CERT C++

Perforce QAC for C++ 2026.1

CERT C++ 2016 Edition plus website 25 March 2024.

The SEI CERT C++ standard focuses on issues specific to C++ that are not covered by the CERT C Rules. The CERT C rules that are applicable to C++ are included in the list and those that are not applicable have been explicitly excluded.

Rule Enforcement Summary

		Total
a	Total Number of Rules	163
b	Total Number of 'Not Statically Enforceable' Rules (Assisted/Unassisted)	0
c	Total Number of Enforceable Rules (a-b)	163
d	Total Number of Enforced Rules	163
e	Total Number of Unenforced Rules (c-d)	0
f	Enforced Rules Percentage (d/c)	100%
g	Unenforced Rules Percentage (e/c)	0%

Id	Description	Level	Enforced
Declarations and Initialization (DCL)			
DCL30-C	Declare objects with appropriate storage durations	L2	Yes
DCL39-C	Avoid information leakage when passing a structure across a trust boundary	L3	Yes
DCL40-C	Do not create incompatible declarations of the same function or object	L3	Yes
DCL50-CPP	Do not define a C-style variadic function	L1	Yes
DCL51-CPP	Do not declare or define a reserved identifier	L3	Yes
DCL52-CPP	Never qualify a reference type with const or volatile	L3	Yes
DCL53-CPP	Do not write syntactically ambiguous declarations	L3	Yes

© Perforce Software, Inc. All rights reserved.

Id	Description	Level	Enforced
DCL54-CPP	Overload allocation and deallocation functions as a pair in the same scope	L2	Yes
DCL55-CPP	Avoid information leakage when passing a class object across a trust boundary	L3	Yes
DCL56-CPP	Avoid cycles during initialization of static objects	L3	Yes
DCL57-CPP	Do not let exceptions escape from destructors or deallocation functions	L2	Yes
DCL58-CPP	Do not modify the standard namespaces	L2	Yes
DCL59-CPP	Do not define an unnamed namespace in a header file	L3	Yes
DCL60-CPP	Obey the one-definition rule	L3	Yes
Expressions (EXP)			
EXP34-C	Do not dereference null pointers	L1	Yes
EXP35-C	Do not modify objects with temporary lifetime	L2	Yes
EXP36-C	Do not cast pointers into more strictly aligned pointer types	L3	Yes
EXP37-C	Call functions with the correct number and type of arguments	L3	Yes
EXP39-C	Do not access a variable through a pointer of an incompatible type	L3	Yes
EXP42-C	Do not compare padding data	L1	Yes
EXP45-C	Do not perform assignments in selection statements	L2	Yes
EXP46-C	Do not use a bitwise operator with a Boolean-like operand	L2	Yes
EXP47-C	Do not call va_arg with an argument of the incorrect type	L2	Yes
EXP50-CPP	Do not depend on the order of evaluation for side effects	L2	Yes
EXP51-CPP	Do not delete an array through a pointer of the incorrect type	L3	Yes
EXP52-CPP	Do not rely on side effects in unevaluated operands	L3	Yes
EXP53-CPP	Do not read uninitialized memory	L1	Yes
EXP54-CPP	Do not access an object outside of its lifetime	L2	Yes
EXP55-CPP	Do not access a cv-qualified object through a cv-unqualified type	L2	Yes

© Perforce Software, Inc. All rights reserved.

