**C++ static code analysis**

Unique rules to find Bugs, Vulnerabilities, Security Hotspots, and Code Smells in your C++ code

### ["memset" should not be used to delete sensitive data](https://rules.sonarsource.com/cpp/RSPEC-5798)

[Vulnerability](https://rules.sonarsource.com/cpp/RSPEC-5798)

### [POSIX functions should not be called with arguments that trigger buffer overflows](https://rules.sonarsource.com/cpp/RSPEC-5782)

[Vulnerability](https://rules.sonarsource.com/cpp/RSPEC-5782)

### [XML parsers should not be vulnerable to XXE attacks](https://rules.sonarsource.com/cpp/RSPEC-2755)

[Vulnerability](https://rules.sonarsource.com/cpp/RSPEC-2755)

### [Function-like macros should not be invoked without all of their arguments](https://rules.sonarsource.com/cpp/RSPEC-961)

[Bug](https://rules.sonarsource.com/cpp/RSPEC-961)

### [The address of an automatic object should not be assigned to another object that may persist after the first object has ceased to exist](https://rules.sonarsource.com/cpp/RSPEC-946)

[Bug](https://rules.sonarsource.com/cpp/RSPEC-946)

### [Assigning to an optional should directly target the optional](https://rules.sonarsource.com/cpp/RSPEC-6427)

[Bug](https://rules.sonarsource.com/cpp/RSPEC-6427)

### [Result of the standard remove algorithms should not be ignored](https://rules.sonarsource.com/cpp/RSPEC-6223)

[Bug](https://rules.sonarsource.com/cpp/RSPEC-6223)

### ["std::scoped\_lock" should be created with constructor arguments](https://rules.sonarsource.com/cpp/RSPEC-5999)

[Bug](https://rules.sonarsource.com/cpp/RSPEC-5999)

### [Objects should not be sliced](https://rules.sonarsource.com/cpp/RSPEC-5912)

[Bug](https://rules.sonarsource.com/cpp/RSPEC-5912)

### [Immediately dangling references should not be created](https://rules.sonarsource.com/cpp/RSPEC-5553)

[Bug](https://rules.sonarsource.com/cpp/RSPEC-5553)

### ["pthread\_mutex\_t" should be unlocked in the reverse order they were locked](https://rules.sonarsource.com/cpp/RSPEC-5489)

[Bug](https://rules.sonarsource.com/cpp/RSPEC-5489)

### ["pthread\_mutex\_t" should be properly initialized and destroyed](https://rules.sonarsource.com/cpp/RSPEC-5487)

[Bug](https://rules.sonarsource.com/cpp/RSPEC-5487)

### ["pthread\_mutex\_t" should not be consecutively locked or unlocked twice](https://rules.sonarsource.com/cpp/RSPEC-5486)

[Bug](https://rules.sonarsource.com/cpp/RSPEC-5486)

### ["std::move" and "std::forward" should not be confused](https://rules.sonarsource.com/cpp/RSPEC-5417)

[Bug](https://rules.sonarsource.com/cpp/RSPEC-5417)

### [A call to "wait()" on a "std::condition\_variable" should have a condition](https://rules.sonarsource.com/cpp/RSPEC-5404)

[Bug](https://rules.sonarsource.com/cpp/RSPEC-5404)

### [A pointer to a virtual base class shall only be cast to a pointer to a derived class by means of dynamic\_cast](https://rules.sonarsource.com/cpp/RSPEC-5302)

[Bug](https://rules.sonarsource.com/cpp/RSPEC-5302)

### [Functions with "noreturn" attribute should not return](https://rules.sonarsource.com/cpp/RSPEC-5267)

[Bug](https://rules.sonarsource.com/cpp/RSPEC-5267)

### [RAII objects should not be temporary](https://rules.sonarsource.com/cpp/RSPEC-5184)

[Bug](https://rules.sonarsource.com/cpp/RSPEC-5184)

### ["memcmp" should only be called with pointers to trivially copyable types with no padding](https://rules.sonarsource.com/cpp/RSPEC-5000)

