
```

39799

This calculation does not work well when intermixing commands with and without counts; for example, 3<control>-F is not equivalent to entering the <control>-F command three times, and is not reversible by entering the <control>-B command three times. For consistency with other vi commands that take counts, IEEE Std 1003.1-2001 requires a different calculation.

39800

Scroll Forward

39801

The 4BSD and System V implementations of vi differed on the initial value used by the scroll command. 4BSD used:

39802

```
((window edit option) +1) /2
```

39803

while System V used the value of the scroll edit option. The System V version is specified by IEEE Std 1003.1-2001 because the standard developers believed that it was more intuitive and permitted the user a method of setting the scroll value initially without also setting the number of lines that are displayed.

39804

Scroll Forward by Line

39805

Historically, the <control>-E and <control>-Y commands considered it an error if the last and first lines, respectively, were already on the screen. IEEE Std 1003.1-2001 requires conformance to historical practice. Historically, the <control>-E and <control>-Y commands had no effect in open mode. For simplicity and consistency of specification, IEEE Std 1003.1-2001 requires that they behave as usual, albeit with a single line screen.

39806

Clear and Redisplay

39807

The historical <control>-L command refreshed the screen exactly as it was supposed to be currently displayed, replacing any '@' characters for lines that had been deleted but not updated on the screen with refreshed '@' characters. The intent of the <control>-L command is to refresh when the screen has been accidentally overwritten; for example, by a write command from another user, or modem noise.

39831

Redraw Screen

39832

The historical <control>-R command redisplayed only when necessary to update lines that had been deleted but not updated on the screen and that were flagged with '@' characters. There is no requirement that the screen be in any way refreshed if no lines of this form are currently displayed. IEEE Std 1003.1-2001 permits implementations to extend this command to refresh lines on the screen flagged with '@' characters because they are too long to be displayed in the current framework; however, the current line and column need not be modified.

39833

39834

39835

39836

39837

39838

Search for tagstring

39839

39840

39841

39842

39843

Historically, the first non-<blank> at or after the cursor was the first character, and all subsequent characters that were word characters, up to the end of the line, were included. For example, with the cursor on the leading space or on the '#' character in the text "#bar@", the tag was "#bar". On the character 'b' it was "bar", and on the 'a' it was "ar". IEEE Std 1003.1-2001 requires this behavior.

39844

Replace Text with Results from Shell Command

39845

39846

39847

39848

39849

Historically, the <, >, and ! commands considered most cursor motions other than line-oriented motions an error; for example, the command >/foo<CR> succeeded, while the command >l failed, even though the text region described by the two commands might be identical. For consistency, all three commands only consider entire lines and not partial lines, and the region is defined as any line that contains a character that was specified by the motion.

39850

Move to Matching Character

39851

39852

39853

Other matching characters have been left implementation-defined in order to allow extensions such as matching '<' and '>' for searching HTML, or #ifdef, #else, and #endif for searching C source.

39854

Repeat Substitution

39855

39856

IEEE Std 1003.1-2001 requires that any c and g flags specified to the previous substitute command be ignored; however, the r flag may still apply, if supported by the implementation.

39857

Return to Previous (Context or Section)

39858

39859

39860

39861

39862

39863

39864

39865

39866

39867

39868

The [[,]], (,), {, and } commands are all affected by “section boundaries”, but in some historical implementations not all of the commands recognize the same section boundaries. This is a bug, not a feature, and a unique section-boundary algorithm was not described for each command. One special case that is preserved is that the sentence command moves to the end of the last line of the edit buffer while the other commands go to the beginning, in order to preserve the traditional character cut semantics of the sentence command. Historically, vi section boundaries at the beginning and end of the edit buffer were the first non-<blank> on the first and last lines of the edit buffer if one exists; otherwise, the last character of the first and last lines of the edit buffer if one exists. To increase consistency with other section locations, this has been simplified by IEEE Std 1003.1-2001 to the first character of the first and last lines of the edit buffer, or the first and the last lines of the edit buffer if they are empty.

39869

39870

39871

39872

39873

39874

Sentence boundaries were problematic in the historical vi. They were not only the boundaries as defined for the section and paragraph commands, but they were the first non-<blank> that occurred after those boundaries, as well. Historically, the vi section commands were documented as taking an optional window size as a count preceding the command. This was not implemented in historical versions, so IEEE Std 1003.1-2001 requires that the count repeat the command, for consistency with other vi commands.

39875 **Repeat**

39876 Historically, mapped commands other than text input commands could not be repeated using
39877 the **period** command. IEEE Std 1003.1-2001 requires conformance to historical practice.

39878 The restrictions on the interpretation of special characters (for example, <control>-H) in the
39879 repetition of text input mode commands is intended to match historical practice. For example,
39880 given the input sequence:

39881 *iab<control>-H<control>-H<control>-Hdef<escape>*

39882 the user should be informed of an error when the sequence is first entered, but not during a
39883 command repetition. The character <control>-T is specifically exempted from this restriction.
39884 Historical implementations of *vi* ignored <control>-T characters that were input in the original
39885 command during command repetition. IEEE Std 1003.1-2001 prohibits this behavior.

39886 **Find Regular Expression**

39887 Historically, commands did not affect the line searched to or from if the motion command was a
39888 search (/, ?, N, n) and the final position was the start/end of the line. There were some special
39889 cases and *vi* was not consistent. IEEE Std 1003.1-2001 does not permit this behavior, for
39890 consistency. Historical implementations permitted but were unable to handle searches as
39891 motion commands that wrapped (that is, due to the edit option **wrapscan**) to the original
39892 location. IEEE Std 1003.1-2001 requires that this behavior be treated as an error.

39893 Historically, the syntax "/RE/0" was used to force the command to cut text in line mode.
39894 IEEE Std 1003.1-2001 requires conformance to historical practice.

39895 Historically, in open mode, a **z** specified to a search command redisplayed the current line
39896 instead of displaying the current screen with the current line highlighted. For consistency and
39897 simplicity of specification, IEEE Std 1003.1-2001 does not permit this behavior.

39898 Historically, trailing **z** commands were permitted and ignored if entered as part of a search used
39899 as a motion command. For consistency and simplicity of specification, IEEE Std 1003.1-2001 does
39900 not permit this behavior.

39901 **Execute an ex Command**

39902 Historically, *vi* implementations restricted the commands that could be entered on the colon
39903 command line (for example, **append** and **change**), and some other commands were known to
39904 cause them to fail catastrophically. For consistency, IEEE Std 1003.1-2001 does not permit these
39905 restrictions. When executing an *ex* command by entering :, it is not possible to enter a <newline>
39906 as part of the command because it is considered the end of the command. A different approach
39907 is to enter *ex* command mode by using the *vi Q* command (and later resuming visual mode with
39908 the *ex vi* command). In *ex* command mode, the single-line limitation does not exist. So, for
39909 example, the following is valid:

39910 Q
39911 s/break here/break\
39912 here/
39913 vi

39914 IEEE Std 1003.1-2001 requires that, if the *ex* command overwrites any part of the screen that
39915 would be erased by a refresh, *vi* pauses for a character from the user. Historically, this character
39916 could be any character; for example, a character input by the user before the message appeared,
39917 or even a mapped character. This is probably a bug, but implementations that have tried to be
39918 more rigorous by requiring that the user enter a specific character, or that the user enter a
39919 character after the message was displayed, have been forced by user indignation back into

39920 historical behavior. IEEE Std 1003.1-2001 requires conformance to historical practice.

39921 **Shift Left (Right)**

39922 Refer to the Rationale for the ! and / commands. Historically, the < and > commands sometimes
39923 moved the cursor to the first non-<blank> (for example if the command was repeated or with _
39924 as the motion command), and sometimes left it unchanged. IEEE Std 1003.1-2001 does not
39925 permit this inconsistency, requiring instead that the cursor always move to the first non-
39926 <blank>. Historically, the < and > commands did not support buffer arguments, although some
39927 implementations allow the specification of an optional buffer. This behavior is neither required
39928 nor disallowed by IEEE Std 1003.1-2001.

39929 **Execute**

39930 Historically, buffers could execute other buffers, and loops, infinite and otherwise, were
39931 possible. IEEE Std 1003.1-2001 requires conformance to historical practice. The *buffer syntax of
39932 ex is not required in vi, because it is not historical practice and has been used in some vi
39933 implementations to support additional scripting languages.

39934 **Reverse Case**

39935 Historically, the ~ command ignored any associated count, and acted only on the characters in
39936 the current line. For consistency with other vi commands, IEEE Std 1003.1-2001 requires that an
39937 associated count act on the next count characters, and that the command move to subsequent
39938 lines if warranted by count, to make it possible to modify large pieces of text in a reasonably
39939 efficient manner. There exist vi implementations that optionally require an associated motion
39940 command for the ~ command. Implementations supporting this functionality are encouraged to
39941 base it on the **tildedop** edit option and handle the text regions and cursor positioning identically
39942 to the **yank** command.

39943 **Append**

39944 Historically, counts specified to the A, a, I, and i commands repeated the input of the first line
39945 count times, and did not repeat the subsequent lines of the input text. IEEE Std 1003.1-2001
39946 requires that the entire text input be repeated count times.

39947 **Move Backward to Preceding Word**

39948 Historically, vi became confused if word commands were used as motion commands in empty
39949 files. IEEE Std 1003.1-2001 requires that this be an error. Historical implementations of vi had a
39950 large number of bugs in the word movement commands, and they varied greatly in behavior in
39951 the presence of empty lines, “words” made up of a single character, and lines containing only
39952 <blank>s. For consistency and simplicity of specification, IEEE Std 1003.1-2001 does not permit
39953 this behavior.

39954 **Change to End-of-Line**

39955 Some historical implementations of the C command did not behave as described by
39956 IEEE Std 1003.1-2001 when the \$ key was remapped because they were implemented by pushing
39957 the \$ key onto the input queue and reprocessing it. IEEE Std 1003.1-2001 does not permit this
39958 behavior. Historically, the C, S, and s commands did not copy replaced text into the numeric
39959 buffers. For consistency and simplicity of specification, IEEE Std 1003.1-2001 requires that they
39960 behave like their respective c commands in all respects.

39961	Delete
39962	Historically, lines in open mode that were deleted were scrolled up, and an @ glyph written over the beginning of the line. In the case of terminals that are incapable of the necessary cursor motions, the editor erased the deleted line from the screen. IEEE Std 1003.1-2001 requires conformance to historical practice; that is, if the terminal cannot display the '@' character, the line cannot remain on the screen.
39967	Delete to End-of-Line
39968	Some historical implementations of the D command did not behave as described by IEEE Std 1003.1-2001 when the \$ key was remapped because they were implemented by pushing the \$ key onto the input queue and reprocessing it. IEEE Std 1003.1-2001 does not permit this behavior.
39972	Join
39973	An historical oddity of <i>vi</i> is that the commands J , 1J , and 2J are all equivalent. IEEE Std 1003.1-2001 requires conformance to historical practice. The <i>vi J</i> command is specified in terms of the <i>ex join</i> command with an <i>ex command count</i> value. The address correction for a <i>count</i> that is past the end of the edit buffer is necessary for historical compatibility for both <i>ex</i> and <i>vi</i> .
39978	Mark Position
39979	Historical practice is that only lowercase letters, plus 'v' and '^', could be used to mark a cursor position. IEEE Std 1003.1-2001 requires conformance to historical practice, but encourages implementations to support other characters as marks as well.
39982	Repeat Regular Expression Find (Forward and Reverse)
39983	Historically, the N and n commands could not be used as motion components for the c command. With the exception of the cN command, which worked if the search crossed a line boundary, the text region would be discarded, and the user would not be in text input mode. For consistency and simplicity of specification, IEEE Std 1003.1-2001 does not permit this behavior.
39987	Insert Empty Line (Below and Above)
39988	Historically, counts to the O and o commands were used as the number of physical lines to open, if the terminal was dumb and the slowopen option was not set. This was intended to minimize traffic over slow connections and repainting for dumb terminals. IEEE Std 1003.1-2001 does not permit this behavior, requiring that a <i>count</i> to the open command behave as for other text input commands. This change to historical practice was made for consistency, and because a superset of the functionality is provided by the slowopen edit option.
39994	Put from Buffer (Following and Before)
39995	Historically, <i>counts</i> to the p and P commands were ignored if the buffer was a line mode buffer, but were (mostly) implemented as described in IEEE Std 1003.1-2001 if the buffer was a character mode buffer. Because implementations exist that do not have this limitation, and because pasting lines multiple times is generally useful, IEEE Std 1003.1-2001 requires that <i>count</i> be supported for all p and P commands.
40000	Historical implementations of <i>vi</i> were widely known to have major problems in the p and P commands, particularly when unusual regions of text were copied into the edit buffer. The standard developers viewed these as bugs, and they are not permitted for consistency and

40003 simplicity of specification.
40004 Historically, a **P** or **p** command (or an **ex put** command executed from open or visual mode)
40005 executed in an empty file, left an empty line as the first line of the file. For consistency and
40006 simplicity of specification, IEEE Std 1003.1-2001 does not permit this behavior.

40007 Replace Character

40008 Historically, the **r** command did not correctly handle the *erase* and *word erase* characters as
40009 arguments, nor did it handle an associated *count* greater than 1 with a <carriage-return>
40010 argument, for which it replaced *count* characters with a single <newline>. IEEE Std 1003.1-2001
40011 does not permit these inconsistencies.

40012 Historically, the **r** command permitted the <control>-V escaping of entered characters, such as
40013 <ESC> and the <carriage-return>; however, it required two leading <control>-V characters
40014 instead of one. IEEE Std 1003.1-2001 requires that this be changed for consistency with the other
40015 text input commands of *vi*.

40016 Historically, it is an error to enter the **r** command if there are less than *count* characters at or after
40017 the cursor in the line. While a reasonable and unambiguous extension would be to permit the **r**
40018 command on empty lines, it would require that too large a *count* be adjusted to match the
40019 number of characters at or after the cursor for consistency, which is sufficiently different from
40020 historical practice to be avoided. IEEE Std 1003.1-2001 requires conformance to historical
40021 practice.

40022 Replace Characters

40023 Historically, if there were **autoindent** characters in the line on which the **R** command was run,
40024 and **autoindent** was set, the first <newline> would be properly indented and no characters
40025 would be replaced by the <newline>. Each additional <newline> would replace *n* characters,
40026 where *n* was the number of characters that were needed to indent the rest of the line to the
40027 proper indentation level. This behavior is a bug and is not permitted by IEEE Std 1003.1-2001.

40028 Undo

40029 Historical practice for cursor positioning after undoing commands was mixed. In most cases,
40030 when undoing commands that affected a single line, the cursor was moved to the start of added
40031 or changed text, or immediately after deleted text. However, if the user had moved from the line
40032 being changed, the column was either set to the first non-<blank>, returned to the origin of the
40033 command, or remained unchanged. When undoing commands that affected multiple lines or
40034 entire lines, the cursor was moved to the first character in the first line restored. As an example
40035 of how inconsistent this was, a search, followed by an **o** text input command, followed by an
40036 **undo** would return the cursor to the location where the **o** command was entered, but a **cw**
40037 command followed by an **o** command followed by an **undo** would return the cursor to the first
40038 non-<blank> of the line. IEEE Std 1003.1-2001 requires the most useful of these behaviors, and
40039 discards the least useful, in the interest of consistency and simplicity of specification.

- 40040 **Yank**
- 40041 Historically, the **yank** command did not move to the end of the motion if the motion was in the
40042 forward direction. It moved to the end of the motion if the motion was in the backward
40043 direction, except for the **_** command, or for the **G** and **'** commands when the end of the motion
40044 was on the current line. This was further complicated by the fact that for a number of motion
40045 commands, the **yank** command moved the cursor but did not update the screen; for example, a
40046 subsequent command would move the cursor from the end of the motion, even though the
40047 cursor on the screen had not reflected the cursor movement for the **yank** command.
40048 IEEE Std 1003.1-2001 requires that all **yank** commands associated with backward motions move
40049 the cursor to the end of the motion for consistency, and specifically, to make **'** commands as
40050 motions consistent with search patterns as motions.
- 40051 **Yank Current Line**
- 40052 Some historical implementations of the **Y** command did not behave as described by
40053 IEEE Std 1003.1-2001 when the **'** key was remapped because they were implemented by
40054 pushing the **'** key onto the input queue and reprocessing it. IEEE Std 1003.1-2001 does not
40055 permit this behavior.
- 40056 **Redraw Window**
- 40057 Historically, the **z** command always redrew the screen. This is permitted but not required by
40058 IEEE Std 1003.1-2001, because of the frequent use of the **z** command in macros such as **map n nz**,
40059 for screen positioning, instead of its use to change the screen size. The standard developers
40060 believed that expanding or scrolling the screen offered a better interface for users. The ability to
40061 redraw the screen is preserved if the optional new window size is specified, and in the
40062 **<control>-L** and **<control>-R** commands.
