NAME

abort − cause abnormal process termination

SYNOPSIS

#include <stdlib.h>

void abort(void);

DESCRIPTION

The **abort**() first unblocks the **SIGABRT** signal, and then raises that signal for the calling process (as though **raise**(3) was called). This results in the abnormal termination of the process unless the **SIGABRT** signal is caught and the signal handler does not return (see **longjmp**(3)).

If the **SIGABRT** signal is ignored, or caught by a handler that returns, the **abort**() function will still terminate the process. It does this by restoring the default disposition for **SIGABRT** and then raising the signal for a second time.

RETURN VALUE

The **abort**() function never returns.

ATTRIBUTES

For an explanation of the terms used in this section, see **attributes**(7).

Interface	Attribute	Value
abort()	Thread safety	MT-Safe

NOTES

Up until glibc 2.26, if the **abort**() function caused process termination, all open streams were closed and flushed (as with **fclose**(3)). However, in some cases this could result in deadlocks and data corruption. Therefore, starting with glibc 2.27, **abort**() terminates the process without flushing streams. POSIX.1 permits either possible behavior, saying that **abort**() "may include an attempt to effect fclose() on all open streams".

CONFORMING TO

SVr4, POSIX.1-2001, POSIX.1-2008, 4.3BSD, C89, C99.

SEE ALSO

gdb(1), **sigaction**(2), **assert**(3), **exit**(3), **longjmp**(3), **raise**(3)

COLOPHON

This page is part of release 5.02 of the Linux *man-pages* project. A description of the project, information about reporting bugs, and the latest version of this page, can be found at https://www.kernel.org/doc/man−pages/.