

INTERNATIONAL  
STANDARD

**ISO/IEC**  
**9945-1**

**IEEE**  
**Std 1003.1**

First edition  
1990-12-07

---

---

**Information technology — Portable Operating  
System Interface (POSIX) —**

**Part 1 :**  
**System Application Program Interface (API)**  
**[C Language]**

**(standards.iteh.ai)**

*Technologies de l'information — Interface pour la portabilité des systèmes (POSIX) —*

*Partie 1 : Interface programme de systèmes d'application (API) [Langage C]*

<https://standards.iteh.ai/catalog/standards/sist/1003-1-1990/iso-iec-9945-1-1990>  
7c5612b5b049/iso-iec-9945-1-1990



Reference number  
ISO/IEC 9945-1 : 1990 (E)  
IEEE Std 1003.1-1990

## iTeh STANDARD PREVIEW (standards.iteh.ai)

ISO/IEC 9945-1:1990 ISBN 1-55937-061-0

<https://standards.iteh.ai/catalog/standards/sist/febcb600-97e1-480a-9304-7c5612b5b049/iso-iec-9945-1-1990>  
Library of Congress Catalog Number 90-084554

Quote in 8.1.2.3 on Returns is taken from X3.159-1989,  
developed under the auspices of the American National Standards  
Accredited Committee X3 Technical Committee X3J11, Computer and  
Business Equipment Manufacturers Association (CBEMA),  
311 First St, N.W., Suite 500, Washington, DC 20001.

©Copyright 1990 by

**The Institute of Electrical and Electronics Engineers, Inc.**  
**345 East 47th Street, New York, NY 10017, USA**

*No part of this publication may be reproduced in any form,  
in an electronic retrieval system or otherwise,  
without the prior written permission of the publisher.*

International Standard ISO/IEC 9945-1: 1990

IEEE Std 1003.1-1990

(Revision of IEEE Std 1003.1-1988)

**Information technology—Portable  
Operating System Interface (POSIX)  
Part 1:  
System Application Program Interface  
(API) [C Language]**

Sponsor

**Technical Committee on Operating Systems  
and Application Environments  
of the  
IEEE Computer Society**

Approved September 28, 1990

IEEE Standards Board

Approved 1990 by the  
**International Organization for Standardization**

and by the

**International Electrotechnical Commission**

**Abstract:** ISO/IEC 9945-1: 1990 (IEEE Std 1003.1-1990), *Information technology—Portable Operating System Interface (POSIX)—Part 1: System Application Program Interface (API) [C Language]* is part of the POSIX series of standards for applications and user interfaces to open systems. It defines the applications interface to basic system services for input/output, file system access, and process management. It also defines a format for data interchange. This standard is stated in terms of its C binding.

**Keywords:** API, application portability, C (programming language), data processing, information interchange, open systems, operating system, portable application, POSIX, programming language, system configuration computer interface



Adopted as an International Standard by the  
International Organization for Standardization  
and by the  
International Electrotechnical Commission



Published by  
The Institute of Electrical and Electronics Engineers, Inc.

**iTeh STANDARD PREVIEW**  
**(standards.iteh.ai)**

ISO/IEC 9945-1:1990

<https://standards.iteh.ai/catalog/standards/sist/fbcb600-97e1-480a-9304-7c5612b5b049/iso-iec-9945-1-1990>

**IEEE Standards** documents are developed within the Technical Committees of the IEEE Societies and the Standards Coordinating Committees of the IEEE Standards Board. Members of the committees serve voluntarily and without compensation. They are not necessarily members of the Institute. The standards developed within IEEE represent a consensus of the broad expertise on the subject within the Institute as well as those activities outside of IEEE that have expressed an interest in participating in the development of the standard.

Use of an IEEE Standard is wholly voluntary. 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.

Comments for revision of IEEE Standards are welcome from any interested party, regardless of membership affiliation with IEEE. Suggestions for changes in documents should be in the form of a proposed change of text, together with appropriate supporting comments. <https://standards.ieee.org/catalog/standards/sist/fbcb600-97e1-480a-9304-c924b0578c1e/iec-9945-1-1990>

**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 all concerned interests, it is important to ensure that any interpretation has also received the concurrence of a balance of interests. For this reason, the IEEE and the members of its technical committees are not able to provide an instant response to interpretation requests except in those cases where the matter has previously received formal consideration.