- 40063 The semantics of **z^** are confusing at best. Historical practice is that the screen before the screen
40064 that ended with the specified line is displayed. IEEE Std 1003.1-2001 requires conformance to
40065 historical practice.
- 40066 Historically, the **z** command would not display a partial line at the top or bottom of the screen. If
40067 the partial line would normally have been displayed at the bottom of the screen, the command
40068 worked, but the partial line was replaced with '@' characters. If the partial line would normally
40069 have been displayed at the top of the screen, the command would fail. For consistency and
40070 simplicity of specification, IEEE Std 1003.1-2001 does not permit this behavior.
- 40071 Historically, the **z** command with a line specification of 1 ignored the command. For consistency
40072 and simplicity of specification, IEEE Std 1003.1-2001 does not permit this behavior.
- 40073 Historically, the **z** command did not set the cursor column to the first non-<blank> for the
40074 character if the first screen was to be displayed, and was already displayed. For consistency and
40075 simplicity of specification, IEEE Std 1003.1-2001 does not permit this behavior.
- 40076 **Input Mode Commands in vi**
- 40077 Historical implementations of **vi** did not permit the user to erase more than a single line of input,
40078 or to use normal erase characters such as *line erase*, *worderase*, and *erase* to erase **autoindent**
40079 characters. As there exist implementations of **vi** that do not have these limitations, both
40080 behaviors are permitted, but only historical practice is required. In the case of these extensions,
40081 **vi** is required to pause at the **autoindent** and previous line boundaries.
- 40082 Historical implementations of **vi** updated only the portion of the screen where the current cursor
40083 character was displayed. For example, consider the **vi** input keystrokes:

40084 iabcd<escape>0C<tab>

40085 Historically, the <tab> would overwrite the characters "abcd" when it was displayed. Other
40086 implementations replace only the 'a' character with the <tab>, and then push the rest of the
40087 characters ahead of the cursor. Both implementations have problems. The historical
40088 implementation is probably visually nicer for the above example; however, for the keystrokes:

40089 iabcd<ESC>0R<tab><ESC>

40090 the historical implementation results in the string "bcd" disappearing and then magically
40091 reappearing when the <ESC> character is entered. IEEE Std 1003.1-2001 requires the former
40092 behavior when overwriting erase-columns—that is, overwriting characters that are no longer
40093 logically part of the edit buffer—and the latter behavior otherwise.

40094 Historical implementations of vi discarded the <control>-D and <control>-T characters when
40095 they were entered at places where their command functionality was not appropriate.
40096 IEEE Std 1003.1-2001 requires that the <control>-T functionality always be available, and that
40097 <control>-D be treated as any other key when not operating on **autoindent** characters.

40098 NUL

40099 Some historical implementations of vi limited the number of characters entered using the NUL
40100 input character to 256 bytes. IEEE Std 1003.1-2001 permits this limitation; however,
40101 implementations are encouraged to remove this limit.

40102 <control>-D

40103 See also Rationale for the input mode command <newline>. The hidden assumptions in the
40104 <control>-D command (and in the vi **autoindent** specification in general) is that <space>s take
40105 up a single column on the screen and that <tab>s are comprised of an integral number of
40106 <space>s.

40107 <newline>

40108 Implementations are permitted to rewrite **autoindent** characters in the line when <newline>,
40109 <carriage-return>, <control>-D, and <control>-T are entered, or when the **shift** commands are
40110 used, because historical implementations have both done so and found it necessary to do so. For
40111 example, a <control>-D when the cursor is preceded by a single <tab>, with **tabstop** set to 8, and
40112 **shiftwidth** set to 3, will result in the <tab> being replaced by several <space>s.

40113 <control>-T

40114 See also the Rationale for the input mode command <newline>. Historically, <control>-T only
40115 worked if no non-<blank>s had yet been input in the current input line. In addition, the
40116 characters inserted by <control>-T were treated as **autoindent** characters, and could not be
40117 erased using normal user erase characters. Because implementations exist that do not have
40118 these limitations, and as moving to a column boundary is generally useful, IEEE Std 1003.1-2001
40119 requires that both limitations be removed.

40120	<control>-V
40121	Historically, vi used ^V, regardless of the value of the literal-next character of the terminal.
40122	IEEE Std 1003.1-2001 requires conformance to historical practice.
40123	The uses described for <control>-V can also be accomplished with <control>-Q, which is useful
40124	on terminals that use <control>-V for the down-arrow function. However, most historical
40125	implementations use <control>-Q for the <i>termios</i> START character, so the editor will generally
40126	not receive the <control>-Q unless stty ixon mode is set to off. (In addition, some historical
40127	implementations of vi explicitly set ixon mode to on, so it was difficult for the user to set it to
40128	off.) Any of the command characters described in IEEE Std 1003.1-2001 can be made ineffective
40129	by their selection as <i>termios</i> control characters, using the stty utility or other methods described
40130	in the System Interfaces volume of IEEE Std 1003.1-2001.
40131	<ESC>
40132	Historically, SIGINT alerted the terminal when used to end input mode. This behavior is
40133	permitted, but not required, by IEEE Std 1003.1-2001.
40134	FUTURE DIRECTIONS
40135	None.
40136	SEE ALSO
40137	<i>ed, ex, stty</i>
40138	CHANGE HISTORY
40139	First released in Issue 2.
40140	Issue 5
40141	The FUTURE DIRECTIONS section is added.
40142	Issue 6
40143	This utility is marked as part of the User Portability Utilities option.
40144	The APPLICATION USAGE section is added.
40145	The obsolescent SYNOPSIS is removed.
40146	The following new requirements on POSIX implementations derive from alignment with the
40147	Single UNIX Specification:
40148	• The reindent command description is added.
40149	The vi utility has been extensively rewritten for alignment with the IEEE P1003.2b draft
40150	standard.
40151	IEEE PASC Interpretations 1003.2 #57, #62, #63, #64, #78, and #188 are applied.
40152	IEEE PASC Interpretation 1003.2 #207 is applied, clarifying the description of the R command in
40153	a manner similar to the descriptions of other text input mode commands such as i , o , and O .
40154	The -l option is removed.
40155	IEEE Std 1003.1-2001/Cor 1-2002, item XCU/TC1/D6/41 is applied, adding [count] to the 1
40156	Synopsis for l . 1
40157	IEEE Std 1003.1-2001/Cor 1-2002, item XCU/TC1/D6/42 is applied, adding [count] to the 1
40158	Synopsis for l . 1

40159 NAME

40160 wait — await process completion

40161 SYNOPSIS

40162 wait [pid...]

40163 DESCRIPTION

40164 When an asynchronous list (see Section 2.9.3.1 (on page 50)) is started by the shell, the process ID
40165 of the last command in each element of the asynchronous list shall become known in the current
40166 shell execution environment; see Section 2.12 (on page 61).

40167 If the *wait* utility is invoked with no operands, it shall wait until all process IDs known to the
40168 invoking shell have terminated and exit with a zero exit status.

40169 If one or more *pid* operands are specified that represent known process IDs, the *wait* utility shall
40170 wait until all of them have terminated. If one or more *pid* operands are specified that represent
40171 unknown process IDs, *wait* shall treat them as if they were known process IDs that exited with
40172 exit status 127. The exit status returned by the *wait* utility shall be the exit status of the process
40173 requested by the last *pid* operand.

40174 The known process IDs are applicable only for invocations of *wait* in the current shell execution
40175 environment.

40176 OPTIONS

40177 None.

40178 OPERANDS

40179 The following operand shall be supported:

40180 *pid* One of the following:

40181 1. The unsigned decimal integer process ID of a command, for which the utility
40182 is to wait for the termination.

40183 2. A job control job ID (see the Base Definitions volume of IEEE Std 1003.1-2001,
40184 Section 3.203, Job Control Job ID) that identifies a background process group
40185 to be waited for. The job control job ID notation is applicable only for
40186 invocations of *wait* in the current shell execution environment; see Section
40187 2.12 (on page 61). The exit status of *wait* shall be determined by the last
40188 command in the pipeline.

40189 **Note:** The job control job ID type of *pid* is only available on systems supporting
40190 the User Portability Utilities option.

40191 STDIN

40192 Not used.

40193 INPUT FILES

40194 None.

40195 ENVIRONMENT VARIABLES

40196 The following environment variables shall affect the execution of *wait*:

40197 *LANG* Provide a default value for the internationalization variables that are unset or null.
40198 (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2,
40199 Internationalization Variables for the precedence of internationalization variables
40200 used to determine the values of locale categories.)

40201 *LC_ALL* If set to a non-empty string value, override the values of all the other
40202 internationalization variables.

40203	<i>LC_CTYPE</i>	Determine the locale for the interpretation of sequences of bytes of text data as characters (for example, single-byte as opposed to multi-byte characters in arguments).
40206	<i>LC_MESSAGES</i>	Determine the locale that should be used to affect the format and contents of diagnostic messages written to standard error.
40209 XSI	<i>NLSPATH</i>	Determine the location of message catalogs for the processing of <i>LC_MESSAGES</i> .
40210	ASYNCHRONOUS EVENTS	
40211		Default.
40212	STDOUT	
40213		Not used.
40214	STDERR	
40215		The standard error shall be used only for diagnostic messages.
40216	OUTPUT FILES	
40217		None.
40218	EXTENDED DESCRIPTION	
40219		None.
40220	EXIT STATUS	
40221		If one or more operands were specified, all of them have terminated or were not known by the invoking shell, and the status of the last operand specified is known, then the exit status of <i>wait</i> shall be the exit status information of the command indicated by the last operand specified. If the process terminated abnormally due to the receipt of a signal, the exit status shall be greater than 128 and shall be distinct from the exit status generated by other signals, but the exact value is unspecified. (See the <i>kill -l</i> option.) Otherwise, the <i>wait</i> utility shall exit with one of the following values:
40228	0	The <i>wait</i> utility was invoked with no operands and all process IDs known by the invoking shell have terminated.
40230	1-126	The <i>wait</i> utility detected an error.
40231	127	The command identified by the last <i>pid</i> operand specified is unknown.
40232	CONSEQUENCES OF ERRORS	
40233		Default.
40234	APPLICATION USAGE	
40235		On most implementations, <i>wait</i> is a shell built-in. If it is called in a subshell or separate utility execution environment, such as one of the following:
40237	(<i>wait</i>)	
40238	nohup <i>wait</i> ...	
40239	find . -exec <i>wait</i> ... \\;	
40240	it returns immediately because there are no known process IDs to wait for in those environments.	
40242		Historical implementations of interactive shells have discarded the exit status of terminated background processes before each shell prompt. Therefore, the status of background processes was usually lost unless it terminated while <i>wait</i> was waiting for it. This could be a serious problem when a job that was expected to run for a long time actually terminated quickly with a syntax or initialization error because the exit status returned was usually zero if the requested

40247 process ID was not found. This volume of IEEE Std 1003.1-2001 requires the implementation to
40248 keep the status of terminated jobs available until the status is requested, so that scripts like:

```
40249 j1&
40250 p1=$!
40251 j2&
40252 wait $p1
40253 echo Job 1 exited with status $?
40254 wait $!
40255 echo Job 2 exited with status $?
```

40256 work without losing status on any of the jobs. The shell is allowed to discard the status of any
40257 process if it determines that the application cannot get the process ID for that process from the
40258 shell. It is also required to remember only {CHILD_MAX} number of processes in this way. Since
40259 the only way to get the process ID from the shell is by using the '\$!' shell parameter, the shell is
40260 allowed to discard the status of an asynchronous list if "\$!" was not referenced before another
40261 asynchronous list was started. (This means that the shell only has to keep the status of the last
40262 asynchronous list started if the application did not reference "\$!". If the implementation of the
40263 shell is smart enough to determine that a reference to "\$!" was not saved anywhere that the
40264 application can retrieve it later, it can use this information to trim the list of saved information.
40265 Note also that a successful call to *wait* with no operands discards the exit status of all
40266 asynchronous lists.)

40267 If the exit status of *wait* is greater than 128, there is no way for the application to know if the
40268 waited-for process exited with that value or was killed by a signal. Since most utilities exit with
40269 small values, there is seldom any ambiguity. Even in the ambiguous cases, most applications
40270 just need to know that the asynchronous job failed; it does not matter whether it detected an
40271 error and failed or was killed and did not complete its job normally.

40272 EXAMPLES

40273 Although the exact value used when a process is terminated by a signal is unspecified, if it is
40274 known that a signal terminated a process, a script can still reliably determine which signal by
40275 using *kill* as shown by the following script:

```
40276 sleep 1000&
40277 pid=$!
40278 kill -kill $pid
40279 wait $pid
40280 echo $pid was terminated by a SIG$(kill -l $?) signal.
```

40281 If the following sequence of commands is run in less than 31 seconds:

```
40282 sleep 257 | sleep 31 &
40283 jobs -l %%
```

40284 either of the following commands returns the exit status of the second *sleep* in the pipeline:

```
40285 wait <pid of sleep 31>
40286 wait %%
```

40287 RATIONALE

40288 The description of *wait* does not refer to the *waitpid()* function from the System Interfaces
40289 volume of IEEE Std 1003.1-2001 because that would needlessly overspecify this interface.
40290 However, the wording means that *wait* is required to wait for an explicit process when it is given
40291 an argument so that the status information of other processes is not consumed. Historical
40292 implementations use the *wait()* function defined in the System Interfaces volume of
40293 IEEE Std 1003.1-2001 until *wait()* returns the requested process ID or finds that the requested

40294 process does not exist. Because this means that a shell script could not reliably get the status of
40295 all background children if a second background job was ever started before the first job finished,
40296 it is recommended that the *wait* utility use a method such as the functionality provided by the
40297 *waitpid()* function.

40298 The ability to wait for multiple *pid* operands was adopted from the KornShell.

40299 This new functionality was added because it is needed to determine the exit status of any
40300 asynchronous list accurately. The only compatibility problem that this change creates is for a
40301 script like

40302 while sleep 60 do
40303 job& echo Job started \$(date) as \$! done

40304 which causes the shell to monitor all of the jobs started until the script terminates or runs out of
40305 memory. This would not be a problem if the loop did not reference "\$!" or if the script would
40306 occasionally *wait* for jobs it started.

40307 **FUTURE DIRECTIONS**

40308 None.

40309 **SEE ALSO**

40310 Chapter 2 (on page 29), *kill*, *sh*, the System Interfaces volume of IEEE Std 1003.1-2001, *wait()*,
40311 *waitpid()*

40312 **CHANGE HISTORY**

40313 First released in Issue 2.

40314 NAME

40315 wc — word, line, and byte or character count

40316 SYNOPSIS

40317 `wc [-c|-m][-lw][file...]`

40318 DESCRIPTION

40319 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.

40321 The utility also shall write a total count for all named files, if more than one input file is specified.

40323 The `wc` utility shall consider a *word* to be a non-zero-length string of characters delimited by white space.

40325 OPTIONS

40326 The `wc` utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section 12.2, Utility Syntax Guidelines.

40328 The following options shall be supported:

40329 `-c` Write to the standard output the number of bytes in each input file.

40330 `-l` Write to the standard output the number of <newline>s in each input file.

40331 `-m` Write to the standard output the number of characters in each input file.

40332 `-w` Write to the standard output the number of words in each input file.

40333 When any option is specified, `wc` shall report only the information requested by the specified options.

40335 OPERANDS

40336 The following operand shall be supported:

40337 `file` A pathname of an input file. If no `file` operands are specified, the standard input shall be used.

40339 STDIN

40340 The standard input shall be used only if no `file` operands are specified. See the INPUT FILES section.

40342 INPUT FILES

40343 The input files may be of any type.

40344 ENVIRONMENT VARIABLES

40345 The following environment variables shall affect the execution of `wc`:

40346 `LANG` Provide a default value for the internationalization variables that are unset or null. (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2, Internationalization Variables for the precedence of internationalization variables used to determine the values of locale categories.)

40350 `LC_ALL` If set to a non-empty string value, override the values of all the other internationalization variables.

40352 `LC_CTYPE` Determine the locale for the interpretation of sequences of bytes of text data as characters (for example, single-byte as opposed to multi-byte characters in arguments and input files) and which characters are defined as white space characters.

- 40356 ***LC_MESSAGES***
40357 Determine the locale that should be used to affect the format and contents of
40358 diagnostic messages written to standard error and informative messages written to
40359 standard output.
- 40360 XSI ***NLSPATH*** Determine the location of message catalogs for the processing of *LC_MESSAGES*.
- 40361 **ASYNCHRONOUS EVENTS**
- 40362 Default.
- 40363 **STDOUT**
- 40364 By default, the standard output shall contain an entry for each input file of the form:
40365 "%d %d %d %s\n", <newlines>, <words>, <bytes>, <file>
40366 If the **-m** option is specified, the number of characters shall replace the <bytes> field in this
40367 format.