[Bug](https://rules.sonarsource.com/cpp/RSPEC-5000)

### ["memcpy", "memmove", and "memset" should only be called with pointers to trivially copyable types](https://rules.sonarsource.com/cpp/RSPEC-4999)

[Bug](https://rules.sonarsource.com/cpp/RSPEC-4999)

### ["std::auto\_ptr" should not be used](https://rules.sonarsource.com/cpp/RSPEC-4997)

[Bug](https://rules.sonarsource.com/cpp/RSPEC-4997)

### [Destructors should be "noexcept"](https://rules.sonarsource.com/cpp/RSPEC-3654)

[Bug](https://rules.sonarsource.com/cpp/RSPEC-3654)

### [Stack allocated memory and non-owned memory should not be freed](https://rules.sonarsource.com/cpp/RSPEC-3590)

[Bug](https://rules.sonarsource.com/cpp/RSPEC-3590)

### [Closed resources should not be accessed](https://rules.sonarsource.com/cpp/RSPEC-3588)

[Bug](https://rules.sonarsource.com/cpp/RSPEC-3588)

### [Dynamically allocated memory should be released](https://rules.sonarsource.com/cpp/RSPEC-3584)

[Bug](https://rules.sonarsource.com/cpp/RSPEC-3584)

### [Freed memory should not be used](https://rules.sonarsource.com/cpp/RSPEC-3529)

[Bug](https://rules.sonarsource.com/cpp/RSPEC-3529)

### [Memory locations should not be released more than once](https://rules.sonarsource.com/cpp/RSPEC-3520)

[Bug](https://rules.sonarsource.com/cpp/RSPEC-3520)

### [Memory access should be explicitly bounded to prevent buffer overflows](https://rules.sonarsource.com/cpp/RSPEC-3519)

[Bug](https://rules.sonarsource.com/cpp/RSPEC-3519)

### [Printf-style format strings should not lead to unexpected behavior at runtime](https://rules.sonarsource.com/cpp/RSPEC-2275)

[Bug](https://rules.sonarsource.com/cpp/RSPEC-2275)

### [Recursion should not be infinite](https://rules.sonarsource.com/cpp/RSPEC-2190)

[Bug](https://rules.sonarsource.com/cpp/RSPEC-2190)

### [Resources should be closed](https://rules.sonarsource.com/cpp/RSPEC-2095)

[Bug](https://rules.sonarsource.com/cpp/RSPEC-2095)

### [Appropriate memory de-allocation should be used](https://rules.sonarsource.com/cpp/RSPEC-1232)

[Bug](https://rules.sonarsource.com/cpp/RSPEC-1232)

### [Hard-coded credentials are security-sensitive](https://rules.sonarsource.com/cpp/RSPEC-2068)

[Security Hotspot](https://rules.sonarsource.com/cpp/RSPEC-2068)

### ["goto" should jump to labels declared later in the same function](https://rules.sonarsource.com/cpp/RSPEC-999)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-999)

### [The name "main" should not be used for any function other than the global "main" function](https://rules.sonarsource.com/cpp/RSPEC-998)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-998)

### [Only standard forms of the "defined" directive should be used](https://rules.sonarsource.com/cpp/RSPEC-969)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-969)

### [Switch labels should not be nested inside non-switch blocks](https://rules.sonarsource.com/cpp/RSPEC-916)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-916)

### [The right-hand operands of && and || should not contain side effects](https://rules.sonarsource.com/cpp/RSPEC-912)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-912)

### [Digraphs should not be used](https://rules.sonarsource.com/cpp/RSPEC-798)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-798)

### [Trigraphs should not be used](https://rules.sonarsource.com/cpp/RSPEC-797)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-797)

### [Use "std::variant" instead of unions with non-trivial types.](https://rules.sonarsource.com/cpp/RSPEC-6025)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-6025)

### [A single statement should not have more than one resource allocation](https://rules.sonarsource.com/cpp/RSPEC-5502)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-5502)