Comments on standards and requests for interpretations should be addressed to:

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

IEEE Standards documents are adopted by the Institute of Electrical and Electronics Engineers without regard to whether their adoption may involve patents on articles, materials, or processes. Such adoption does not assume any liability to any patent owner, nor does it assume any obligation whatever to parties adopting the standards documents.

# Contents

	PAGE
Foreword . . . . .	viii
Introduction . . . . .	ix
Section 1: General . . . . .	1
1.1 Scope . . . . .	1
1.2 Normative References . . . . .	2
1.3 Conformance . . . . .	2
Section 2: Terminology and General Requirements . . . . .	9
2.1 Conventions . . . . .	9
2.2 Definitions . . . . .	10
2.3 General Concepts . . . . .	21
2.4 Error Numbers . . . . .	23
2.5 Primitive System Data Types . . . . .	27
2.6 Environment Description . . . . .	27
2.7 C Language Definitions . . . . .	29
2.8 Numerical Limits . . . . .	34
2.9 Symbolic Constants . . . . .	37
Section 3: Process Primitives . . . . .	41
3.1 Process Creation and Execution . . . . .	41
3.1.1 Process Creation . . . . .	41
3.1.2 Execute a File . . . . .	42
3.2 Process Termination . . . . .	46
3.2.1 Wait for Process Termination . . . . .	47
3.2.2 Terminate a Process . . . . .	49
3.3 Signals . . . . .	51
3.3.1 Signal Concepts . . . . .	51
3.3.2 Send a Signal to a Process . . . . .	56
3.3.3 Manipulate Signal Sets . . . . .	57
3.3.4 Examine and Change Signal Action . . . . .	58
3.3.5 Examine and Change Blocked Signals . . . . .	60
3.3.6 Examine Pending Signals . . . . .	62
3.3.7 Wait for a Signal . . . . .	62
3.4 Timer Operations . . . . .	63
3.4.1 Schedule Alarm . . . . .	63
3.4.2 Suspend Process Execution . . . . .	64
3.4.3 Delay Process Execution . . . . .	65
Section 4: Process Environment . . . . .	67
4.1 Process Identification . . . . .	67
4.1.1 Get Process and Parent Process IDs . . . . .	67

	PAGE
4.2 User Identification . . . . .	68
4.2.1 Get Real User, Effective User, Real Group, and Effective Group IDs . . . . .	68
4.2.2 Set User and Group IDs . . . . .	68
4.2.3 Get Supplementary Group IDs . . . . .	70
4.2.4 Get User Name . . . . .	71
4.3 Process Groups . . . . .	72
4.3.1 Get Process Group ID . . . . .	72
4.3.2 Create Session and Set Process Group ID . . . . .	72
4.3.3 Set Process Group ID for Job Control . . . . .	73
4.4 System Identification . . . . .	74
4.4.1 Get System Name . . . . .	74
4.5 Time . . . . .	75
4.5.1 Get System Time . . . . .	75
4.5.2 Get Process Times . . . . .	76
4.6 Environment Variables . . . . .	77
4.6.1 Environment Access . . . . .	77
4.7 Terminal Identification . . . . .	78
4.7.1 Generate Terminal Pathname . . . . .	78
4.7.2 Determine Terminal Device Name . . . . .	79
4.8 Configurable System Variables . . . . .	80
4.8.1 Get Configurable System Variables . . . . .	80
Section 5: Files and Directories . . . . .	83
5.1 Directories . . . . .	83
5.1.1 Format of Directory Entries . . . . .	83
5.1.2 Directory Operations . . . . .	83
5.2 Working Directory . . . . .	86
5.2.1 Change Current Working Directory . . . . .	86
5.2.2 Get Working Directory Pathname . . . . .	87
5.3 General File Creation . . . . .	88
5.3.1 Open a File . . . . .	88
5.3.2 Create a New File or Rewrite an Existing One . . . . .	91
5.3.3 Set File Creation Mask . . . . .	91
5.3.4 Link to a File . . . . .	92
5.4 Special File Creation . . . . .	94
5.4.1 Make a Directory . . . . .	94
5.4.2 Make a FIFO Special File . . . . .	95
5.5 File Removal . . . . .	96
5.5.1 Remove Directory Entries . . . . .	96
5.5.2 Remove a Directory . . . . .	98
5.5.3 Rename a File . . . . .	99
5.6 File Characteristics . . . . .	101
5.6.1 File Characteristics: Header and Data Structure . . . . .	101
5.6.2 Get File Status . . . . .	103
5.6.3 Check File Accessibility . . . . .	104
5.6.4 Change File Modes . . . . .	106
5.6.5 Change Owner and Group of a File . . . . .	107