40368 If any options are specified and the **-l** option is not specified, the number of <newline>s shall
40369 not be written.
40370 If any options are specified and the **-w** option is not specified, the number of words shall not be
40371 written.
40372 If any options are specified and neither **-c** nor **-m** is specified, the number of bytes or characters
40373 shall not be written.
40374 If no input *file* operands are specified, no name shall be written and no <blank>s preceding the
40375 pathname shall be written.
40376 If more than one input *file* operand is specified, an additional line shall be written, of the same
40377 format as the other lines, except that the word **total** (in the POSIX locale) shall be written instead
40378 of a pathname and the total of each column shall be written as appropriate. Such an additional
40379 line, if any, is written at the end of the output.
- 40380 **STDERR**
- 40381 The standard error shall be used only for diagnostic messages.
- 40382 **OUTPUT FILES**
- 40383 None.
- 40384 **EXTENDED DESCRIPTION**
- 40385 None.
- 40386 **EXIT STATUS**
- 40387 The following exit values shall be returned:
- 40388 0 Successful completion.
- 40389 >0 An error occurred.
- 40390 **CONSEQUENCES OF ERRORS**
- 40391 Default.

40392 APPLICATION USAGE

40393 The **-m** option is not a switch, but an option at the same level as **-c**. Thus, to produce the full
40394 default output with character counts instead of bytes, the command required is:

40395 `wc -mlw`

40396 EXAMPLES

40397 None.

40398 RATIONALE

40399 The output file format pseudo-*printf()* string differs from the System V version of *wc*:

40400 `"%7d%7d%7d %s\n"`

40401 which produces possibly ambiguous and unparsable results for very large files, as it assumes no
40402 number shall exceed six digits.

40403 Some historical implementations use only <space>, <tab>, and <newline> as word separators.
40404 The equivalent of the ISO C standard *isspace()* function is more appropriate.

40405 The **-c** option stands for “character” count, even though it counts bytes. This stems from the
40406 sometimes erroneous historical view that bytes and characters are the same size. Due to
40407 international requirements, the **-m** option (reminiscent of “multi-byte”) was added to obtain
40408 actual character counts.

40409 Early proposals only specified the results when input files were text files. The current
40410 specification more closely matches historical practice. (Bytes, words, and <newline>s are
40411 counted separately and the results are written when an end-of-file is detected.)

40412 Historical implementations of the *wc* utility only accepted one argument to specify the options
40413 **-c**, **-l**, and **-w**. Some of them also had multiple occurrences of an option cause the
40414 corresponding count to be written multiple times and had the order of specification of the
40415 options affect the order of the fields on output, but did not document either of these. Because
40416 common usage either specifies no options or only one option, and because none of this was
40417 documented, the changes required by this volume of IEEE Std 1003.1-2001 should not break
40418 many historical applications (and do not break any historical conforming applications).

40419 FUTURE DIRECTIONS

40420 None.

40421 SEE ALSO

40422 *cksum*

40423 CHANGE HISTORY

40424 First released in Issue 2.

40425 NAME

40426 what — identify SCCS files (**DEVELOPMENT**)

40427 SYNOPSIS

40428 XSI `what [-s] file...`

40429

40430 DESCRIPTION

40431 The *what* utility shall search the given files for all occurrences of the pattern that *get* (see *get*)
40432 substitutes for the %Z% keyword ("@(#)") and shall write to standard output what follows
40433 until the first occurrence of one of the following:

40434 " > newline \ NUL

40435 OPTIONS

40436 The *what* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section
40437 12.2, Utility Syntax Guidelines.

40438 The following option shall be supported:

40439 **-s** Quit after finding the first occurrence of the pattern in each file.

40440 OPERANDS

40441 The following operands shall be supported:

40442 *file* A pathname of a file to search.

40443 STDIN

40444 Not used.

40445 INPUT FILES

40446 The input files shall be of any file type.

40447 ENVIRONMENT VARIABLES

40448 The following environment variables shall affect the execution of *what*:

40449 *LANG* Provide a default value for the internationalization variables that are unset or null.
40450 (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2,
40451 Internationalization Variables for the precedence of internationalization variables
40452 used to determine the values of locale categories.)

40453 *LC_ALL* If set to a non-empty string value, override the values of all the other
40454 internationalization variables.

40455 *LC_CTYPE* Determine the locale for the interpretation of sequences of bytes of text data as
40456 characters (for example, single-byte as opposed to multi-byte characters in
40457 arguments and input files).

40458 *LC_MESSAGES*

40459 Determine the locale that should be used to affect the format and contents of
40460 diagnostic messages written to standard error.

40461 *NLSPATH* Determine the location of message catalogs for the processing of *LC_MESSAGES*.

40462 ASYNCHRONOUS EVENTS

40463 Default.

40464 STDOUT

40465 The standard output shall consist of the following for each *file* operand:

40466 "%s:\n\t%s\n", <pathname>, <identification string>

40467 STDERR

40468 The standard error shall be used only for diagnostic messages.

40469 OUTPUT FILES

40470 None.

40471 EXTENDED DESCRIPTION

40472 None.

40473 EXIT STATUS

40474 The following exit values shall be returned:

40475 0 Any matches were found.

40476 1 Otherwise.

40477 CONSEQUENCES OF ERRORS

40478 Default.

40479 APPLICATION USAGE

40480 The *what* utility is intended to be used in conjunction with the SCCS command *get*, which
40481 automatically inserts identifying information, but it can also be used where the information is
40482 inserted by any other means.

40483 When the string "@(#)" is included in a library routine in a shared library, it might not be found
40484 in an **a.out** file using that library routine.

40485 EXAMPLES

40486 If the C-language program in file **f.c** contains:

40487 `char ident[] = "@(#)" identification information";`

40488 and **f.c** is compiled to yield **f.o** and **a.out**, then the command:

40489 `what f.c f.o a.out`

40490 writes:

40491 `f.c:`

40492 `identification information`

40493 `...`

40494 `f.o:`

40495 `identification information`

40496 `...`

40497 `a.out:`

40498 `identification information`

40499 `...`

40500 RATIONALE

40501 None.

40502 FUTURE DIRECTIONS

40503 None.

40504 SEE ALSO

40505 *get*

40506 CHANGE HISTORY

40507 First released in Issue 2.

40508 NAME

40509 who — display who is on the system

40510 SYNOPSIS

40511 UP who [-mTu]

40512 XSI who [-mu]-s[-bHlprt][file]

40513 who [-mTu][-abdHlprt][file]

40514 who -q [file]

40515 who am i

40516 who am I

40517

40518 DESCRIPTION

40519 The *who* utility shall list various pieces of information about accessible users. The domain of accessibility is implementation-defined.

40521 XSI Based on the options given, *who* can also list the user's name, terminal line, login time, elapsed time since activity occurred on the line, and the process ID of the command interpreter for each current system user.

40524 OPTIONS

40525 The *who* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section 12.2, Utility Syntax Guidelines.

40527 The following options shall be supported. The metavariables, such as <line>, refer to fields described in the STDOUT section.

40529 XSI **-a** Process the implementation-defined database or named file with the **-b**, **-d**, **-l**, **-p**, **-r**, **-t**, **-T** and **-u** options turned on.

40531 XSI **-b** Write the time and date of the last reboot.

40532 XSI **-d** Write a list of all processes that have expired and not been respawned by the *init* system process. The <exit> field shall appear for dead processes and contain the termination and exit values of the dead process. This can be useful in determining why a process terminated.

40536 XSI **-H** Write column headings above the regular output.

40537 XSI **-l** (The letter ell.) List only those lines on which the system is waiting for someone to login. The <name> field shall be LOGIN in such cases. Other fields shall be the same as for user entries except that the <state> field does not exist.

40540 **-m** Output only information about the current terminal.

40541 XSI **-p** List any other process that is currently active and has been previously spawned by *init*.

40543 XSI **-q** (Quick.) List only the names and the number of users currently logged on. When this option is used, all other options shall be ignored.

40545 XSI **-r** Write the current run-level of the *init* process.

40546 XSI **-s** List only the <name>, <line>, and <time> fields. This is the default case.

40547 XSI **-t** Indicate the last change to the system clock.

40548	-T	Show the state of each terminal, as described in the STDOUT section.
40549	-u	Write “idle time” for each displayed user in addition to any other information. The idle time is the time since any activity occurred on the user’s terminal. The method of determining this is unspecified. This option shall list only those users who are currently logged in. The <i><name></i> is the user’s login name. The <i><line></i> is the name of the line as found in the directory /dev. The <i><time></i> is the time that the user logged in. The <i><activity></i> is the number of hours and minutes since activity last occurred on that particular line. A dot indicates that the terminal has seen activity in the last minute and is therefore “current”. If more than twenty-four hours have elapsed or the line has not been used since boot time, the entry shall be marked <i><old></i> . This field is useful when trying to determine whether a person is working at the terminal or not. The <i><pid></i> is the process ID of the user’s login process.
40550		
40551 XSI		
40552		
40553		
40554		
40555		
40556		
40557		
40558		
40559		

40560 OPERANDS

40561 XSI	The following operands shall be supported:
40562	am i, am I In the POSIX locale, limit the output to describing the invoking user, equivalent to the -m option. The am and i or I must be separate arguments.
40564	file Specify a pathname of a file to substitute for the implementation-defined database of logged-on users that <i>who</i> uses by default.
40565	

40566 STDIN

40567 Not used.

40568 INPUT FILES

40569 None.

40570 ENVIRONMENT VARIABLES

40571	The following environment variables shall affect the execution of <i>who</i> :
40572	LANG Provide a default value for the internationalization variables that are unset or null. (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2, Internationalization Variables for the precedence of internationalization variables used to determine the values of locale categories.)
40573	
40574	
40575	
40576	LC_ALL If set to a non-empty string value, override the values of all the other internationalization variables.
40577	
40578	LC_CTYPE Determine the locale for the interpretation of sequences of bytes of text data as characters (for example, single-byte as opposed to multi-byte characters in arguments).
40579	
40580	
40581	LC_MESSAGES
40582	Determine the locale that should be used to affect the format and contents of diagnostic messages written to standard error.
40583	
40584	LC_TIME Determine the locale used for the format and contents of the date and time strings.
40585 XSI	NLSPATH Determine the location of message catalogs for the processing of LC_MESSAGES .
40586	TZ Determine the timezone used when writing date and time information. If TZ is unset or null, an unspecified default timezone shall be used.
40587	

40588 ASYNCHRONOUS EVENTS

40589 Default.

40590 STDOUT

40591 The *who* utility shall write its default format to the standard output in an implementation-defined format, subject only to the requirement of containing the information described above.

40593 XSI OF XSI-conformant systems shall write the default information to the standard output in the following general format:

40595 `<name>[<state>]<line><time>[<activity>][<pid>][<comment>][<exit>]`

40596 The following format shall be used for the **-T** option:

40597 `"%s %c %s %s\n" <name>, <terminal state>, <terminal name>,`
40598 `<time of login>`

40599 where *<terminal state>* is one of the following characters:

- 40600 + The terminal allows write access to other users.
- 40601 - The terminal denies write access to other users.
- 40602 ? The terminal write-access state cannot be determined.

40603 In the POSIX locale, the *<time of login>* shall be equivalent in format to the output of:

40604 `date + "%b %e %H:%M"`

40605 If the **-u** option is used with **-T**, the idle time shall be added to the end of the previous format in
40606 an unspecified format.

40607 STDERR

40608 The standard error shall be used only for diagnostic messages.

40609 OUTPUT FILES

40610 None.

40611 EXTENDED DESCRIPTION

40612 None.

40613 EXIT STATUS

40614 The following exit values shall be returned:

- 40615 0 Successful completion.
- 40616 >0 An error occurred.

40617 CONSEQUENCES OF ERRORS

40618 Default.

40619 APPLICATION USAGE

40620 The name *init* used for the system process is the most commonly used on historical systems, but
40621 it may vary.

40622 The “domain of accessibility” referred to is a broad concept that permits interpretation either on
40623 a very secure basis or even to allow a network-wide implementation like the historical *rwho*.

40624 EXAMPLES

40625 None.

40626 RATIONALE

40627 Due to differences between historical implementations, the base options provided were a
40628 compromise to allow users to work with those functions. The standard developers also
40629 considered removing all the options, but felt that these options offered users valuable
40630 functionality. Additional options to match historical systems are available on XSI-conformant

40631 systems.

40632 It is recognized that the *who* command may be of limited usefulness, especially in a multi-level
40633 secure environment. The standard developers considered, however, that having some standard
40634 method of determining the “accessibility” of other users would aid user portability.

40635 No format was specified for the default *who* output for systems not supporting the XSI
40636 Extension. In such a user-oriented command, designed only for human use, this was not
40637 considered to be a deficiency.

40638 The format of the terminal name is unspecified, but the descriptions of *ps*, *talk*, and *write* require
40639 that they use the same format.

40640 It is acceptable for an implementation to produce no output for an invocation of *who mil*.

40641 **FUTURE DIRECTIONS**

40642 None.

40643 **SEE ALSO**

40644 *mesg*

40645 **CHANGE HISTORY**

40646 First released in Issue 2.

40647 **Issue 6**

40648 This utility is marked as part of the User Portability Utilities option.

40649 The *TZ* entry is added to the ENVIRONMENT VARIABLES section.

40650 NAME

40651 write — write to another user

40652 SYNOPSIS

40653 UP write *user_name* [*terminal*]

40654

40655 DESCRIPTION

40656 The *write* utility shall read lines from the user's standard input and write them to the terminal of
40657 another user. When first invoked, it shall write the message:

40658 **Message from** *sender-login-id* (*sending-terminal*) [*date*]...

40659 to *user_name*. When it has successfully completed the connection, the sender's terminal shall be
40660 alerted twice to indicate that what the sender is typing is being written to the recipient's
40661 terminal.

40662 If the recipient wants to reply, this can be accomplished by typing:

40663 write *sender-login-id* [*sending-terminal*]

40664 upon receipt of the initial message. Whenever a line of input as delimited by an NL, EOF, or EOL
40665 special character (see the Base Definitions volume of IEEE Std 1003.1-2001, Chapter 11, General
40666 Terminal Interface) is accumulated while in canonical input mode, the accumulated data shall be
40667 written on the other user's terminal. Characters shall be processed as follows:

- 40668 • Typing <alert> shall write the alert character to the recipient's terminal.
- 40669 • Typing the erase and kill characters shall affect the sender's terminal in the manner described
40670 by the **termios** interface in the Base Definitions volume of IEEE Std 1003.1-2001, Chapter 11,
40671 General Terminal Interface.
- 40672 • Typing the interrupt or end-of-file characters shall cause *write* to write an appropriate
40673 message ("EOT\n" in the POSIX locale) to the recipient's terminal and exit.
- 40674 • Typing characters from *LC_CTYPE* classifications **print** or **space** shall cause those characters
40675 to be sent to the recipient's terminal.
- 40676 • When and only when the *stty iexten* local mode is enabled, the existence and processing of
40677 additional special control characters and multi-byte or single-byte functions is
40678 implementation-defined.
- 40679 • Typing other non-printable characters shall cause implementation-defined sequences of
40680 printable characters to be written to the recipient's terminal.

40681 To write to a user who is logged in more than once, the *terminal* argument can be used to indicate
40682 which terminal to write to; otherwise, the recipient's terminal is selected in an implementation-
40683 defined manner and an informational message is written to the sender's standard output,
40684 indicating which terminal was chosen.

40685 Permission to be a recipient of a *write* message can be denied or granted by use of the *mesg*
40686 utility. However, a user's privilege may further constrain the domain of accessibility of other
40687 users' terminals. The *write* utility shall fail when the user lacks the appropriate privileges to
40688 perform the requested action.

40689 OPTIONS

40690 None.

40691 OPERANDS

40692 The following operands shall be supported:

40693 *user_name* Login name of the person to whom the message shall be written. The application
40694 shall ensure that this operand is of the form returned by the *who* utility.

40695 *terminal* Terminal identification in the same format provided by the *who* utility.

40696 STDIN

40697 Lines to be copied to the recipient's terminal are read from standard input.

40698 INPUT FILES

40699 None.

40700 ENVIRONMENT VARIABLES

40701 The following environment variables shall affect the execution of *write*:

40702 *LANG* Provide a default value for the internationalization variables that are unset or null.
40703 (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2,
40704 Internationalization Variables for the precedence of internationalization variables
40705 used to determine the values of locale categories.)

40706 *LC_ALL* If set to a non-empty string value, override the values of all the other
40707 internationalization variables.