### [Facilities in <random> should be used instead of "srand", "rand" and "random\_shuffle"](https://rules.sonarsource.com/cpp/RSPEC-5020)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-5020)

### [Move and swap operations should be "noexcept"](https://rules.sonarsource.com/cpp/RSPEC-5018)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-5018)

### ["case" ranges should cover multiple values](https://rules.sonarsource.com/cpp/RSPEC-3936)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-3936)

### [Array indices should be placed between brackets](https://rules.sonarsource.com/cpp/RSPEC-3729)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-3729)

### [Comparison operators should not be virtual](https://rules.sonarsource.com/cpp/RSPEC-3692)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-3692)

### [Assignment operators should not be "virtual"](https://rules.sonarsource.com/cpp/RSPEC-3657)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-3657)

### [Redundant pointer operator sequences should be removed](https://rules.sonarsource.com/cpp/RSPEC-3491)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-3491)

### [Child class fields should not shadow parent class fields](https://rules.sonarsource.com/cpp/RSPEC-2387)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-2387)

### [Non-reentrant POSIX functions should be replaced with their reentrant versions](https://rules.sonarsource.com/cpp/RSPEC-1912)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-1912)

### ["goto" statements should not be used to jump into blocks](https://rules.sonarsource.com/cpp/RSPEC-1909)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-1909)

### [Keywords introduced in later specifications should not be used as identifiers](https://rules.sonarsource.com/cpp/RSPEC-1760)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-1760)

### [Context-sensitive keywords should not be used as identifiers](https://rules.sonarsource.com/cpp/RSPEC-1669)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-1669)

### [Switch cases should end with an unconditional "break" statement](https://rules.sonarsource.com/cpp/RSPEC-128)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-128)

### ["switch" statements should not contain non-case labels](https://rules.sonarsource.com/cpp/RSPEC-1219)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-1219)

### [Control should not be transferred into a complex logic block using a "goto" or a "switch" statement](https://rules.sonarsource.com/cpp/RSPEC-1036)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-1036)

### [Accessing files should not introduce TOCTOU vulnerabilities](https://rules.sonarsource.com/cpp/RSPEC-5847)

[Vulnerability](https://rules.sonarsource.com/cpp/RSPEC-5847)

### [Cipher algorithms should be robust](https://rules.sonarsource.com/cpp/RSPEC-5547)

[Vulnerability](https://rules.sonarsource.com/cpp/RSPEC-5547)

### [Encryption algorithms should be used with secure mode and padding scheme](https://rules.sonarsource.com/cpp/RSPEC-5542)

[Vulnerability](https://rules.sonarsource.com/cpp/RSPEC-5542)

### [Server hostnames should be verified during SSL/TLS connections](https://rules.sonarsource.com/cpp/RSPEC-5527)

[Vulnerability](https://rules.sonarsource.com/cpp/RSPEC-5527)

### [Server certificates should be verified during SSL/TLS connections](https://rules.sonarsource.com/cpp/RSPEC-4830)

[Vulnerability](https://rules.sonarsource.com/cpp/RSPEC-4830)

### [Cryptographic keys should be robust](https://rules.sonarsource.com/cpp/RSPEC-4426)

[Vulnerability](https://rules.sonarsource.com/cpp/RSPEC-4426)

### [Weak SSL/TLS protocols should not be used](https://rules.sonarsource.com/cpp/RSPEC-4423)

[Vulnerability](https://rules.sonarsource.com/cpp/RSPEC-4423)

### [Insecure functions should not be used](https://rules.sonarsource.com/cpp/RSPEC-1081)

[Vulnerability](https://rules.sonarsource.com/cpp/RSPEC-1081)

### ["scanf()" and "fscanf()" format strings should specify a field width for the "%s" string placeholder](https://rules.sonarsource.com/cpp/RSPEC-1079)

[Vulnerability](https://rules.sonarsource.com/cpp/RSPEC-1079)

### [Function exit paths should have appropriate return values](https://rules.sonarsource.com/cpp/RSPEC-935)

[Bug](https://rules.sonarsource.com/cpp/RSPEC-935)