	PAGE
5.6.6 Set File Access and Modification Times . . . . .	108
5.7 Configurable Pathname Variables . . . . .	110
5.7.1 Get Configurable Pathname Variables . . . . .	110
Section 6: Input and Output Primitives . . . . .	113
6.1 Pipes . . . . .	113
6.1.1 Create an Inter-Process Channel . . . . .	113
6.2 File Descriptor Manipulation . . . . .	114
6.2.1 Duplicate an Open File Descriptor . . . . .	114
6.3 File Descriptor Deassignment . . . . .	115
6.3.1 Close a File . . . . .	115
6.4 Input and Output . . . . .	116
6.4.1 Read from a File . . . . .	116
6.4.2 Write to a File . . . . .	118
6.5 Control Operations on Files . . . . .	121
6.5.1 Data Definitions for File Control Operations . . . . .	121
6.5.2 File Control . . . . .	121
6.5.3 Reposition Read/Write File Offset . . . . .	127
Section 7: Device- and Class-Specific Functions . . . . .	129
7.1 General Terminal Interface . . . . .	129
7.1.1 Interface Characteristics . . . . .	129
7.1.1.1 Opening a Terminal Device File . . . . .	129
7.1.1.2 Process Groups . . . . .	129
7.1.1.3 The Controlling Terminal . . . . .	130
7.1.1.4 Terminal Access Control . . . . .	130
7.1.1.5 Input Processing and Reading Data . . . . .	131
7.1.1.6 Canonical Mode Input Processing . . . . .	132
7.1.1.7 Noncanonical Mode Input Processing . . . . .	132
7.1.1.8 Writing Data and Output Processing . . . . .	133
7.1.1.9 Special Characters . . . . .	133
7.1.1.10 Modem Disconnect . . . . .	135
7.1.1.11 Closing a Terminal Device File . . . . .	135
7.1.2 Parameters That Can Be Set . . . . .	135
7.1.2.1 <i>termios</i> Structure . . . . .	135
7.1.2.2 Input Modes . . . . .	136
7.1.2.3 Output Modes . . . . .	137
7.1.2.4 Control Modes . . . . .	138
7.1.2.5 Local Modes . . . . .	139
7.1.2.6 Special Control Characters . . . . .	140
7.1.2.7 Baud Rate Values . . . . .	141
7.1.3 Baud Rate Functions . . . . .	141
7.1.3.1 Synopsis . . . . .	141
7.1.3.2 Description . . . . .	142
7.1.3.3 Returns . . . . .	142
7.1.3.4 Errors . . . . .	142
7.1.3.5 Cross-References . . . . .	142
7.2 General Terminal Interface Control Functions . . . . .	143
7.2.1 Get and Set State . . . . .	143



	PAGE
7.2.2 Line Control Functions . . . . .	145
7.2.3 Get Foreground Process Group ID . . . . .	147
7.2.4 Set Foreground Process Group ID . . . . .	148
 Section 8: Language-Specific Services for the C Programming Language . . . . .	
8.1 Referenced C Language Routines . . . . .	151
8.1.1 Extensions to Time Functions . . . . .	152
8.1.2 Extensions to <i>setlocale()</i> Function . . . . .	154
8.2 C Language Input/Output Functions . . . . .	155
8.2.1 Map a Stream Pointer to a File Descriptor . . . . .	156
8.2.2 Open a Stream on a File Descriptor . . . . .	157
8.2.3 Interactions of Other <i>FILE</i> -Type C Functions . . . . .	158
8.2.4 Operations on Files — the <i>remove()</i> Function . . . . .	162
8.3 Other C Language Functions . . . . .	162
8.3.1 Nonlocal Jumps . . . . .	162
8.3.2 Set Time Zone . . . . .	162
 Section 9: System Databases . . . . .	
9.1 System Databases . . . . .	165
9.2 Database Access . . . . .	166
9.2.1 Group Database Access . . . . .	166
9.2.2 User Database Access . . . . .	167
 Section 10: Data Interchange Format . . . . .	
10.1 Archive/Interchange File Format . . . . .	169
10.1.1 Extended <i>tar</i> Format . . . . .	169
10.1.2 Extended <i>cpio</i> Format . . . . .	173
10.1.3 Multiple Volumes . . . . .	177
 Annex A (informative) Bibliography . . . . .	
A.1 Related Open Systems Standards . . . . .	179
A.2 Other Standards . . . . .	181
A.3 Historical Documentation and Introductory Texts . . . . .	182
 Annex B (informative) Rationale and Notes . . . . .	
B.1 Scope and Normative References . . . . .	185
B.2 Definitions and General Requirements . . . . .	196
B.3 Process Primitives . . . . .	226
B.4 Process Environment . . . . .	246
B.5 Files and Directories . . . . .	253
B.6 Input and Output Primitives . . . . .	264
B.7 Device- and Class-Specific Functions . . . . .	273
B.8 Language-Specific Services for the C Programming Language . . . . .	283
B.9 System Databases . . . . .	293
B.10 Data Interchange Format . . . . .	294
 Annex C (informative) Header Contents Samples . . . . .	
	301