40708 *LC_CTYPE* Determine the locale for the interpretation of sequences of bytes of text data as
40709 characters (for example, single-byte as opposed to multi-byte characters in
40710 arguments and input files). If the recipient's locale does not use an *LC_CTYPE*
40711 equivalent to the sender's, the results are undefined.

40712 *LC_MESSAGES*

40713 Determine the locale that should be used to affect the format and contents of
40714 diagnostic messages written to standard error and informative messages written to
40715 standard output.

40716 XSI *NLSPATH* Determine the location of message catalogs for the processing of *LC_MESSAGES*.

40717 ASYNCHRONOUS EVENTS

40718 If an interrupt signal is received, *write* shall write an appropriate message on the recipient's
40719 terminal and exit with a status of zero. It shall take the standard action for all other signals.

40720 STDOUT

40721 An informational message shall be written to standard output if a recipient is logged in more
40722 than once.

40723 STDERR

40724 The standard error shall be used only for diagnostic messages.

40725 OUTPUT FILES

40726 The recipient's terminal is used for output.

40727 EXTENDED DESCRIPTION

40728 None.

40729 EXIT STATUS

40730 The following exit values shall be returned:

40731 0 Successful completion.

40732 >0 The addressed user is not logged on or the addressed user denies permission.

40733 CONSEQUENCES OF ERRORS

40734 Default.

40735 APPLICATION USAGE

40736 The *talk* utility is considered by some users to be a more usable utility on full-screen terminals.

40737 EXAMPLES

40738 None.

40739 RATIONALE

40740 The *write* utility was included in this volume of IEEE Std 1003.1-2001 since it can be
40741 implemented on all terminal types. The standard developers considered the *talk* utility, which
40742 cannot be implemented on certain terminals, to be a “better” communications interface. Both of
40743 these programs are in widespread use on historical implementations. Therefore, the standard
40744 developers decided that both utilities should be specified.

40745 The format of the terminal name is unspecified, but the descriptions of *ps*, *talk*, *who*, and *write*
40746 require that they all use or accept the same format.

40747 FUTURE DIRECTIONS

40748 None.

40749 SEE ALSO

40750 *mesg*, *talk*, *who*, the Base Definitions volume of IEEE Std 1003.1-2001, Chapter 11, General
40751 Terminal Interface

40752 CHANGE HISTORY

40753 First released in Issue 2.

40754 Issue 5

40755 The FUTURE DIRECTIONS section is added.

40756 Issue 6

40757 This utility is marked as part of the User Portability Utilities option.

40758 The normative text is reworded to avoid use of the term “must” for application requirements.

40759 NAME

40760 xargs — construct argument lists and invoke utility

40761 SYNOPSIS

40762 XSI xargs [-t][-p][-E eofstr][-I replstr][-L number][-n number [-x]]
40763 [-s size][utility [argument...]]

40764 DESCRIPTION

40765 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 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 one of the following occurs:

- 40770 • An end-of-file condition is detected on standard input.
- 40771 • The logical end-of-file string (see the **-E eofstr** option) is found on standard input after double-quote processing, apostrophe processing, and backslash escape processing (see next paragraph).
- 40774 • An invocation of a constructed command line returns an exit status of 255.

40775 The application shall ensure that arguments in the standard input are separated by unquoted <blank>s, unescaped <blank>s, or <newline>s. A string of zero or more non-double-quote ('"') characters and non-<newline>s can be quoted by enclosing them in double-quotes. A string of zero or more non-apostrophe ('') characters and non-<newline>s can be quoted by enclosing them in apostrophes. Any unquoted character can be escaped by preceding it with a backslash. The utility named by *utility* shall be executed one or more times until the end-of-file is reached or the logical end-of file string is found. The results are unspecified if the utility named by *utility* attempts to read from its standard input.

40783 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 the System Interfaces volume of IEEE Std 1003.1-2001) shall not exceed {ARG_MAX}-2 048 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}.

40790 OPTIONS

40791 The *xargs* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section 12.2, Utility Syntax Guidelines.

40793 The following options shall be supported:

40794 **-E eofstr** Use *eofstr* as the logical end-of-file string. If **-E** is not specified, it is unspecified whether the logical end-of-file string is the underscore character ('_') or the end-of-file string capability is disabled. When *eofstr* is the null string, the logical end-of-file string capability shall be disabled and underscore characters shall be taken literally.

40799 XSI **-I replstr** Insert mode: *utility* is executed for each line from standard input, taking the entire line as a single argument, inserting it in *arguments* for each occurrence of *replstr*. A maximum of five arguments in *arguments* can each contain one or more instances of *replstr*. Any <blank>s at the beginning of each line shall be ignored. Constructed arguments cannot grow larger than 255 bytes. Option **-x** shall be forced on.

40805 XSI	-L <i>number</i>	The <i>utility</i> shall be executed for each non-empty <i>number</i> lines of arguments from standard input. The last invocation of <i>utility</i> shall be with fewer lines of arguments if fewer than <i>number</i> remain. A line is considered to end with the first <newline> unless the last character of the line is a <blank>; a trailing <blank> signals continuation to the next non-empty line, inclusive. The -L and -n options are mutually-exclusive; the last one specified shall take effect.
40811	-n <i>number</i>	Invoke <i>utility</i> using as many standard input arguments as possible, up to <i>number</i> (a positive decimal integer) arguments maximum. Fewer arguments shall be used if:
40813		• The command line length accumulated exceeds the size specified by the -s option (or {LINE_MAX} if there is no -s option).
40814		• The last iteration has fewer than <i>number</i> , but not zero, operands remaining.
40816	-p	Prompt mode: the user is asked whether to execute <i>utility</i> at each invocation. Trace mode (-t) is turned on to write the command instance to be executed, followed by a prompt to standard error. An affirmative response read from /dev/tty shall execute the command; otherwise, that particular invocation of <i>utility</i> shall be skipped.
40821	-s <i>size</i>	Invoke <i>utility</i> using as many standard input arguments as possible yielding a command line length less than <i>size</i> (a positive decimal integer) bytes. Fewer arguments shall be used if:
40824		• The total number of arguments exceeds that specified by the -n option.
40825 XSI		• The total number of lines exceeds that specified by the -L option.
40826		• End-of-file is encountered on standard input before <i>size</i> bytes are accumulated.
40827		Values of <i>size</i> up to at least {LINE_MAX} bytes shall be supported, provided that the constraints specified in the DESCRIPTION are met. It shall not be considered an error if a value larger than that supported by the implementation or exceeding the constraints specified in the DESCRIPTION is given; <i>xargs</i> shall use the largest value it supports within the constraints.
40832	-t	Enable trace mode. Each generated command line shall be written to standard error just prior to invocation.
40834	-x	Terminate if a command line containing <i>number</i> arguments (see the -n option above) or <i>number</i> lines (see the -L option above) will not fit in the implied or specified size (see the -s option above).
40835 XSI		
40836		

40837 OPERANDS

40838	The following operands shall be supported:	
40839	<i>utility</i>	The name of the utility to be invoked, found by search path using the <i>PATH</i> environment variable, described in the Base Definitions volume of IEEE Std 1003.1-2001, Chapter 8, Environment Variables. If <i>utility</i> is omitted, the default shall be the <i>echo</i> utility. If the <i>utility</i> operand names any of the special built-in utilities in Section 2.14 (on page 64), the results are undefined.
40844	<i>argument</i>	An initial option or operand for the invocation of <i>utility</i> .

40845 STDIN

40846	The standard input shall be a text file. The results are unspecified if an end-of-file condition is detected immediately following an escaped <newline>.	
40847		

40848 INPUT FILES

40849 The file `/dev/tty` shall be used to read responses required by the `-p` option.

40850 ENVIRONMENT VARIABLES

40851 The following environment variables shall affect the execution of *xargs*:

40852 *LANG* Provide a default value for the internationalization variables that are unset or null.
(See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2, Internationalization Variables for the precedence of internationalization variables used to determine the values of locale categories.)

40856 *LC_ALL* If set to a non-empty string value, override the values of all the other internationalization variables.

40858 *LC_COLLATE* Determine the locale for the behavior of ranges, equivalence classes, and multi-character collating elements used in the extended regular expression defined for the `yesexpr` locale keyword in the *LC_MESSAGES* category.

40862 *LC_CTYPE* Determine the locale for the interpretation of sequences of bytes of text data as characters (for example, single-byte as opposed to multi-byte characters in arguments and input files) and the behavior of character classes used in the extended regular expression defined for the `yesexpr` locale keyword in the *LC_MESSAGES* category.

40867 *LC_MESSAGES* Determine the locale for the processing of affirmative responses and that should be used to affect the format and contents of diagnostic messages written to standard error.

40871 XSI *NLSPATH* Determine the location of message catalogs for the processing of *LC_MESSAGES*.

40872 *PATH* Determine the location of *utility*, as described in the Base Definitions volume of IEEE Std 1003.1-2001, Chapter 8, Environment Variables.

40874 ASYNCHRONOUS EVENTS

40875 Default.

40876 STDOUT

40877 Not used.

40878 STDERR

40879 The standard error shall be used for diagnostic messages and the `-t` and `-p` options. If the `-t` option is specified, the *utility* and its constructed argument list shall be written to standard error, as it will be invoked, prior to invocation. If `-p` is specified, a prompt of the following format shall be written (in the POSIX locale):

40883 " ? . . . "

40884 at the end of the line of the output from `-t`.

40885 OUTPUT FILES

40886 None.

40887 EXTENDED DESCRIPTION

40888 None.

40889 **EXIT STATUS**

40890 The following exit values shall be returned:

- 40891 0 All invocations of *utility* returned exit status zero.
- 40892 1-125 A command line meeting the specified requirements could not be assembled, one or more of the invocations of *utility* returned a non-zero exit status, or some other error occurred.
- 40895 126 The utility specified by *utility* was found but could not be invoked.
- 40896 127 The utility specified by *utility* could not be found.

40897 **CONSEQUENCES OF ERRORS**

40898 If a command line meeting the specified requirements cannot be assembled, the utility cannot be
40899 invoked, an invocation of the utility is terminated by a signal, or an invocation of the utility exits
40900 with exit status 255, the *xargs* utility shall write a diagnostic message and exit without
40901 processing any remaining input.

40902 **APPLICATION USAGE**

40903 The 255 exit status allows a utility being used by *xargs* to tell *xargs* to terminate if it knows no
40904 further invocations using the current data stream will succeed. Thus, *utility* should explicitly *exit*
40905 with an appropriate value to avoid accidentally returning with 255.

40906 Note that input is parsed as lines; <blank>s separate arguments. If *xargs* is used to bundle output
40907 of commands like *find dir -print* or *ls* into commands to be executed, unexpected results are
40908 likely if any filenames contain any <blank>s or <newline>s. This can be fixed by using *find* to
40909 call a script that converts each file found into a quoted string that is then piped to *xargs*. Note
40910 that the quoting rules used by *xargs* are not the same as in the shell. They were not made
40911 consistent here because existing applications depend on the current rules and the shell syntax is
40912 not fully compatible with it. An easy rule that can be used to transform any string into a quoted
40913 form that *xargs* interprets correctly is to precede each character in the string with a backslash.

40914 On implementations with a large value for {ARG_MAX}, *xargs* may produce command lines
40915 longer than {LINE_MAX}. For invocation of utilities, this is not a problem. If *xargs* is being used
40916 to create a text file, users should explicitly set the maximum command line length with the *-s*
40917 option.

40918 The *command*, *env*, *nice*, *nohup*, *time*, and *xargs* utilities have been specified to use exit code 127 if
40919 an error occurs so that applications can distinguish “failure to find a utility” from “invoked
40920 utility exited with an error indication”. The value 127 was chosen because it is not commonly
40921 used for other meanings; most utilities use small values for “normal error conditions” and the
40922 values above 128 can be confused with termination due to receipt of a signal. The value 126 was
40923 chosen in a similar manner to indicate that the utility could be found, but not invoked. Some
40924 scripts produce meaningful error messages differentiating the 126 and 127 cases. The distinction
40925 between exit codes 126 and 127 is based on KornShell practice that uses 127 when all attempts to
40926 *exec* the utility fail with [ENOENT], and uses 126 when any attempt to *exec* the utility fails for
40927 any other reason.

40928 **EXAMPLES**

- 40929 1. The following command combines the output of the parenthesised commands onto one
40930 line, which is then written to the end-of-file **log**:

```
40931 (logname; date; printf "%s\n" "$0 $*") | xargs >>log
```

- 40932 2. The following command invokes *diff* with successive pairs of arguments originally typed
40933 as command line arguments (assuming there are no embedded <blank>s in the elements of
40934 the original argument list):

```
40935     printf "%s\n" "$*" | xargs -n 2 -x diff
40936 3. In the following commands, the user is asked which files in the current directory are to be
40937      archived. The files are archived into arch; a, one at a time, or b, many at a time.
40938      a. ls | xargs -p -L 1 ar -r arch
40939      b. ls | xargs -p -L 1 | xargs ar -r arch
40940 4. The following executes with successive pairs of arguments originally typed as command
40941      line arguments:
40942      echo $* | xargs -n 2 diff
40943 5. On XSI-conformant systems, the following moves all files from directory $1 to directory $2,
40944      and echoes each move command just before doing it:
40945      ls $1 | xargs -I {} -t mv ${1/{} ${2/{}}}
```

40946 RATIONALE

40947 The *xargs* utility was usually found only in System V-based systems; BSD systems included an
40948 *apply* utility that provided functionality similar to *xargs -n number*. The SVID lists *xargs* as a
40949 software development extension. This volume of IEEE Std 1003.1-2001 does not share the view
40950 that it is used only for development, and therefore it is not optional.

40951 The classic application of the *xargs* utility is in conjunction with the *find* utility to reduce the
40952 number of processes launched by a simplistic use of the *find -exec* combination. The *xargs* utility
40953 is also used to enforce an upper limit on memory required to launch a process. With this basis in
40954 mind, this volume of IEEE Std 1003.1-2001 selected only the minimal features required.

40955 Although the 255 exit status is mostly an accident of historical implementations, it allows a
40956 utility being used by *xargs* to tell *xargs* to terminate if it knows no further invocations using the
40957 current data stream shall succeed. Any non-zero exit status from a utility falls into the 1-125
40958 range when *xargs* exits. There is no statement of how the various non-zero utility exit status
40959 codes are accumulated by *xargs*. The value could be the addition of all codes, their highest
40960 value, the last one received, or a single value such as 1. Since no algorithm is arguably better
40961 than the others, and since many of the standard utilities say little more (portably) than
40962 "pass/fail", no new algorithm was invented.

40963 Several other *xargs* options were withdrawn because simple alternatives already exist within this
40964 volume of IEEE Std 1003.1-2001. For example, the **-i replstr** option can be just as efficiently
40965 performed using a shell **for** loop. Since *xargs* calls an **exec** function with each input line, the **-i**
40966 option does not usually exploit the grouping capabilities of *xargs*.

40967 The requirement that *xargs* never produces command lines such that invocation of *utility* is
40968 within 2 048 bytes of hitting the POSIX *exec {ARG_MAX}* limitations is intended to guarantee
40969 that the invoked utility has room to modify its environment variables and command line
40970 arguments and still be able to invoke another utility. Note that the minimum *{ARG_MAX}*
40971 allowed by the System Interfaces volume of IEEE Std 1003.1-2001 is 4 096 bytes and the
40972 minimum value allowed by this volume of IEEE Std 1003.1-2001 is 2 048 bytes; therefore, the
40973 2 048 bytes difference seems reasonable. Note, however, that *xargs* may never be able to invoke a
40974 utility if the environment passed in to *xargs* comes close to using *{ARG_MAX}* bytes.

40975 The version of *xargs* required by this volume of IEEE Std 1003.1-2001 is required to wait for the
40976 completion of the invoked command before invoking another command. This was done because
40977 historical scripts using *xargs* assumed sequential execution. Implementations wanting to provide
40978 parallel operation of the invoked utilities are encouraged to add an option enabling parallel
40979 invocation, but should still wait for termination of all of the children before *xargs* terminates
40980 normally.

40981 The **-e** option was omitted from the ISO POSIX-2:1993 standard in the belief that the *eofstr*
40982 option-argument was recognized only when it was on a line by itself and before quote and
40983 escape processing were performed, and that the logical end-of-file processing was only enabled
40984 if a **-e** option was specified. In that case, a simple *sed* script could be used to duplicate the **-e**
40985 functionality. Further investigation revealed that:

- 40986 • The logical end-of-file string was checked for after quote and escape processing, making a *sed*
40987 script that provided equivalent functionality much more difficult to write.
- 40988 • The default was to perform logical end-of-file processing with an underscore as the logical
40989 end-of-file string.