Id	Description	Level	Enforced
EXP56-CPP	Do not call a function with a mismatched language linkage	L3	Yes
EXP57-CPP	Do not cast or delete pointers to incomplete classes	L3	Yes
EXP58-CPP	Pass an object of the correct type to va_start	L3	Yes
EXP59-CPP	Use offsetof() on valid types and members	L3	Yes
EXP60-CPP	Do not pass a nonstandard-layout type object across execution boundaries	L1	Yes
EXP61-CPP	A lambda object must not outlive any of its reference captured objects	L2	Yes
EXP62-CPP	Do not access the bits of an object representation that are not part of the object's value representation	L2	Yes
EXP63-CPP	Do not rely on the value of a moved-from object	L2	Yes
Integers (INT)			
INT30-C	Ensure that unsigned integer operations do not wrap	L2	Yes
INT31-C	Ensure that integer conversions do not result in lost or misinterpreted data	L1	Yes
INT32-C	Ensure that operations on signed integers do not result in overflow	L1	Yes
INT33-C	Ensure that division and remainder operations do not result in divide-by-zero errors	L2	Yes
INT34-C	Do not shift an expression by a negative number of bits or by greater than or equal to the number of bits that exist in the operand	L3	Yes
INT35-C	Use correct integer precisions	L3	Yes
INT36-C	Converting a pointer to integer or integer to pointer	L3	Yes
INT50-CPP	Do not cast to an out-of-range enumeration value	L3	Yes
Containers (CTR)			
ARR30-C	Do not form or use out-of-bounds pointers or array subscripts	L2	Yes
ARR37-C	Do not add or subtract an integer to a pointer to a non-array object	L2	Yes
ARR38-C	Guarantee that library functions do not form invalid pointers	L1	Yes
ARR39-C	Do not add or subtract a scaled integer to a pointer	L2	Yes

© Perforce Software, Inc. All rights reserved.

Id	Description	Level	Enforced
CTR50-CPP	Guarantee that container indices and iterators are within the valid range	L2	Yes
CTR51-CPP	Use valid references, pointers, and iterators to reference elements of a container	L2	Yes
CTR52-CPP	Guarantee that library functions do not overflow	L1	Yes
CTR53-CPP	Use valid iterator ranges	L2	Yes
CTR54-CPP	Do not subtract iterators that do not refer to the same container	L2	Yes
CTR55-CPP	Do not use an additive operator on an iterator if the result would overflow	L1	Yes
CTR56-CPP	Do not use pointer arithmetic on polymorphic objects	L2	Yes
CTR57-CPP	Provide a valid ordering predicate	L3	Yes
CTR58-CPP	Predicate function objects should not be mutable	L3	Yes
Characters and Strings (STR)			
STR30-C	Do not attempt to modify string literals	L2	Yes
STR31-C	Guarantee that storage for strings has sufficient space for character data and the null terminator	L2	Yes
STR32-C	Do not pass a non-null-terminated character sequence to a library function that expects a string	L1	Yes
STR34-C	Cast characters to unsigned char before converting to larger integer sizes	L2	Yes
STR37-C	Arguments to character-handling functions must be representable as an unsigned char	L3	Yes
STR38-C	Do not confuse narrow and wide character strings and functions	L1	Yes
STR50-CPP	Guarantee that storage for strings has sufficient space for character data and the null terminator	L1	Yes
STR51-CPP	Do not attempt to create a std::string from a null pointer	L1	Yes
STR52-CPP	Use valid references, pointers, and iterators to reference elements of a basic_string	L2	Yes
STR53-CPP	Range check element access	L2	Yes

© Perforce Software, Inc. All rights reserved.

Id	Description	Level	Enforced
Memory Management (MEM)			
MEM30-C	Do not access freed memory	L2	Yes
MEM31-C	Free dynamically allocated memory when no longer needed	L3	Yes
MEM34-C	Only free memory allocated dynamically	L2	Yes
MEM35-C	Allocate sufficient memory for an object	L2	Yes
MEM36-C	Do not modify the alignment of objects by calling realloc()	L3	Yes
MEM50-CPP	Do not access freed memory	L1	Yes
MEM51-CPP	Properly deallocate dynamically allocated resources	L1	Yes
MEM52-CPP	Detect and handle memory allocation errors	L1	Yes
MEM53-CPP	Explicitly construct and destruct objects when manually managing object lifetime	L1	Yes
MEM54-CPP	Provide placement new with properly aligned pointers to sufficient storage capacity	L1	Yes
MEM55-CPP	Honor replacement dynamic storage management requirements	L1	Yes
MEM56-CPP	Do not store an already-owned pointer value in an unrelated smart pointer	L1	Yes
MEM57-CPP	Avoid using default operator new for over-aligned types	L2	Yes
Input Output (FIO)			
FIO30-C	Exclude user input from format strings	L1	Yes
FIO32-C	Do not perform operations on devices that are only appropriate for files	L3	Yes
FIO34-C	Distinguish between characters read from a file and EOF or WEOF	L1	Yes
FIO37-C	Do not assume that fgets() or fgetws() returns a nonempty string when successful	L1	Yes
FIO38-C	Do not copy a FILE object	L3	Yes
FIO39-C	Do not alternately input and output from a stream without an intervening flush or positioning call	L2	Yes
FIO40-C	Reset strings on fgets() or fgetws() failure	L2	Yes

© Perforce Software, Inc. All rights reserved.