	PAGE
Annex D (informative) Profiles . . . . .	313
D.1 Definitions . . . . .	313
D.2 Options in This Part of ISO/IEC 9945 . . . . .	314
D.3 Related Standards . . . . .	315
D.4 Related Activities . . . . .	315
D.5 Relationship to IEEE Draft Project 1003.0 . . . . .	315
Annex E (informative) Sample National Profile . . . . .	317
E.1 (Example) Profile for Denmark . . . . .	318
Identifier Index . . . . .	321
Alphabetic Topical Index . . . . .	327

TABLES

Table 2-1 – Primitive System Data Types . . . . .	27
Table 2-2 – Reserved Header Symbols . . . . .	31
Table 2-3 – Minimum Values . . . . .	35
Table 2-4 – Run-Time Inceasable Values . . . . .	35
Table 2-5 – Run-Time Invariant Values (Possibly Indeterminate) . . . . .	36
Table 2-6 – Pathname Variable Values . . . . .	36
Table 2-7 – Invariant Value . . . . .	37
Table 2-8 – Symbolic Constants for the <i>access()</i> Function . . . . .	38
Table 2-9 – Symbolic Constants for the <i>lseek()</i> Function . . . . .	38
Table 2-10 – Compile-Time Symbolic Constants . . . . .	38
Table 2-11 – Execution-Time Symbolic Constants . . . . .	39
Table 3-1 – Required Signals . . . . .	52
Table 3-2 – Job Control Signals . . . . .	52
Table 4-1 – <i>uname()</i> Structure Members . . . . .	75
Table 4-2 – Configurable System Variables . . . . .	80

Table 5-1	- <i>stat</i> Structure . . . . .	101
Table 5-2	- Configurable Pathname Variables . . . . .	111
Table 6-1	- <i>cmd</i> Values for <i>fcntl()</i> . . . . .	122
Table 6-2	- File Descriptor Flags Used for <i>fcntl()</i> . . . . .	122
Table 6-3	- <i>l_type</i> Values for Record Locking With <i>fcntl()</i> . . . . .	122
Table 6-4	- <i>oflag</i> Values for <i>open()</i> . . . . .	122
Table 6-5	- File Status Flags Used for <i>open()</i> and <i>fcntl()</i> . . . . .	122
Table 6-6	- File Access Modes Used for <i>open()</i> and <i>fcntl()</i> . . . . .	123
Table 6-7	- Mask for Use With File Access Modes . . . . .	123
Table 6-8	- <i>flock</i> Structure . . . . .	125
Table 6-9	- <i>fcntl()</i> Return Values . . . . .	126
Table 7-1	- <i>termios</i> Structure . . . . .	136
Table 7-2	- <i>termios c_iflag</i> Field . . . . .	136
Table 7-3	- <i>termios c_cflag</i> Field . . . . .	138
Table 7-4	- <i>termios c_lflag</i> Field . . . . .	139
Table 7-5	- <i>termios c_cc</i> Special Control Characters . . . . .	140
Table 7-6	- <i>termios</i> Baud Rate Values . . . . .	141
Table 9-1	- <i>group</i> Structure . . . . .	166
Table 9-2	- <i>passwd</i> Structure . . . . .	167
Table 10-1	- <i>tar</i> Header Block . . . . .	170
Table 10-2	- Byte-Oriented <i>cpio</i> Archive Entry . . . . .	174
Table 10-3	- Values for <i>cpio c_mode</i> Field . . . . .	175
Table B-1	- Suggested Feature Test Macros . . . . .	222