40990 To correct this misunderstanding, the **-E** *eofstr* option was adopted from the X/Open Portability
40991 Guide. Users should note that the description of the **-E** option matches historical documentation
40992 of the **-e** option (which was not adopted because it did not support the Utility Syntax
40993 Guidelines), by saying that if *eofstr* is the null string, logical end-of-file processing is disabled.
40994 Historical implementations of *xargs* actually did not disable logical end-of-file processing; they
40995 treated a null argument found in the input as a logical end-of-file string. (A null *string* argument
40996 could be generated using single or double quotes (' ' or " "). Since this behavior was not
40997 documented historically, it is considered to be a bug.

40998 FUTURE DIRECTIONS

40999 None.

41000 SEE ALSO

41001 Chapter 2 (on page 29), *echo*, *find*, the System Interfaces volume of IEEE Std 1003.1-2001, *exec*

41002 CHANGE HISTORY

41003 First released in Issue 2.

41004 Issue 5

41005 A second FUTURE DIRECTION is added.

41006 Issue 6

41007 The obsolescent **-e**, **-i**, and **-l** options are removed.

41008 The following new requirements on POSIX implementations derive from alignment with the
41009 Single UNIX Specification:

- 41010 • The **-p** option is added.
- 41011 • In the INPUT FILES section, the file **/dev/tty** is used to read responses required by the **-p**
41012 option.
- 41013 • The STDERR section is updated to describe the **-p** option.

41014 The description of the **-E** option is aligned with the ISO POSIX-2:1993 standard.

41015 The normative text is reworded to avoid use of the term “must” for application requirements.

41016 NAME

41017 yacc — yet another compiler compiler (**DEVELOPMENT**)

41018 SYNOPSIS

41019 CD `yacc [-dltv][-b file_prefix][-p sym_prefix] grammar`

41020

41021 DESCRIPTION

41022 The *yacc* utility shall read a description of a context-free grammar in *grammar* and write C source code, conforming to the ISO C standard, to a code file, and optionally header information into a header file, in the current directory. The C code shall define a function and related routines and macros for an automaton that executes a parsing algorithm meeting the requirements in **Algorithms** (on page 1075).

41027 The form and meaning of the grammar are described in the EXTENDED DESCRIPTION section.

41028 The C source code and header file shall be produced in a form suitable as input for the C compiler (see *c99*).

41030 OPTIONS

41031 The *yacc* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section 12.2, Utility Syntax Guidelines.

41033 The following options shall be supported:

41034 **-b** *file_prefix* Use *file_prefix* instead of *y* as the prefix for all output filenames. The code file *y.tab.c*, the header file *y.tab.h* (created when **-d** is specified), and the description file *y.output* (created when **-v** is specified), shall be changed to *file_prefix.tab.c*, *file_prefix.tab.h*, and *file_prefix.output*, respectively.

41038 **-d** Write the header file; by default only the code file is written. The **#define** statements associate the token codes assigned by *yacc* with the user-declared token names. This allows source files other than *y.tab.c* to access the token codes.

41041 **-l** Produce a code file that does not contain any **#line** constructs. If this option is not present, it is unspecified whether the code file or header file contains **#line** directives. This should only be used after the grammar and the associated actions are fully debugged.

41045 **-p** *sym_prefix* Use *sym_prefix* instead of *yy* as the prefix for all external names produced by *yacc*. The names affected shall include the functions *yparse()*, *yylex()*, and *yyerror()*, and the variables *yylval*, *ychar*, and *yydebug*. (In the remainder of this section, the six symbols cited are referenced using their default names only as a notational convenience.) Local names may also be affected by the **-p** option; however, the **-p** option shall not affect **#define** symbols generated by *yacc*.

41052 **-t** Modify conditional compilation directives to permit compilation of debugging code in the code file. Runtime debugging statements shall always be contained in the code file, but by default conditional compilation directives prevent their compilation.

41056 **-v** Write a file containing a description of the parser and a report of conflicts generated by ambiguities in the grammar.

41058 OPERANDS

41059 The following operand is required:

41060 *grammar* A pathname of a file containing instructions, hereafter called *grammar*, for which a parser is to be created. The format for the grammar is described in the EXTENDED DESCRIPTION section.

41063 STDIN

41064 Not used.

41065 INPUT FILES

41066 The file *grammar* shall be a text file formatted as specified in the EXTENDED DESCRIPTION section.

41068 ENVIRONMENT VARIABLES

41069 The following environment variables shall affect the execution of yacc:

41070 *LANG* Provide a default value for the internationalization variables that are unset or null. (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2, Internationalization Variables for the precedence of internationalization variables used to determine the values of locale categories.)

41074 *LC_ALL* If set to a non-empty string value, override the values of all the other internationalization variables.

41076 *LC_CTYPE* Determine the locale for the interpretation of sequences of bytes of text data as characters (for example, single-byte as opposed to multi-byte characters in arguments and input files).

41079 *LC_MESSAGES*

41080 Determine the locale that should be used to affect the format and contents of diagnostic messages written to standard error.

41082 XSI *NLSPATH* Determine the location of message catalogs for the processing of *LC_MESSAGES*.

41083 The *LANG* and *LC_** variables affect the execution of the yacc utility as stated. The *main()* function defined in **Yacc Library** (on page 1075) shall call:

41085 `setlocale(LC_ALL, "")`

41086 and thus the program generated by yacc shall also be affected by the contents of these variables at runtime.

41088 ASYNCHRONOUS EVENTS

41089 Default.

41090 STDOUT

41091 Not used.

41092 STDERR

41093 If shift/reduce or reduce/reduce conflicts are detected in *grammar*, yacc shall write a report of those conflicts to the standard error in an unspecified format.

41095 Standard error shall also be used for diagnostic messages.

41096 OUTPUT FILES

41097 The code file, the header file, and the description file shall be text files. All are described in the following sections.

41099

Code File41100
41101
41102
41103
41104

This file shall contain the C source code for the *yyparse()* function. It shall contain code for the various semantic actions with macro substitution performed on them as described in the EXTENDED DESCRIPTION section. It also shall contain a copy of the **#define** statements in the header file. If a **%union** declaration is used, the declaration for YYSTYPE shall also be included in this file.

41105

Header File41106
41107
41108
41109

The header file shall contain **#define** statements that associate the token numbers with the token names. This allows source files other than the code file to access the token codes. If a **%union** declaration is used, the declaration for YYSTYPE and an *extern YYSTYPE yylval* declaration shall also be included in this file.

41110

Description File41111
41112
41113
41114
41115

The description file shall be a text file containing a description of the state machine corresponding to the parser, using an unspecified format. Limits for internal tables (see **Limits** (on page 1076)) shall also be reported, in an implementation-defined manner. (Some implementations may use dynamic allocation techniques and have no specific limit values to report.)

41116

EXTENDED DESCRIPTION41117
41118
41119

The *yacc* command accepts a language that is used to define a grammar for a target language to be parsed by the tables and code generated by *yacc*. The language accepted by *yacc* as a grammar for the target language is described below using the *yacc* input language itself.

41120
41121
41122
41123
41124
41125

The input *grammar* includes rules describing the input structure of the target language and code to be invoked when these rules are recognized to provide the associated semantic action. The code to be executed shall appear as bodies of text that are intended to be C-language code. The C-language inclusions are presumed to form a correct function when processed by *yacc* into its output files. The code included in this way shall be executed during the recognition of the target language.

41126
41127
41128
41129
41130
41131
41132
41133

Given a grammar, the *yacc* utility generates the files described in the OUTPUT FILES section. The code file can be compiled and linked using *c99*. If the declaration and programs sections of the grammar file did not include definitions of *main()*, *yylex()*, and *yyerror()*, the compiled output requires linking with externally supplied versions of those functions. Default versions of *main()* and *yyerror()* are supplied in the *yacc* library and can be linked in by using the **-ly** operand to *c99*. The *yacc* library interfaces need not support interfaces with other than the default *yy* symbol prefix. The application provides the lexical analyzer function, *yylex()*; the *lex* utility is specifically designed to generate such a routine.

41134

Input Language41135
41136
41137
41138

The application shall ensure that every specification file consists of three sections in order: *declarations*, *grammar rules*, and *programs*, separated by double percent signs ("%%"). The declarations and programs sections can be empty. If the latter is empty, the preceding "%%" mark separating it from the rules section can be omitted.

41139

The input is free form text following the structure of the grammar defined below.

41140 **Lexical Structure of the Grammar**

41141 The <blank>s, <newline>s, and <form-feed>s shall be ignored, except that the application shall
 41142 ensure that they do not appear in names or multi-character reserved symbols. Comments shall
 41143 be enclosed in "/* . . . */", and can appear wherever a name is valid.

41144 Names are of arbitrary length, made up of letters, periods ('.'), underscores ('_'), and non-
 41145 initial digits. Uppercase and lowercase letters are distinct. Conforming applications shall not
 41146 use names beginning in yy or YY since the yacc parser uses such names. Many of the names
 41147 appear in the final output of yacc, and thus they should be chosen to conform with any
 41148 additional rules created by the C compiler to be used. In particular they appear in #define
 41149 statements.

41150 A literal shall consist of a single character enclosed in single-quotes (''). All of the escape
 41151 sequences supported for character constants by the ISO C standard shall be supported by yacc.

41152 The relationship with the lexical analyzer is discussed in detail below.

41153 The application shall ensure that the NUL character is not used in grammar rules or literals.

41154 **Declarations Section**

41155 The declarations section is used to define the symbols used to define the target language and
 41156 their relationship with each other. In particular, much of the additional information required to
 41157 resolve ambiguities in the context-free grammar for the target language is provided here.

41158 Usually yacc assigns the relationship between the symbolic names it generates and their
 41159 underlying numeric value. The declarations section makes it possible to control the assignment
 41160 of these values.

41161 It is also possible to keep semantic information associated with the tokens currently on the parse
 41162 stack in a user-defined C-language **union**, if the members of the union are associated with the
 41163 various names in the grammar. The declarations section provides for this as well.

41164 The first group of declarators below all take a list of names as arguments. That list can optionally
 41165 be preceded by the name of a C union member (called a *tag* below) appearing within '<' and
 41166 '>'. (As an exception to the typographical conventions of the rest of this volume of
 41167 IEEE Std 1003.1-2001, in this case <*tag*> does not represent a metavariable, but the literal angle
 41168 bracket characters surrounding a symbol.) The use of *tag* specifies that the tokens named on this
 41169 line shall be of the same C type as the union member referenced by *tag*. This is discussed in
 41170 more detail below.

41171 For lists used to define tokens, the first appearance of a given token can be followed by a
 41172 positive integer (as a string of decimal digits). If this is done, the underlying value assigned to it
 41173 for lexical purposes shall be taken to be that number.

41174 The following declares *name* to be a token:

41175 %token [<*tag*>] *name* [*number*][*name* [*number*]...]

1

41176 If *tag* is present, the C type for all tokens on this line shall be declared to be the type referenced
 41177 by *tag*. If a positive integer, *number*, follows a *name*, that value shall be assigned to the token.

41178 The following declares *name* to be a token, and assigns precedence to it:

41179 %left [<*tag*>] *name* [*number*][*name* [*number*]...]
 41180 %right [<*tag*>] *name* [*number*][*name* [*number*]...]

41181 One or more lines, each beginning with one of these symbols, can appear in this section. All
 41182 tokens on the same line have the same precedence level and associativity; the lines are in order

41183 of increasing precedence or binding strength. **%left** denotes that the operators on that line are
41184 left associative, and **%right** similarly denotes right associative operators. If **tag** is present, it shall
41185 declare a C type for *names* as described for **%token**.

41186 The following declares *name* to be a token, and indicates that this cannot be used associatively:

41187 **%nonassoc [<tag>] name [number][name [number]]...**

41188 If the parser encounters associative use of this token it reports an error. If **tag** is present, it shall
41189 declare a C type for *names* as described for **%token**.

41190 The following declares that union member *names* are non-terminals, and thus it is required to
41191 have a **tag** field at its beginning:

41192 **%type <tag> name...**

41193 Because it deals with non-terminals only, assigning a token number or using a literal is also
41194 prohibited. If this construct is present, *yacc* shall perform type checking; if this construct is not
41195 present, the parse stack shall hold only the **int** type.

41196 Every name used in *grammar* not defined by a **%token**, **%left**, **%right**, or **%nonassoc** declaration
41197 is assumed to represent a non-terminal symbol. The *yacc* utility shall report an error for any
41198 non-terminal symbol that does not appear on the left side of at least one grammar rule.

41199 Once the type, precedence, or token number of a name is specified, it shall not be changed. If the
41200 first declaration of a token does not assign a token number, *yacc* shall assign a token number.
41201 Once this assignment is made, the token number shall not be changed by explicit assignment.

41202 The following declarators do not follow the previous pattern.

41203 The following declares the non-terminal *name* to be the *start symbol*, which represents the largest,
41204 most general structure described by the grammar rules:

41205 **%start name**

41206 By default, it is the left-hand side of the first grammar rule; this default can be overridden with
41207 this declaration.

41208 The following declares the *yacc* value stack to be a union of the various types of values desired:

41209 **%union { body of union (in C) }**

41210 By default, the values returned by actions (see below) and the lexical analyzer shall be of type
41211 **int**. The *yacc* utility keeps track of types, and it shall insert corresponding union member names
41212 in order to perform strict type checking of the resulting parser.

41213 Alternatively, given that at least one **<tag>** construct is used, the union can be declared in a
41214 header file (which shall be included in the declarations section by using a **#include** construct
41215 within **%{** and **%}**), and a **typedef** used to define the symbol **YYSTYPE** to represent this union.
41216 The effect of **%union** is to provide the declaration of **YYSTYPE** directly from the *yacc* input.

41217 C-language declarations and definitions can appear in the declarations section, enclosed by the
41218 following marks:

41219 **%{ ... %}**

41220 These statements shall be copied into the code file, and have global scope within it so that they
41221 can be used in the rules and program sections.

41222 The application shall ensure that the declarations section is terminated by the token **%%**.

- 41223 **Grammar Rules in yacc**
- 41224 The rules section defines the context-free grammar to be accepted by the function *yacc* generates,
 41225 and associates with those rules C-language actions and additional precedence information. The
 41226 grammar is described below, and a formal definition follows.
- 41227 The rules section is comprised of one or more grammar rules. A grammar rule has the form:
- 41228 A : BODY ;
- 41229 The symbol **A** represents a non-terminal name, and **BODY** represents a sequence of zero or
 41230 more *names*, *literals*, and *semantic actions* that can then be followed by optional *precedence rules*.
 41231 Only the names and literals participate in the formation of the grammar; the semantic actions
 41232 and precedence rules are used in other ways. The colon and the semicolon are *yacc* punctuation.
 41233 If there are several successive grammar rules with the same left-hand side, the vertical bar ' | '
 41234 can be used to avoid rewriting the left-hand side; in this case the semicolon appears only after
 41235 the last rule. The BODY part can be empty (or empty of names and literals) to indicate that the
 41236 non-terminal symbol matches the empty string.
- 41237 The *yacc* utility assigns a unique number to each rule. Rules using the vertical bar notation are
 41238 distinct rules. The number assigned to the rule appears in the description file.
- 41239 The elements comprising a BODY are:
- 41240 *name, literal* These form the rules of the grammar: *name* is either a *token* or a *non-terminal*; *literal*
 41241 stands for itself (less the lexically required quotation marks).
- 41242 *semantic action*
- 41243 With each grammar rule, the user can associate actions to be performed each time
 41244 the rule is recognized in the input process. (Note that the word "action" can also
 41245 refer to the actions of the parser—shift, reduce, and so on.)
- 41246 These actions can return values and can obtain the values returned by previous
 41247 actions. These values are kept in objects of type YYSTYPE (see %union). The
 41248 result value of the action shall be kept on the parse stack with the left-hand side of
 41249 the rule, to be accessed by other reductions as part of their right-hand side. By
 41250 using the <tag> information provided in the declarations section, the code
 41251 generated by *yacc* can be strictly type checked and contain arbitrary information. In
 41252 addition, the lexical analyzer can provide the same kinds of values for tokens, if
 41253 desired.
- 41254 An action is an arbitrary C statement and as such can do input or output, call
 41255 subprograms, and alter external variables. An action is one or more C statements
 41256 enclosed in curly braces '{' and '}'.
- 41257 Certain pseudo-variables can be used in the action. These are macros for access to
 41258 data structures known internally to *yacc*.
- 41259 *\$\$* The value of the action can be set by assigning it to *\$\$*. If type
 41260 checking is enabled and the type of the value to be assigned cannot
 41261 be determined, a diagnostic message may be generated.
- 41262 *\$number* This refers to the value returned by the component specified by the
 41263 token *number* in the right side of a rule, reading from left to right;
 41264 *number* can be zero or negative. If *number* is zero or negative, it refers
 41265 to the data associated with the name on the parser's stack preceding
 41266 the leftmost symbol of the current rule. (That is, "\$0" refers to the
 41267 name immediately preceding the leftmost name in the current rule to
 41268 be found on the parser's stack and "\$-1" refers to the symbol to its

41269
41270
41271
41272

left.) If *number* refers to an element past the current point in the rule, or beyond the bottom of the stack, the result is undefined. If type checking is enabled and the type of the value to be assigned cannot be determined, a diagnostic message may be generated.