Id	Description	Level	Enforced
FIO41-C	Do not call getc(), putc(), getwc(), or putwc() with a stream argument that has side effects	L3	Yes
FIO42-C	Close files when they are no longer needed	L3	Yes
FIO44-C	Only use values for fsetpos() that are returned from fgetpos()	L3	Yes
FIO45-C	Avoid TOCTOU race conditions while accessing files	L2	Yes
FIO46-C	Do not access a closed file	L3	Yes
FIO47-C	Use valid format strings	L2	Yes
FIO50-CPP	Do not alternately input and output from a file stream without an intervening positioning call	L2	Yes
FIO51-CPP	Close files when they are no longer needed	L3	Yes
Exceptions and Error Handling (ERR)			
ERR30-C	Set errno to zero before calling a library function known to set errno, and check errno only after the function returns a value indicating failure	L1	Yes
ERR32-C	Do not rely on indeterminate values of errno	L3	Yes
ERR33-C	Detect and handle standard library errors	L1	Yes
ERR34-C	Detect errors when converting a string to a number	L2	Yes
ERR50-CPP	Do not abruptly terminate the program	L3	Yes
ERR51-CPP	Handle all exceptions	L3	Yes
ERR52-CPP	Do not use setjmp() or longjmp()	L3	Yes
ERR53-CPP	Do not reference base classes or class data members in a constructor or destructor function-try-block handler	L3	Yes
ERR54-CPP	Catch handlers should order their parameter types from most derived to least derived	L1	Yes
ERR55-CPP	Honor exception specifications	L2	Yes
ERR56-CPP	Guarantee exception safety	L2	Yes
ERR57-CPP	Do not leak resources when handling exceptions	L3	Yes
ERR58-CPP	Handle all exceptions thrown before main() begins executing	L2	Yes

© Perforce Software, Inc. All rights reserved.

Id	Description	Level	Enforced
ERR59-CPP	Do not throw an exception across execution boundaries	L1	Yes
ERR60-CPP	Exception objects must be nothrow copy constructible	L3	Yes
ERR61-CPP	Catch exceptions by lvalue reference	L3	Yes
ERR62-CPP	Detect errors when converting a string to a number	L3	Yes
Object Oriented Programming (OOP)			
OOP50-CPP	Do not invoke virtual functions from constructors or destructors	L3	Yes
OOP51-CPP	Do not slice derived objects	L3	Yes
OOP52-CPP	Do not delete a polymorphic object without a virtual destructor	L2	Yes
OOP53-CPP	Write constructor member initializers in the canonical order	L3	Yes
OOP54-CPP	Gracefully handle self-copy assignment	L3	Yes
OOP55-CPP	Do not use pointer-to-member operators to access nonexistent members	L2	Yes
OOP56-CPP	Honor replacement handler requirements	L3	Yes
OOP57-CPP	Prefer special member functions and overloaded operators to C Standard Library functions	L2	Yes
OOP58-CPP	Copy operations must not mutate the source object	L2	Yes
Concurrency (CON)			
CON33-C	Avoid race conditions when using library functions	L3	Yes
CON37-C	Do not call signal() in a multithreaded program	L3	Yes
CON40-C	Do not refer to an atomic variable twice in an expression	L2	Yes
CON41-C	Wrap functions that can fail spuriously in a loop	L3	Yes
CON43-C	Do not allow data races in multithreaded code	L3	Yes
CON50-CPP	Do not destroy a mutex while it is locked	L3	Yes
CON51-CPP	Ensure actively held locks are released on exceptional conditions	L2	Yes
CON52-CPP	Prevent data races when accessing bit-fields from multiple threads	L2	Yes
CON53-CPP	Avoid deadlock by locking in a predefined order	L3	Yes

© Perforce Software, Inc. All rights reserved.