STANDARD PREVIEW  
(standards.iteh.ai)

ISO/IEC 9945-1:1990

<https://standards.iteh.ai/catalog/standards/sist/f6cb600-97e1-480a-9304-7c5612b50049/iso-iec-9945-1-1990>

## Foreword

1 ISO (the International Organization for Standardization) and IEC (the International  
2 Electrotechnical Commission) together form a system for worldwide stan-  
3 dardization as a whole. National bodies that are members of ISO or IEC partici-  
4 pate in the development of International Standards through technical committees  
5 established by the respective organization to deal with particular fields of techni-  
6 cal activity. ISO and IEC technical committees collaborate in fields of mutual  
7 interest. Other international organizations, governmental and nongovernmental,  
8 in liaison with ISO and IEC, also take part in the work.

9 In the field of information technology, ISO and IEC have established a joint techni-  
10 cal committee, ISO/IEC JTC 1. Draft International Standards adopted by the joint  
11 technical committee are circulated to national bodies for approval before their  
12 acceptance as International Standards. They are approved in accordance with  
13 procedures requiring at least 75% approval by the national bodies voting.

14 International Standard ISO/IEC 9945-1:1990 was prepared by Joint Technical  
15 Committee ISO/IEC JTC 1, *Information technology*.

16 ISO/IEC 9945 consists of the following parts, under the general title *Information*  
17 *technology—Portable operating system interface (POSIX)*:

- 18 — *Part 1: System application program interface (API) [C language]*
- 19 — *Part 2: Shell and utilities* (under development)
- 20 — *Part 3: System administration* (under development)

21 Annexes A to E of ISO/IEC 9945-1 are provided for information only.



International Organization for Standardization/International Electrotechnical Commission  
Case postale 56 • CH-1211 Genève 20 • Switzerland

## Introduction

(This Introduction is not a normative part of ISO/IEC 9945-1 Information technology—Portable operating system interface (POSIX)—Part 1: System application programming interface (API) [C Language], but is included for information only.)

1 The purpose of this part of ISO/IEC 9945 is to define a standard operating system  
2 interface and environment based on the UNIX<sup>1)</sup> Operating System documentation  
3 to support application portability at the source level. This is intended for systems  
4 implementors and applications software developers.

5 Initially,<sup>2)</sup> the focus of this part of ISO/IEC 9945 is to provide standardized ser-  
6 vices via a C language interface. Future revisions are expected to contain bind-  
7 ings for other programming languages as well as for the C language. This will be  
8 accomplished by breaking this part of ISO/IEC 9945 into multiple portions—one  
9 defining core requirements independent of any programming language, and oth-  
10 ers composed of programming language bindings.

11 The core requirements portion will define a set of required services common to  
12 any programming language that can be reasonably expected to form a language  
13 binding to this part of ISO/IEC 9945. These services will be described in terms of  
14 functional requirements and will not define programming language-dependent  
15 interfaces. Language bindings will consist of two major parts. One will contain  
16 the programming language's standardized interface for accessing the core services  
17 defined in the programming language-independent core requirements section of  
18 this part of ISO/IEC 9945. The other will contain a standardized interface for  
19 language-specific services. Any implementation claiming conformance to this part  
20 of ISO/IEC 9945 with any language binding will be required to comply with both  
21 sections of the language binding.

22 Within this document, the term "POSIX.1" refers to this part of ISO/IEC 9945  
23 itself.

### 24 Organization of This Part of ISO/IEC 9945

25 This part of ISO/IEC 9945 is divided into four elements:

- 26 (1) Statement of scope and list of normative references (Section 1)
- 27 (2) Definitions and global concepts (Section 2)
- 28 (3) The various interface facilities (Sections 3 through 9)
- 29 (4) Data interchange format (Section 10)

30 1) UNIX is a registered trademark of AT&T in the USA and other countries.

31 2) The vertical rules in the right margin depict technical or significant non-editorial changes from  
32 IEEE Std 1003.1-1988 to IEEE Std 1003.1-1990. A vertical rule beside an empty line indicates  
33 deleted text.