41273 \$<tag>*number*
41274
41275
41276
41277
41278

These correspond exactly to the corresponding symbols without the *tag* inclusion, but allow for strict type checking (and preclude unwanted type conversions). The effect is that the macro is expanded to use *tag* to select an element from the YYSTYPE union (using *dataname.tag*). This is particularly useful if *number* is not positive.

41279 \$<tag>\$
41280
41281
41282

This imposes on the reference the type of the union member referenced by *tag*. This construction is applicable when a reference to a left context value occurs in the grammar, and provides yacc with a means for selecting a type.

41283
41284
41285
41286
41287
41288
41289
41290

Actions can occur anywhere in a rule (not just at the end); an action can access values returned by actions to its left, and in turn the value it returns can be accessed by actions to its right. An action appearing in the middle of a rule shall be equivalent to replacing the action with a new non-terminal symbol and adding an empty rule with that non-terminal symbol on the left-hand side. The semantic action associated with the new rule shall be equivalent to the original action. The use of actions within rules might introduce conflicts that would not otherwise exist.

41291
41292
41293

By default, the value of a rule shall be the value of the first element in it. If the first element does not have a type (particularly in the case of a literal) and type checking is turned on by %type, an error message shall result.

41294 precedence
41295
41296
41297
41298
41299
41300
41301

The keyword %prec can be used to change the precedence level associated with a particular grammar rule. Examples of this are in cases where a unary and binary operator have the same symbolic representation, but need to be given different precedences, or where the handling of an ambiguous if-else construction is necessary. The reserved symbol %prec can appear immediately after the body of the grammar rule and can be followed by a token name or a literal. It shall cause the precedence of the grammar rule to become that of the following token name or literal. The action for the rule as a whole can follow %prec.

41302
41303

If a program section follows, the application shall ensure that the grammar rules are terminated by %%.

41304 Programs Section

41305
41306
41307
41308
41309
41310

The programs section can include the definition of the lexical analyzer yylex(), and any other functions; for example, those used in the actions specified in the grammar rules. It is unspecified whether the programs section precedes or follows the semantic actions in the output file; therefore, if the application contains any macro definitions and declarations intended to apply to the code in the semantic actions, it shall place them within "%{ . . . %}" in the declarations section.

41311 **Input Grammar**

41312 The following input to yacc yields a parser for the input to yacc. This formal syntax takes
 41313 precedence over the preceding text syntax description.

41314 The lexical structure is defined less precisely; **Lexical Structure of the Grammar** (on page 1067)
 41315 defines most terms. The correspondence between the previous terms and the tokens below is as
 41316 follows.

41317 **IDENTIFIER** This corresponds to the concept of *name*, given previously. It also includes
 41318 literals as defined previously.

41319 **C_IDENTIFIER** This is a name, and additionally it is known to be followed by a colon. A literal
 41320 cannot yield this token.

41321 **NUMBER** A string of digits (a non-negative decimal integer).

41322 **TYPE, LEFT, MARK, LCURL, RCURL**

41323 These correspond directly to %type, %left, %% , %{, and %}.

41324 **{...}** This indicates C-language source code, with the possible inclusion of '\$'
 41325 macros as discussed previously.

41326 /* Grammar for the input to yacc. */

41327 /* Basic entries. */

41328 /* The following are recognized by the lexical analyzer. */

41329 %token IDENTIFIER /* Includes identifiers and literals */

41330 %token C_IDENTIFIER /* identifier (but not literal)
 followed by a :. */

41332 %token NUMBER /* [0-9][0-9]* */

41333 /* Reserved words : %type=>TYPE %left=>LEFT, and so on */

41334 %token LEFT RIGHT NONASSOC TOKEN PREC TYPE START UNION

41335 %token MARK /* The %% mark. */

41336 %token LCURL /* The %{ mark. */

41337 %token RCURL /* The %} mark. */

41338 /* 8-bit character literals stand for themselves; */

41339 /* tokens have to be defined for multi-byte characters. */

41340 %start spec

41341 %%

41342 spec : defs MARK rules tail

41343 ;

41344 tail : MARK

41345 {

41346 /* In this action, set up the rest of the file. */

41347 }

41348 | /* Empty; the second MARK is optional. */

41349 ;

41350 defs : /* Empty. */

41351 | defs def

41352 ;

41353 def : START IDENTIFIER

41354 | UNION

```
41355      {
41356          /* Copy union definition to output. */
41357      }
41358      | LCURL
41359      {
41360          /* Copy C code to output file. */
41361      }
41362      RCURL
41363      | rword tag nlist
41364      ;
41365      rword : TOKEN
41366      | LEFT
41367      | RIGHT
41368      | NONASSOC
41369      | TYPE
41370      ;
41371      tag : /* Empty: union tag ID optional. */
41372      | '<' IDENTIFIER '>'
41373      ;
41374      nlist : nmno
41375      | nlist nmno
41376      ;
41377      nmno : IDENTIFIER           /* Note: literal invalid with % type. */
41378      | IDENTIFIER NUMBER        /* Note: invalid with % type. */
41379      ;
41380      /* Rule section */
41381      rules : C_IDENTIFIER rbody prec
41382      | rules rule
41383      ;
41384      rule : C_IDENTIFIER rbody prec
41385      | '|' rbody prec
41386      ;
41387      rbody : /* empty */
41388      | rbody IDENTIFIER
41389      | rbody act
41390      ;
41391      act : '{'
41392      {
41393          /* Copy action, translate $$, and so on. */
41394      }
41395      '}'
41396      ;
41397      prec : /* Empty */
41398      | PREC IDENTIFIER
41399      | PREC IDENTIFIER act
41400      | prec ';'
41401      ;
```

41402

Conflicts

41403

The parser produced for an input grammar may contain states in which conflicts occur. The conflicts occur because the grammar is not LALR(1). An ambiguous grammar always contains at least one LALR(1) conflict. The *yacc* utility shall resolve all conflicts, using either default rules or user-specified precedence rules.

41407

Conflicts are either shift/reduce conflicts or reduce/reduce conflicts. A shift/reduce conflict is where, for a given state and lookahead symbol, both a shift action and a reduce action are possible. A reduce/reduce conflict is where, for a given state and lookahead symbol, reductions by two different rules are possible.

41411

The rules below describe how to specify what actions to take when a conflict occurs. Not all shift/reduce conflicts can be successfully resolved this way because the conflict may be due to something other than ambiguity, so incautious use of these facilities can cause the language accepted by the parser to be much different from that which was intended. The description file shall contain sufficient information to understand the cause of the conflict. Where ambiguity is the reason either the default or explicit rules should be adequate to produce a working parser.

41417

The declared precedences and associativities (see **Declarations Section** (on page 1067)) are used to resolve parsing conflicts as follows:

1. A precedence and associativity is associated with each grammar rule; it is the precedence and associativity of the last token or literal in the body of the rule. If the %prec keyword is used, it overrides this default. Some grammar rules might not have both precedence and associativity.
2. If there is a shift/reduce conflict, and both the grammar rule and the input symbol have precedence and associativity associated with them, then the conflict is resolved in favor of the action (shift or reduce) associated with the higher precedence. If the precedences are the same, then the associativity is used; left associative implies reduce, right associative implies shift, and non-associative implies an error in the string being parsed.
3. When there is a shift/reduce conflict that cannot be resolved by rule 2, the shift is done. Conflicts resolved this way are counted in the diagnostic output described in **Error Handling**.
4. When there is a reduce/reduce conflict, a reduction is done by the grammar rule that occurs earlier in the input sequence. Conflicts resolved this way are counted in the diagnostic output described in **Error Handling**.

41434

Conflicts resolved by precedence or associativity shall not be counted in the shift/reduce and reduce/reduce conflicts reported by *yacc* on either standard error or in the description file.

41436

Error Handling

41437

The token **error** shall be reserved for error handling. The name **error** can be used in grammar rules. It indicates places where the parser can recover from a syntax error. The default value of **error** shall be 256. Its value can be changed using a %token declaration. The lexical analyzer should not return the value of **error**.

41441

The parser shall detect a syntax error when it is in a state where the action associated with the lookahead symbol is **error**. A semantic action can cause the parser to initiate error handling by executing the macro YYERROR. When YYERROR is executed, the semantic action passes control back to the parser. YYERROR cannot be used outside of semantic actions.

41445

When the parser detects a syntax error, it normally calls *yyerror()* with the character string "syntax error" as its argument. The call shall not be made if the parser is still recovering

41447 from a previous error when the error is detected. The parser is considered to be recovering from
41448 a previous error until the parser has shifted over at least three normal input symbols since the
41449 last error was detected or a semantic action has executed the macro *yyerrok*. The parser shall not
41450 call *yyerror()* when YYERROR is executed.

41451 The macro function YYRECOVERING shall return 1 if a syntax error has been detected and the
41452 parser has not yet fully recovered from it. Otherwise, zero shall be returned.

41453 When a syntax error is detected by the parser, the parser shall check if a previous syntax error
41454 has been detected. If a previous error was detected, and if no normal input symbols have been
41455 shifted since the preceding error was detected, the parser checks if the lookahead symbol is an
41456 endmarker (see **Interface to the Lexical Analyzer**). If it is, the parser shall return with a non-
41457 zero value. Otherwise, the lookahead symbol shall be discarded and normal parsing shall
41458 resume.

41459 When YYERROR is executed or when the parser detects a syntax error and no previous error has
41460 been detected, or at least one normal input symbol has been shifted since the previous error was
41461 detected, the parser shall pop back one state at a time until the parse stack is empty or the
41462 current state allows a shift over **error**. If the parser empties the parse stack, it shall return with a
41463 non-zero value. Otherwise, it shall shift over **error** and then resume normal parsing. If the parser
41464 reads a lookahead symbol before the error was detected, that symbol shall still be the lookahead
41465 symbol when parsing is resumed.

41466 The macro *yyerrok* in a semantic action shall cause the parser to act as if it has fully recovered
41467 from any previous errors. The macro *yyclearin* shall cause the parser to discard the current
41468 lookahead token. If the current lookahead token has not yet been read, *yyclearin* shall have no
41469 effect.

41470 The macro YYACCEPT shall cause the parser to return with the value zero. The macro
41471 YYABORT shall cause the parser to return with a non-zero value.

41472 **Interface to the Lexical Analyzer**

41473 The *yylex()* function is an integer-valued function that returns a *token number* representing the
41474 kind of token read. If there is a value associated with the token returned by *yylex()* (see the
41475 discussion of *tag* above), it shall be assigned to the external variable *yyval*.

41476 If the parser and *yylex()* do not agree on these token numbers, reliable communication between
41477 them cannot occur. For (single-byte character) literals, the token is simply the numeric value of
41478 the character in the current character set. The numbers for other tokens can either be chosen by
41479 *yacc*, or chosen by the user. In either case, the **#define** construct of C is used to allow *yylex()* to
41480 return these numbers symbolically. The **#define** statements are put into the code file, and the
41481 header file if that file is requested. The set of characters permitted by *yacc* in an identifier is larger
41482 than that permitted by C. Token names found to contain such characters shall not be included in
41483 the **#define** declarations.

41484 If the token numbers are chosen by *yacc*, the tokens other than literals shall be assigned numbers
41485 greater than 256, although no order is implied. A token can be explicitly assigned a number by
41486 following its first appearance in the declarations section with a number. Names and literals not
41487 defined this way retain their default definition. All token numbers assigned by *yacc* shall be
41488 unique and distinct from the token numbers used for literals and user-assigned tokens. If
41489 duplicate token numbers cause conflicts in parser generation, *yacc* shall report an error;
41490 otherwise, it is unspecified whether the token assignment is accepted or an error is reported.

41491 The end of the input is marked by a special token called the *endmarker*, which has a token
41492 number that is zero or negative. (These values are invalid for any other token.) All lexical
41493 analyzers shall return zero or negative as a token number upon reaching the end of their input. If

41494 the tokens up to, but excluding, the endmarker form a structure that matches the start symbol,
41495 the parser shall accept the input. If the endmarker is seen in any other context, it shall be
41496 considered an error.

41497 **Completing the Program**

41498 In addition to *yyparse()* and *yylex()*, the functions *yyerror()* and *main()* are required to make a
41499 complete program. The application can supply *main()* and *yyerror()*, or those routines can be
41500 obtained from the *yacc* library.

41501 **Yacc Library**

41502 The following functions shall appear only in the *yacc* library accessible through the **-ly** operand
41503 to *c99*; they can therefore be redefined by a conforming application:

41504 **int main(void)**

41505 This function shall call *yyparse()* and exit with an unspecified value. Other actions within
41506 this function are unspecified.

41507 **int yyerror(const char *s)**

41508 This function shall write the NUL-terminated argument to standard error, followed by a
41509 <newline>.

41510 The order of the **-ly** and **-lI** operands given to *c99* is significant; the application shall either
41511 provide its own *main()* function or ensure that **-ly** precedes **-lI**.

41512 **Debugging the Parser**

41513 The parser generated by *yacc* shall have diagnostic facilities in it that can be optionally enabled
41514 at either compile time or at runtime (if enabled at compile time). The compilation of the runtime
41515 debugging code is under the control of YYDEBUG, a preprocessor symbol. If YYDEBUG has a
41516 non-zero value, the debugging code shall be included. If its value is zero, the code shall not be
41517 included.

41518 In parsers where the debugging code has been included, the external **int yydebug** can be used to
41519 turn debugging on (with a non-zero value) and off (zero value) at runtime. The initial value of
41520 *yydebug* shall be zero.

41521 When **-t** is specified, the code file shall be built such that, if YYDEBUG is not already defined at
41522 compilation time (using the *c99 -D YYDEBUG* option, for example), YYDEBUG shall be set
41523 explicitly to 1. When **-t** is not specified, the code file shall be built such that, if YYDEBUG is not
41524 already defined, it shall be set explicitly to zero.

41525 The format of the debugging output is unspecified but includes at least enough information to
41526 determine the shift and reduce actions, and the input symbols. It also provides information
41527 about error recovery.

41528 **Algorithms**

41529 The parser constructed by *yacc* implements an LALR(1) parsing algorithm as documented in the
41530 literature. It is unspecified whether the parser is table-driven or direct-coded.

41531 A parser generated by *yacc* shall never request an input symbol from *yylex()* while in a state
41532 where the only actions other than the error action are reductions by a single rule.

41533 The literature of parsing theory defines these concepts.

41534

Limits

41535

The *yacc* utility may have several internal tables. The minimum maximums for these tables are shown in the following table. The exact meaning of these values is implementation-defined. The implementation shall define the relationship between these values and between them and any error messages that the implementation may generate should it run out of space for any internal structure. An implementation may combine groups of these resources into a single pool as long as the total available to the user does not fall below the sum of the sizes specified by this section.

41541

Table 4-22 Internal Limits in yacc

41542

Limit	Minimum Maximum	Description
{INTERMS}	126	Number of tokens.
{NNONTERM}	200	Number of non-terminals.
{NPROD}	300	Number of rules.
{NSTATES}	600	Number of states.
{MEMSIZE}	5 200	Length of rules. The total length, in names (tokens and non-terminals), of all the rules of the grammar. The left-hand side is counted for each rule, even if it is not explicitly repeated, as specified in Grammar Rules in yacc (on page 1069).
{ACTSIZE}	4 000	Number of actions. “Actions” here (and in the description file) refer to parser actions (shift, reduce, and so on) not to semantic actions defined in Grammar Rules in yacc (on page 1069).

41559 EXIT STATUS

41560

The following exit values shall be returned:

41561

0 Successful completion.

41562

>0 An error occurred.

41563 CONSEQUENCES OF ERRORS

41564

If any errors are encountered, the run is aborted and *yacc* exits with a non-zero status. Partial code files and header files may be produced. The summary information in the description file shall always be produced if the **-v** flag is present.

41567 APPLICATION USAGE

41568

Historical implementations experience name conflicts on the names **yacc.tmp**, **yacc.acts**, **yacc.debug**, **y.tab.c**, **y.tab.h**, and **y.output** if more than one copy of *yacc* is running in a single directory at one time. The **-b** option was added to overcome this problem. The related problem of allowing multiple *yacc* parsers to be placed in the same file was addressed by adding a **-p** option to override the previously hard-coded yy variable prefix.

41573

The description of the **-p** option specifies the minimal set of function and variable names that cause conflict when multiple parsers are linked together. YYSTYPE does not need to be changed. Instead, the programmer can use **-b** to give the header files for different parsers different names, and then the file with the *yylex()* for a given parser can include the header for that parser. Names such as *yyclearerr* do not need to be changed because they are used only in the actions; they do not have linkage. It is possible that an implementation has other names, either internal ones for implementing things such as *yyclearerr*, or providing non-standard features that it wants to change with **-p**.