Id	Description	Level	Enforced
CON54-CPP	Wrap functions that can spuriously wake up in a loop	L3	Yes
CON55-CPP	Preserve thread safety and liveness when using condition variables	L3	Yes
CON56-CPP	Do not speculatively lock a non-recursive mutex that is already owned by the calling thread	L3	Yes
Miscellaneous (MSC)			
ENV30-C	Do not modify the object referenced by the return value of certain functions	L3	Yes
ENV31-C	Do not rely on an environment pointer following an operation that may invalidate it	L3	Yes
ENV32-C	All exit handlers must return normally	L1	Yes
ENV33-C	Do not call system()	L1	Yes
ENV34-C	Do not store pointers returned by certain functions	L3	Yes
FLP30-C	Do not use floating-point variables as loop counters	L2	Yes
FLP32-C	Prevent or detect domain and range errors in math functions	L2	Yes
FLP34-C	Ensure that floating-point conversions are within range of the new type	L3	Yes
FLP36-C	Preserve precision when converting integral values to floating-point type	L3	Yes
FLP37-C	Do not use object representations to compare floating-point values	L3	Yes
MSC30-C	Do not use the rand() function for generating pseudorandom numbers	L3	Yes
MSC32-C	Properly seed pseudorandom number generators	L1	Yes
MSC33-C	Do not pass invalid data to the asctime() function	L2	Yes
MSC37-C	Ensure that control never reaches the end of a non-void function	L2	Yes
MSC38-C	Do not treat a predefined identifier as an object if it might only be implemented as a macro	L3	Yes
MSC39-C	Do not call va_arg() on a va_list that has an indeterminate value	L3	Yes
MSC40-C	Do not violate constraints	L3	Yes
MSC41-C	Never hard code sensitive information	L2	Yes

© Perforce Software, Inc. All rights reserved.

Id	Description	Level	Enforced
MSC50-CPP	Do not use std::rand() for generating pseudorandom numbers	L2	Yes
MSC51-CPP	Ensure your random number generator is properly seeded	L1	Yes
MSC52-CPP	Value-returning functions must return a value from all exit paths	L2	Yes
MSC53-CPP	Do not return from a function declared [[noreturn]]	L3	Yes
MSC54-CPP	A signal handler must be a plain old function	L2	Yes
PRE30-C	Do not create a universal character name through concatenation	L3	Yes
PRE31-C	Avoid side effects in arguments to unsafe macros	L2	Yes
PRE32-C	Do not use preprocessor directives in invocations of function-like macros	L3	Yes
SIG31-C	Do not access shared objects in signal handlers	L1	Yes
SIG34-C	Do not call signal() from within interruptible signal handlers	L3	Yes
SIG35-C	Do not return from a computational exception signal handler	L3	Yes

POSIX Enforcement

Id	Description	Level	Enforced
POSIX (POS)			
POS30-C	Use the readlink() function properly	L1	No
POS34-C	Do not call putenv() with a pointer to an automatic variable as the argument	L2	No
POS35-C	Avoid race conditions while checking for the existence of a symbolic link	L2	Yes
POS36-C	Observe correct revocation order while relinquishing privileges	L1	Yes
POS37-C	Ensure that privilege relinquishment is successful	L1	Yes
POS38-C	Beware of race conditions when using fork and file descriptors	L3	No
POS39-C	Use the correct byte ordering when transferring data between systems	L1	No

© Perforce Software, Inc. All rights reserved.

Id	Description	Level	Enforced
POS44-C	Do not use signals to terminate threads	L3	No
POS47-C	Do not use threads that can be canceled asynchronously	L3	No
POS48-C	Do not unlock or destroy another POSIX thread's mutex	L3	Yes
POS49-C	When data must be accessed by multiple threads, provide a mutex and guarantee no adjacent data is also accessed	L3	No
POS50-C	Declare objects shared between POSIX threads with appropriate storage durations	L3	No
POS51-C	Avoid deadlock with POSIX threads by locking in predefined order	L3	No
POS52-C	Do not perform operations that can block while holding a POSIX lock	L3	No
POS53-C	Do not use more than one mutex for concurrent waiting operations on a condition variable	L2	Yes
POS54-C	Detect and handle POSIX library errors	L1	No

"CERT" is a registered trademark of Carnegie Mellon University.

© Perforce Software, Inc. All rights reserved.