34 Most of the sections describe a single service interface. The C Language binding  
35 for the service interface is given in the subclause labeled Synopsis. The Descrip-  
36 tion subclause provides a specification of the operation performed by the service  
37 interface. Some examples may be provided to illustrate the interfaces described.  
38 In most cases there are also Returns and Errors subclauses specifying return  
39 values and possible error conditions. References are used to direct the reader to  
40 other related sections. Additional material to complement sections in this part of  
41 ISO/IEC 9945 may be found in the Rationale and Notes, Annex B. This annex pro-  
42 vides historical perspectives into the technical choices made by the developers of  
43 this part of ISO/IEC 9945. It also provides information to emphasize consequences  
44 of the interfaces described in the corresponding section of this part of  
45 ISO/IEC 9945.

46 Informative annexes are not part of the standard and are provided for information  
47 only. (There is a type of annex called "normative" that is part of a standard and  
48 imposes requirements, but there are currently no such normative annexes in this  
49 part of ISO/IEC 9945.) They are provided for guidance and to help understanding.

50 In publishing this part of ISO/IEC 9945, its developers simply intend to provide a  
51 yardstick against which various operating system implementations can be meas-  
52 ured for conformance. It is *not* the intent of the developers to measure or rate any  
53 products, to reward or sanction any vendors of products for conformance or lack of  
54 conformance to this part of ISO/IEC 9945, or to attempt to enforce this part of  
55 ISO/IEC 9945 by these or any other means. The responsibility for determining the  
56 degree of conformance or lack thereof with this part of ISO/IEC 9945 rests solely  
57 with the individual who is evaluating the product claiming to be in conformance  
58 with this part of ISO/IEC 9945.

## 59 **Base Documents**

ISO/IEC 9945-1:1990

[https://standards.iteh.ai/catalog/standards/sist/febcb600-97e1-480a-9304-](https://standards.iteh.ai/catalog/standards/sist/febcb600-97e1-480a-9304-3c121579317c/iso-9945-1-1990)

60 The various interface facilities described herein are based on the 1984 */usr/group*  
61 *Standard* derived and published by the UniForum (formerly */usr/group*) Stan-  
62 dards Committee. The 1984 */usr/group Standard* and this part of ISO/IEC 9945  
63 are largely based on UNIX Seventh Edition, UNIX System III, UNIX System V,  
64 4.2BSD, and 4.3BSD documentation,<sup>3)</sup> but wherever possible, compatibility with  
65 other systems derived from the UNIX operating system, or systems compatible  
66 with that system, has been maintained.

## 67 **Background**

68 The developers of POSIX.1 represent a cross-section of hardware manufacturers,  
69 vendors of operating systems and other software development tools, software  
70 designers, consultants, academics, authors, applications programmers, and oth-  
71 ers. In the course of their deliberations, the developers reviewed related Ameri-  
72 can and international standards, both published and in progress.

73 Conceptually, POSIX.1 describes a set of fundamental services needed for the  
74 efficient construction of application programs. Access to these services has been

75 3) The IEEE is grateful to both AT&T and UniForum for permission to use their materials.

76 provided by defining an interface, using the C programming language, that estab-  
77 lishes standard semantics and syntax. Since this interface enables application  
78 writers to write portable applications—it was developed with that goal in mind—  
79 it has been designated POSIX,<sup>4)</sup> an acronym for Portable Operating System  
80 Interface.

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

## 87 Audience

88 The intended audience for ISO/IEC 9945 is all persons concerned with an  
89 industry-wide standard operating system based on the UNIX system. This  
90 includes at least four groups of people:

- 91 (1) Persons buying hardware and software systems;
- 92 (2) Persons managing companies that are deciding on future corporate com-  
93 puting directions;
- 94 (3) Persons implementing operating systems, and especially
- 95 (4) Persons developing applications where portability is an objective.

## 96 Purpose

97 Several principles guided the development of this part of ISO/IEC 9945:

### 98 Application Oriented

99 The basic goal was to promote portability of application programs across  
100 UNIX system environments by developing a clear, consistent, and unam-  
101 biguous standard for the interface specification of a portable operating  
102 system based on the UNIX system documentation. This part of  
103 ISO/IEC 9945 codifies the common, existing definition of the UNIX sys-  
104 tem. There was no attempt to define a new system interface.

## 105 Interface, Not Implementation

106 This part of ISO/IEC 9945 defines an interface, not an implementation.  
107 No distinction is made between library functions and system calls: both  
108 are referred to as functions. No details of the implementation of any  
109 function are given (although historical practice is sometimes indicated  
110 in Annex B). Symbolic names are given for constants (such as signals  
111 and error numbers) rather than numbers.

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