41581 Unary operators that are the same token as a binary operator in general need their precedence
41582 adjusted. This is handled by the **%prec** advisory symbol associated with the particular grammar
41583 rule defining that unary operator. (See **Grammar Rules in yacc** (on page 1069).) Applications
41584 are not required to use this operator for unary operators, but the grammars that do not require it
41585 are rare.

41586 EXAMPLES

41587 Access to the *yacc* library is obtained with library search operands to *c99*. To use the *yacc* library
41588 *main()*:

41589 *c99 y.tab.c -l y*

41590 Both the *lex* library and the *yacc* library contain *main()*. To access the *yacc main()*:

41591 *c99 y.tab.c lex.yy.c -l y -l 1*

41592 This ensures that the *yacc* library is searched first, so that its *main()* is used.

41593 The historical *yacc* libraries have contained two simple functions that are normally coded by the
41594 application programmer. These functions are similar to the following code:

```
41595 #include <locale.h>
41596 int main(void)
41597 {
41598     extern int yyparse();
41599
41600     setlocale(LC_ALL, "");
41601
41602     /* If the following parser is one created by lex, the
41603        application must be careful to ensure that LC_CTYPE
41604        and LC_COLLATE are set to the POSIX locale. */
41605     (void) yyparse();
41606     return (0);
41607 }
41608
41609 #include <stdio.h>
41610
41611 int yyerror(const char *msg)
41612 {
41613     (void) fprintf(stderr, "%s\n", msg);
41614     return (0);
41615 }
```

41612 RATIONALE

41613 The references in **Referenced Documents** (on page xxxvii) may be helpful in constructing the
41614 parser generator. The referenced DeRemer and Pennello article (along with the works it
41615 references) describes a technique to generate parsers that conform to this volume of
41616 IEEE Std 1003.1-2001. Work in this area continues to be done, so implementors should consult
41617 current literature before doing any new implementations. The original Knuth article is the
41618 theoretical basis for this kind of parser, but the tables it generates are impractically large for
41619 reasonable grammars and should not be used. The “equivalent to” wording is intentional to
41620 assure that the best tables that are LALR(1) can be generated.

41621 There has been confusion between the class of grammars, the algorithms needed to generate
41622 parsers, and the algorithms needed to parse the languages. They are all reasonably orthogonal.
41623 In particular, a parser generator that accepts the full range of LR(1) grammars need not generate
41624 a table any more complex than one that accepts SLR(1) (a relatively weak class of LR grammars)
41625 for a grammar that happens to be SLR(1). Such an implementation need not recognize the case,
41626 either; table compression can yield the SLR(1) table (or one even smaller than that) without

41627 recognizing that the grammar is SLR(1). The speed of an LR(1) parser for any class is dependent
41628 more upon the table representation and compression (or the code generation if a direct parser is
41629 generated) than upon the class of grammar that the table generator handles.

41630 The speed of the parser generator is somewhat dependent upon the class of grammar it handles.
41631 However, the original Knuth article algorithms for constructing LR parsers were judged by its
41632 author to be impractically slow at that time. Although full LR is more complex than LALR(1), as
41633 computer speeds and algorithms improve, the difference (in terms of acceptable wall-clock
41634 execution time) is becoming less significant.

41635 Potential authors are cautioned that the referenced DeRemer and Pennello article previously
41636 cited identifies a bug (an over-simplification of the computation of LALR(1) lookahead sets) in
41637 some of the LALR(1) algorithm statements that preceded it to publication. They should take the
41638 time to seek out that paper, as well as current relevant work, particularly Aho's.

41639 The **-b** option was added to provide a portable method for permitting yacc to work on multiple
41640 separate parsers in the same directory. If a directory contains more than one yacc grammar, and
41641 both grammars are constructed at the same time (by, for example, a parallel make program),
41642 conflict results. While the solution is not historical practice, it corrects a known deficiency in
41643 historical implementations. Corresponding changes were made to all sections that referenced
41644 the filenames **y.tab.c** (now "the code file"), **y.tab.h** (now "the header file"), and **y.output** (now
41645 "the description file").

41646 The grammar for yacc input is based on System V documentation. The textual description shows
41647 there that the ';' is required at the end of the rule. The grammar and the implementation do not
41648 require this. (The use of **C_IDENTIFIER** causes a reduce to occur in the right place.)

41649 Also, in that implementation, the constructs such as **%token** can be terminated by a semicolon,
41650 but this is not permitted by the grammar. The keywords such as **%token** can also appear in
41651 uppercase, which is again not discussed. In most places where '%' is used, '\' can be
41652 substituted, and there are alternate spellings for some of the symbols (for example, **%LEFT** can
41653 be "%<" or even "\<").

41654 Historically, <tag> can contain any characters except '>', including white space, in the
41655 implementation. However, since the tag must reference an ISO C standard union member, in
41656 practice conforming implementations need to support only the set of characters for ISO C
41657 standard identifiers in this context.

41658 Some historical implementations are known to accept actions that are terminated by a period.
41659 Historical implementations often allow '\$' in names. A conforming implementation does not
41660 need to support either of these behaviors.

41661 Deciding when to use **%prec** illustrates the difficulty in specifying the behavior of yacc. There
41662 may be situations in which the grammar is not, strictly speaking, in error, and yet yacc cannot
41663 interpret it unambiguously. The resolution of ambiguities in the grammar can in many instances
41664 be resolved by providing additional information, such as using **%type** or **%union** declarations. It
41665 is often easier and it usually yields a smaller parser to take this alternative when it is
41666 appropriate.

41667 The size and execution time of a program produced without the runtime debugging code is
41668 usually smaller and slightly faster in historical implementations.

41669 Statistics messages from several historical implementations include the following types of
41670 information:

41671 *n*/512 terminals, *n*/300 non-terminals
41672 *n*/600 grammar rules, *n*/1500 states
41673 *n* shift/reduce, *n* reduce/reduce conflicts reported

41674 *n*/350 working sets used
41675 Memory: states, etc. *n*/15 000, parser *n*/15 000
41676 *n*/600 distinct lookahead sets
41677 *n* extra closures
41678 *n* shift entries, *n* exceptions
41679 *n* goto entries
41680 *n* entries saved by goto default
41681 Optimizer space used: input *n*/15 000, output *n*/15 000
41682 *n* table entries, *n* zero
41683 Maximum spread: *n*, Maximum offset: *n*

41684 The report of internal tables in the description file is left implementation-defined because all aspects of these limits are also implementation-defined. Some implementations may use dynamic allocation techniques and have no specific limit values to report.

41687 The format of the **y.output** file is not given because specification of the format was not seen to enhance applications portability. The listing is primarily intended to help human users understand and debug the parser; use of **y.output** by a conforming application script would be unusual. Furthermore, implementations have not produced consistent output and no popular format was apparent. The format selected by the implementation should be human-readable, in addition to the requirement that it be a text file.

41693 Standard error reports are not specifically described because they are seldom of use to conforming applications and there was no reason to restrict implementations.

41695 Some implementations recognize "={ " as equivalent to '{ ' because it appears in historical documentation. This construction was recognized and documented as obsolete as long ago as 1978, in the referenced *Yacc: Yet Another Compiler-Compiler*. This volume of IEEE Std 1003.1-2001 chose to leave it as obsolete and omit it.

41699 Multi-byte characters should be recognized by the lexical analyzer and returned as tokens. They should not be returned as multi-byte character literals. The token **error** that is used for error recovery is normally assigned the value 256 in the historical implementation. Thus, the token value 256, which is used in many multi-byte character sets, is not available for use as the value of a user-defined token.

41704 FUTURE DIRECTIONS

41705 None.

41706 SEE ALSO

41707 *c99*, *lex*

41708 CHANGE HISTORY

41709 First released in Issue 2.

41710 Issue 5

41711 The FUTURE DIRECTIONS section is added.

41712 Issue 6

41713 This utility is marked as part of the C-Language Development Utilities option.

41714 Minor changes have been added to align with the IEEE P1003.2b draft standard.

41715 The normative text is reworded to avoid use of the term “must” for application requirements.

41716 IEEE PASC Interpretation 1003.2 #177 is applied, changing the comment on **RCURL** from the }% token to the %}.

41718 NAME

41719 zcat — expand and concatenate data

41720 SYNOPSIS

41721 XSI zcat [*file...*]

41722

41723 DESCRIPTION

41724 The *zcat* utility shall write to standard output the uncompressed form of files that have been
41725 compressed using the *compress* utility. It is the equivalent of *uncompress -c*. Input files are not
41726 affected.

41727 OPTIONS

41728 None.

41729 OPERANDS

41730 The following operand shall be supported:

41731 *file* The pathname of a file previously processed by the *compress* utility. If *file* already
41732 has the **.Z** suffix specified, it is used as submitted. Otherwise, the **.Z** suffix is
41733 appended to the filename prior to processing.

41734 STDIN

41735 The standard input shall be used only if no *file* operands are specified, or if a *file* operand is '**-**'.

41736 INPUT FILES

41737 Input files shall be compressed files that are in the format produced by the *compress* utility.

41738 ENVIRONMENT VARIABLES

41739 The following environment variables shall affect the execution of *zcat*:

41740 *LANG* Provide a default value for the internationalization variables that are unset or null.
41741 (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2,
41742 Internationalization Variables for the precedence of internationalization variables
41743 used to determine the values of locale categories.)

41744 *LC_ALL* If set to a non-empty string value, override the values of all the other
41745 internationalization variables.

41746 *LC_CTYPE* Determine the locale for the interpretation of sequences of bytes of text data as
41747 characters (for example, single-byte as opposed to multi-byte characters in
41748 arguments).

41749 *LC_MESSAGES*

41750 Determine the locale that should be used to affect the format and contents of
41751 diagnostic messages written to standard error.

41752 *NLSPATH* Determine the location of message catalogs for the processing of *LC_MESSAGES*.

41753 ASYNCHRONOUS EVENTS

41754 Default.

41755 STDOUT

41756 The compressed files given as input shall be written on standard output in their uncompressed
41757 form.

41758 STDERR

41759 The standard error shall be used only for diagnostic messages.

41760 OUTPUT FILES

41761 None.

41762 EXTENDED DESCRIPTION

41763 None.

41764 EXIT STATUS

41765 The following exit values shall be returned:

41766 0 Successful completion.

41767 >0 An error occurred.

41768 CONSEQUENCES OF ERRORS

41769 Default.

41770 APPLICATION USAGE

41771 None.

41772 EXAMPLES

41773 None.

41774 RATIONALE

41775 None.

41776 FUTURE DIRECTIONS

41777 None.

41778 SEE ALSO

41779 *compress, uncompress*

41780 CHANGE HISTORY

41781 First released in Issue 4.

Index

<control>-V.....	368	string functions.....	167
<control>-W	368	user-defined functions	169
_CFLAGS.....	215	variables and special variables.....	160
_LDFLAGS.....	216	background work	
_LIBS	216	at	144
_POSIX_VDISABLE	893	batch.....	190
Account_Name	108	bg	208
actions equivalent to functions.....	7	crontab.....	275
admin	126	fg	442
ADV	9	jobs.....	521
AIO	9	nice	664
alias	28, 48, 131	nohup.....	675
alias substitution.....	32	renice.....	820
AND lists.....	51	BAR.....	9
AND-OR list	50	basename.....	187
appending redirected output	44	batch.....	190
ar	134	batch administration.....	104
archives		batch authorization.....	103
ar command.....	134	batch client-server interaction	101
ARG_MAX.....	1062	batch environment	
arithmetic expansion	41	services	101
arithmetic language		utilities	101
bc.....	193	Batch Job Abort.....	103, 114
arithmetic precision and operations	7	batch job creation.....	102
array identifiers.....	198	Batch Job Execution	103, 106
asa	141	Batch Job Exit	103, 113
asynchronous lists	50	batch job identifier.....	122
at	144	Batch Job Message Request	116
at-job.....	144	Batch Job Routing.....	102, 113
automatic storage class	202	batch job states	105
awk	153	Batch Job Status Request	117
actions.....	164	batch job tracking	102
arithmetic functions.....	166	batch notification	104
escape sequences.....	162	batch queue.....	101
expression patterns.....	164	Batch Queue Status Request	119
expressions	156	Batch Server Restart	114
functions.....	166	batch services	104
grammar.....	170	bc.....	193
input/output and general functions	168	grammar	194
lexical conventions.....	176	lexical conventions.....	196
output statements	165	operations	198
overall program structure	155	operators	198
pattern ranges	164	bcc (mailer blind carbon copy)	608
patterns.....	163	BC_BASE_MAX	18
regular expressions	161	BC_DIM_MAX	18
special patterns	163	BC_SCALE_MAX	18

BC_STRING_MAX.....	18, 196	write	1055
BE.....	10	compilers	
bg.....	28, 48, 208	c99.....	211
binary primaries.....	910	fort77	464
break.....	65	yacc.....	1064
built-in utilities.....	28	compound commands.....	52
builtin.....	263	compound-list	50
c99	211	compress.....	264
external symbols.....	215	compression	
standard libraries.....	214	compress	264
cal.....	220	uncompress	952
can.....	1	zcat	1080
carriage-control characters	141	concurrent execution of processes	3
case conditional construct	53	configuration values	484
cat.....	222	conforming application.....	139
cc (mailer carbon copy)	608	consequences of shell errors.....	46
CD.....	10	continue	69
cd	28, 48, 226	control characters	890
cflow	231	controlling terminal.....	3
changing the current working directory.....	7	Coordinated Universal Time (UTC).....	300
character counting.....	1046	copy files commands	
charmap		cp.....	267
with localedef.....	558	dd	302
writing names with locale	553	ln	549
charmap file.....	557, 893	mv	655
Checkpoint.....	108	pax	700
checksums		cp.....	267
cksum.....	248	cpio format.....	720
chgrp	234	CPT	10
chmod	237	CPU time	918
grammar.....	240	cron daemon.....	278
chown.....	244	crontab	275
cksum.....	248	CS	10
cmp.....	253	csplit	279
codeset conversion.....	506	ctags	283
tr	927	current working directory.....	3
COLL_WEIGHTS_MAX	18	cut	288
colon	67	CX	10
comm.....	256	cxref	292
command	28, 48, 259	data keywords	748
command mode	336	date	295
command search and execution.....	48	conversion specifications.....	295
command substitution	40	modified conversion specifications	296
communications commands		dd	302
mailx.....	586	default queue.....	118
talk.....	901	deferred batch services	106
uucp.....	966	Delete Batch Job Request	115
uudecode.....	970	delta	311
uuencode.....	973	destination	123
uustat	978	df	315
ux.....	981	diff.....	319

binary output format	321	regular expressions	337
default output format	321	shell escape command	346
directory comparison format	320	substitute command	344
-c or -C output format	322	undo command	345
-e output format	322	write command	346
-f output format	322	edit buffer	355, 988
directory commands		edit line	855
cd	226	editors	
pwd	759	ed	336
directory lister	571	ex	355
dirname	326	sed	842
disk space commands		vi	988
df	315	ED_FILE_MAX	348
du	329	ED_LINE_MAX	348
ulimit	941	effective group ID	3
documentation	631	effective user ID	3
dot	71	Eighth Edition UNIX	263
double-quotes	30	env	352
du	329	EPERM	272
duplicating an input file descriptor	45	Error_Path	109
duplicating an output file descriptor	45	escape character (backslash)	30
echo	333	escape sequences	
ed	336	awk	162
addresses	338	gencat	474
append command	340	lex	540
change command	340	establish the locale	7
commands	339	eval	73
copy command	345	ex	355
delete command	341	<backslash>	367
edit command	341	<control>-D command	390
edit without checking command	341	<newline>	367
filename command	341	abbreviate command	370
global command	341	addressing	361
global non-matched command	346	adjust window command	388
help command	342	append command	371
help-mode command	342	args command	371
insert command	342	autoindent option	392
interactive global command	342	autoprint option	393
interactive global not-matched command	346	autowrite option	393
join command	343	beautify option	393
line number command	346	change command	371
list command	343	chdir command	372
mark command	343	command descriptions	368
move command	343	copy command	372
null command	347	delete command	372
number command	343	directory option	393
print command	344	edcompatible option	393
prompt command	344	edit command	372
quit command	344	edit options	392
quit without checking command	344	errorbells option	394
read command	344	escape command	389

execute command	391
exrc command.....	394
file command.....	373
global command.....	374
ignorecase option	394
initialization.....	358
input editing.....	366
insert command.....	375
join command	375
list command	376, 394
magic command.....	394
map command.....	376
mark command.....	377
mesg command.....	394
move command.....	378
next command	378
number command.....	379
number option	395
open command	379
paragraphs option.....	395
preserve	355
preserve command	379
print command	380
prompt command	395
put command.....	380
quit command	380
read command	381
readonly command.....	395
recover command.....	381
redraw command.....	396
regular expressions	391
remap command	396
replacement strings.....	391
report command.....	396
rewind command	382
scroll command	366, 396
sections command.....	396
set command	382
shell command.....	382
shell option	397
shift left command	390
shift right command	390
shiftwidth option.....	397
showmatch option	397
showmode command	397
slowopen command	397
source command	383
substitute command	383
suspend command.....	384
tabstop option	397
tag command.....	384
taglength option	397
tags command.....	398
term command.....	398
terse command.....	398
unabbrev command.....	385
undo command.....	385
unmap command	385
version command.....	385
visual command	386
warn command.....	398
window command.....	398
wrapmargin option.....	399
wrapscan option.....	399
write command.....	386
write line number command	390
writeany option	399
xit command.....	387
yank command	387
exec.....	75, 676
exec family	263, 662, 1062
exec, family	28
Execution_Time	110
EXINIT	355
exit	77
exit status and errors.....	46
exit status for commands.....	46
expand	426
export	79
expr.....	429
matching expression.....	431
string operand.....	431
expression argument	165
expression list.....	165
EXPR_NEST_MAX.....	18
extended regular expression	153, 161, 271
.....455, 495, 539, 657, 710, 825, 1060	
extension	
CX	10
OH	13
XSL.....	17
false	28, 48, 434
fc	28, 48, 436
FD.....	10
fg	28, 48, 442
field splitting.....	42
FIFO special files.....	641
file	444
file access permissions.....	4
file comparisons	
cmp	253
comm	256

diff.....	319	more	643
uniq.....	960	nl	667
file contents	6	paste	686
file conversion		pax	700
cut	288	pr	737
dd	302	read	817
expand	426	sed	842
fold	461	tail	898
head	503	tee	905
join	525	tr	927
od	678	uncompress	952
paste	686	unexpand	955
patch	690	zcat	1080
sort	871	find.....	452
strings	880	fold.....	461
tail	898	for loop.....	52
tr	927	fort77	464
tsort	935	external symbols.....	467
unexpand	955	standard libraries.....	466
uniq.....	960	FR.....	10
uudecode	970	FSC.....	10
uuencode	973	function definition command	54
file creation	4	function identifiers	198
file descriptor.....	3, 43	fuser	470
file mode creation mask	3	g-file	311
file permission commands		gencat	473
chgrp	234	escape sequences	474
chmod	237	generated file	311
chown	244	get	476
umask	943	getconf	484
file read	4	getopts	28, 48, 490
file removal	6	global storage class	202
file searching		GNU make	627
grep.....	495	grep	495
file time values	6	grouping commands	52
file tree commands		hash	500
diff.....	319	head	503
find.....	452	here-document	44
ls	571	history command	
mkdir	638	fc	436
rmdir	831	Hold Batch Job Request.....	116
file write	4	Hold_Types.....	110
filters		HOME.....	372
asa.....	141	hunk	692
awk	153	iconv	506
compress	264	id	509
dd	302	if conditional construct.....	53
expand	426	implementation-defined	1
fold	461	inference rule	610
head	503	input field separator	853
iconv	506	input mode	336

IP6	11
ipcrm	513
ipcs	515
I_ISVTX.....	239
jobs	28, 48, 521
Job_Owner	110
join	525
Join_Path	110
Keep_Files	110
keyword-value pairs	123
kill.....	28, 48, 530
legacy.....	1
lex.....	535
actions	541
definitions	537
escape sequences	540
regular expressions	539
rules	538
table sizes	538
user subroutines	539
lex, translation table	545
libraries	
ar command.....	134
LIMIT	17
line counting.....	1046
LINE_MAX.....	18, 154, 348, 356, 607, 850, 963
link.....	547
lists	50
AND-OR.....	50
compound-list	50
ln	549
locale	553
localedef.....	558
Locate Batch Job Request.....	117
locking file.....	241
logger	562
logname	564
lp	566
LR(1) grammars.....	1078
ls.....	571
m4	579
macro processor.....	579
magic file	449
mailx.....	586
change current directory	597
change folder	598
command escapes	604
commands	596
copy messages	597
declare aliases	596
declare alternatives	597
delete aliases.....	603
delete messages	597
delete messages and display	598
direct messages to mbox	600
discard header fields.....	597
display beginning of messages.....	603
display current message number	604
display header summary	599
display list of folders	599
display message	601
display message size	602
echo a string.....	598
edit message	598, 603
execute commands conditionally	600
exit	598
follow up specified messages	599
help	599
hold messages	599
internal variables	593
invoke a shell	602
invoke shell command	604
list available commands	600
mail a message	600
null command	604
pipe message	600
process next specified message	600
quit	601
read mailx commands from a file	602
receive mode	586
reply to a message	601
reply to a message list	601
retain header fields	602
save messages	602
scroll header display	604
send mode	586
set variables	602
start-up	593
touch messages	603
undelete messages	603
unset variables	603
write messages to a file	603
Mail_Points	111
Mail_Users	111
make	610
default rules	620
inference rules	617
internal macros	619
libraries	618
macros	616
makefile execution	614
makefile syntax	613

target rules	614	MX.....	12
make, GNU version	627	NAME_MAX.....	139, 727
man.....	631	newgrp	28, 48, 660
mathematical functions.....	9	NGROUPS_MAX.....	663
may	2	nice.....	664
MC1	11	Ninth Edition UNIX	205, 335, 746
MC2	11	nl	667
MC3	11	nm	671
mesg	635	noclobber option.....	699
message catalog generation	473	nohup	675
MF.....	11	non-printable	348, 848, 904
MIL-STD-1753	468	OB	12
Minimum_Cpu_Interval.....	108	object files.....	671
mkdir.....	638	od	678
mkfifo	641	OF.....	12
ML.....	11	OH	13
MLR	12	open file descriptors for reading and writing.....	46
Modify Batch Job Request	117	open mode	355
MON.....	12	option ADV.....	9
more.....	643	AIO	9
discard and refresh.....	649	BAR.....	9
display position	652	BE	10
examine new file.....	651	CD	10
examine next file.....	651	CPT	10
examine previous file	651	CS	10
go to beginning of file.....	649	FD	10
go to end-of-file.....	649	FR	10
go to tag.....	651	FSC	10
help	648	IP6	11
invoke editor	651	MC1	11
mark position	650	MC2	11
quit.....	652	MC3	11
refresh the screen.....	649	MF	11
repeat search.....	650	ML	11
repeat search in reverse.....	651	MLR	12
return to mark	650	MON	12
return to previous position	650	MPR	12
scroll backward one half screenful	649	MSG	12
scroll backward one line	648	MX	12
scroll backward one screenful	648	PIO	13
scroll forward one half screenful	649	PS	13
scroll forward one line	648	RS	13
scroll forward one screenful	648	RTS	13
search backward for pattern	650	SD	13
search forward for pattern	650	SEM	13
skip forward one line.....	649	SHM	13
motion command	857	SIO	14
Move Batch Job Request	118	SPI	14
MPR	12	SPN	14
MSG	12	SS.....	14
mv	655		

TCT	14
TEF	14
THR	14
TMO	15
TMR	15
TPI	15
TPP	15
TPS	15
TRC	15
TRI	15
TRL	15
TSA	16
TSF	16
TSH	16
TSP	16
TSS	16
TYM	16
UP	16
XSR	17
OR lists	51
ordinary identifiers	198
Output_Path	111
paginators	
more	643
parameter expansion	37
parameters and variables	33
paste	686
patch	690
filename determination	693
patch application	693
patch file format	692
pathchk	696
pathname expansion	42
pathname manipulation	
basename	187
dirname	326
pathchk	696
pathname resolution	7
PATH_MAX	18, 733, 827
pattern matching	452, 709, 967, 983
definition	62
in case statements	53
in shell variables	39
pattern matching notation	62, 458, 726
pattern scanning and processing language	
at	153
patterns matching a single character	62
patterns matching multiple characters	63
patterns used for filename expansion	63
pax	700
archive character set encoding/decoding	731
cpio file data	722
cpio filename	722
cpio header	721
cpio interchange format	720
cpio special entries	723
extended header	713
extended header file times	716
extended header keyword precedence	716
list mode format specifications	708
ustar format	717
ustar interchange format	717
PIO	13
pipelines	49
portable character set	616
positional parameters	33
POSIX2_BC_BASE_MAX	17-18
POSIX2_BC_DIM_MAX	17-18
POSIX2_BC_SCALE_MAX	17-18
POSIX2_BC_STRING_MAX	17-18
POSIX2_COLL_WEIGHTS_MAX	17, 19
POSIX2_EXPR_NEST_MAX	17, 19
POSIX2_LINE_MAX	17, 19
POSIX2_RE_DUP_MAX	17, 19
POSIX2_SYMLINKS	19
pr	737
print-related commands	
fold	461
lp	566
pr	737
printf	742
Priority	112
privileges	636, 666
process attributes	3
process group ID	3
process ID	3
process status report	752
prs	747
PS	13
ps	752
public locale	553
pwd	28, 48, 759
qalter	761
qdel	770
qhold	773
qmove	776
qmsg	779
qrerun	782
qrsls	784
qselect	787
qsig	796
qstat	799

qsub	804	addresses.....	844
Queue Batch Job Request.....	118	editing commands.....	844
quote removal	42	regular expressions	844
quoting.....	30	Select Batch Jobs Request.....	120
read.....	28, 48, 817	SEM.....	13
readonly.....	82	sequential lists	51
real group ID.....	3	Server Shutdown Request	120
real user ID.....	3	Server Status Request	121
redirecting input	44	session membership.....	3
redirecting output.....	44	set.....	86
redirection	43	set-group-ID	3, 273
regular expressions	161, 271, 337, 361	set-user-ID	3, 242, 273
.....	391, 431, 455, 495, 539, 647, 657	set-user-ID scripts	865
.....	667, 707, 825, 844, 1005, 1008, 1060	sh.....	851
related to shell patterns.....	62	command history list.....	855
relational database operator.....	525	command line editing.....	855
Release Batch Job Request	119	vi line editing command mode	857
remove directories.....	831	vi line editing insert mode.....	856
remove files.....	824	vi-mode command line editing.....	855
renice	820	shall.....	2
requested batch services	115	shell command language	29
Rerun Batch Job Request.....	120	alias substitution	32
Rerunable	112	appending redirected output.....	44
reserved words.....	33	arithmetic expansion	41
Resource_List	112	command substitution	40
return.....	84	compound commands.....	52
RE_DUP_MAX.....	18	consequences of shell errors.....	46
rm.....	824	double-quotes.....	30
rmdel	829	duplicating an input file descriptor	45
rmdir.....	831	duplicating an output file descriptor	45
root directory.....	3	escape character (backslash)	30
RS	13	exit status and errors	46
RTS.....	13	exit status for commands.....	46
sact.....	834	field splitting	42
saved set-group-ID.....	3	function definition command	54
saved set-user-ID	3	grammar	55
sccs.....	837	here-document	44
SCCS commands		introduction.....	29
admin	126	lists.....	50
delta.....	311	open file descriptors for reading and writing.....	46
get	476	parameter expansion	37
prs	747	parameters and variables.....	33
rmdel.....	829	pathname expansion.....	42
sact.....	834	pattern matching notation	62
sccs.....	837	patterns matching a single character	62
unget	958	patterns matching multiple characters	63
val	985	patterns used for filename expansion	63
what	1049	pipelines	49
SD.....	13	positional parameters	33
search pattern.....	283	quote removal	42
sed.....	842	quoting	30

redirecting input	44
redirecting output	44
redirection.....	43
reserved words	33
shell commands	47
shell execution environment.....	61
shell grammar lexical conventions	55
shell grammar rules	56
shell variables.....	34
signals and error handling.....	61
simple commands	47
single-quotes	30
special built-in utilities.....	64
special parameters.....	34
tilde expansion.....	37
token recognition.....	31
word expansions	36
shell commands	47
shell execution environment.....	61, 132, 819, 945
shell grammar.....	55
shell grammar lexical conventions	55
shell grammar rules	56
shell introduction.....	29
shell variables.....	34
Shell_Path_List.....	112
shift.....	92
SHM.....	13
should.....	2
SIGCONT.....	358
SIGHUP	337, 358, 988, 1029
SIGINT	313, 337, 357, 1041
Signal Batch Job Request.....	121
signal processes	530
signals and error handling.....	61
SIGQUIT	337
SIGTERM.....	358
simple commands	47
single-quotes	30
SIO	14
sleep.....	868
SLR(1) grammars.....	1078
sort.....	871
special built-in.....	263, 666, 677, 760, 865, 918, 934
special built-in utilities.....	64
break.....	65, 69
characteristics.....	64
colon.....	67
dot.....	71
eval	73
exec.....	75
exit	77
export	79
readonly.....	82
return.....	84
set	86
shift.....	92
times	94
trap	96
unset	99
special parameters.....	34
special targets.....	615
SPI.....	14
split	877
split files	
csplit	279
split	877
SPN.....	14
spoofing	83
SS.....	14
standard error.....	43
standard input.....	43
standard output	43
strings.....	880
strip.....	883
stty	885
combination modes	890
control modes.....	885
input modes.....	886
local modes.....	888
output modes	887
special control character assignments	889
st_gid.....	139
st_mode	139
st_mtime	139
st_size	139
st_uid.....	139
superuser	440, 577, 725
supplementary group IDs	3
system configuration values	484
system name	949
tabs.....	894
tag file creation.....	283
tail	898
talk	901
tar format.....	717
target queue	118
target rule	610
TCT	14
tee	905
TEF.....	14
terminal characteristics	
stty	885

tabs	894	user identity	
tput	924	id	509
tty	937	logname	564
terminate processes.....	530	newgrp	660
terminology.....	1	who	1051
test.....	908	User_List.....	113
THR	14	ustar format	717
tilde expansion.....	37	utility option parsing.....	490
time.....	916	uucp	966
times	94	uudecode	970
TMO	15	uuencode	973
TMPDIR.....	704	uustat	978
TMR	15	uux	981
token recognition.....	31	val	985
touch.....	920	Variable_List.....	113
TPI.....	15	vi	988
TPP.....	15	<ESC>	1028
TPS.....	15	append	1009
tput	924	change	1010
tr.....	927	change to end-of-line.....	1010
Track Batch Job Request.....	121	clear and redisplay	996
trap.....	96	command descriptions.....	989
TRC.....	15	control-D	1025
TRI	15	control-H	1025
TRL	15	control-T	1027
trojan horse	577	control-U	1027
true	28, 48, 933	control-V	1027
TSA	16	current and line above.....	1003
TSF	16	delete	1011
TSH.....	16	delete character.....	1020-1021
tsort.....	935	delete to end-of-line.....	1011
TSP	16	display information	995
TSS	16	edit the alternate file	997
tty	937	enter ex mode.....	1017
TYM	16	execute	1008
type	939	execute an ex command.....	1007
ulimit	941	exit	1023
umask	28, 48, 943	find character	1012
unalias	28, 48, 947	find regular expression	1005
uname	949	Initialization	989
unary primaries.....	910	input mode commands	1023
uncompress.....	952	insert	1013-1014
undefined.....	2	insert empty line	1015-1016
unexpand.....	955	join	1014
unget.....	958	mark position	1014
uniq.....	960	move back	1003, 1009
unlink	964	move cursor	995, 998-999, 1018
unset	99	move down	996
unspecified.....	2	move forward	1003
until loop	54	move to bigword.....	1012, 1020
UP.....	16	move to bottom of screen.....	1014

move to first character in line	1006
move to first non-<blank>	1002
move to line	1013
move to matching character	999
move to middle of screen	1015
move to next section	1002
move to specific column	1004
move to top of screen	1013
move to word	1012, 1019
move up	996
newline	1026
nul	1025, 1028
page backwards	994
page forward	995
put from buffer	1016
redraw screen	997
redraw window	1022
regular expression	1008
repeat	1004
repeat find	1007, 1015
repeat substitution	1000
replace character	1017-1018
replace text with command	998
return to previous context	1000-1001
return to previous section	1001
reverse case	1008
reverse find character	1004
scroll backward	997
scroll backward by line	997
scroll forward	994
scroll forward by line	994
search for tagstring	998
shift left	1007
shift right	1007
substitute character	1018
substitute lines	1018
terminate command or input mode	997
undo	1019
undo current line	1019
yank	1021
yank current line	1021
visual mode	355
wait	28, 48, 1042
warning	
OB	12
OF	12
wc	1046
what	1049
while loop	54
who	1051
word counting	1046
word expansions	36
write	1055
xargs	1058
XSI	17
XSR	17
yacc	1064
algorithms	1075
code file	1066
completing the program	1075
conflicts	1073
debugging the parser	1075
declarations section	1067
description file	1066
error handling	1073
grammar rules	1069
header file	1066
input grammar	1071
input language	1066
interface to the lexical analyzer	1074
lexical structure of the grammar	1067
library	1075
limits	1076
programs section	1070
zcat	1080