### [Coroutine should have co\_return on each execution path or provide return\_void](https://rules.sonarsource.com/cpp/RSPEC-6369)

[Bug](https://rules.sonarsource.com/cpp/RSPEC-6369)

### ["volatile" should not be used to qualify objects for which the meaning is not defined](https://rules.sonarsource.com/cpp/RSPEC-6200)

[Bug](https://rules.sonarsource.com/cpp/RSPEC-6200)

### ["volatile" types should not be used in compound operations](https://rules.sonarsource.com/cpp/RSPEC-6191)

[Bug](https://rules.sonarsource.com/cpp/RSPEC-6191)

### [Values returned from string find-related methods should not be treated as boolean](https://rules.sonarsource.com/cpp/RSPEC-5972)

[Bug](https://rules.sonarsource.com/cpp/RSPEC-5972)

### [Relational and subtraction operators should not be used with pointers to different arrays](https://rules.sonarsource.com/cpp/RSPEC-5658)

[Bug](https://rules.sonarsource.com/cpp/RSPEC-5658)

### [Arguments evaluation order should not be relied on](https://rules.sonarsource.com/cpp/RSPEC-5570)

[Bug](https://rules.sonarsource.com/cpp/RSPEC-5570)

### ["reinterpret\_cast" should be used carefully](https://rules.sonarsource.com/cpp/RSPEC-5275)

[Bug](https://rules.sonarsource.com/cpp/RSPEC-5275)

### [Parameter values should be appropriate](https://rules.sonarsource.com/cpp/RSPEC-3807)

[Bug](https://rules.sonarsource.com/cpp/RSPEC-3807)

### [Zero should not be a possible denominator](https://rules.sonarsource.com/cpp/RSPEC-3518)

[Bug](https://rules.sonarsource.com/cpp/RSPEC-3518)

### [Line-splicing should not be used in "//" comments](https://rules.sonarsource.com/cpp/RSPEC-2323)

[Bug](https://rules.sonarsource.com/cpp/RSPEC-2323)

### [Member variables should be initialized](https://rules.sonarsource.com/cpp/RSPEC-2107)

[Bug](https://rules.sonarsource.com/cpp/RSPEC-2107)

### [Pointers should not be cast to integral types](https://rules.sonarsource.com/cpp/RSPEC-1767)

[Bug](https://rules.sonarsource.com/cpp/RSPEC-1767)

### ["operator delete" should be written along with "operator new"](https://rules.sonarsource.com/cpp/RSPEC-1265)

[Bug](https://rules.sonarsource.com/cpp/RSPEC-1265)

### [Destructors should not throw exceptions](https://rules.sonarsource.com/cpp/RSPEC-1048)

[Bug](https://rules.sonarsource.com/cpp/RSPEC-1048)

### [Handlers of a function-try-block implementation of a class constructor or destructor shall not reference non-static members from this class or its bases](https://rules.sonarsource.com/cpp/RSPEC-1042)

[Bug](https://rules.sonarsource.com/cpp/RSPEC-1042)

### [Empty throws ("throw;") should only be used in the compound statements of catch handlers](https://rules.sonarsource.com/cpp/RSPEC-1039)

[Bug](https://rules.sonarsource.com/cpp/RSPEC-1039)

### [An exception object should not have pointer type](https://rules.sonarsource.com/cpp/RSPEC-1035)

[Bug](https://rules.sonarsource.com/cpp/RSPEC-1035)

### ["sprintf" should not be used](https://rules.sonarsource.com/cpp/RSPEC-6069)

[Security Hotspot](https://rules.sonarsource.com/cpp/RSPEC-6069)

### [Changing working directories without verifying the success is security-sensitive](https://rules.sonarsource.com/cpp/RSPEC-5982)

[Security Hotspot](https://rules.sonarsource.com/cpp/RSPEC-5982)

### [Using "tmpnam", "tmpnam\_s" or "tmpnam\_r" is security-sensitive](https://rules.sonarsource.com/cpp/RSPEC-5824)

[Security Hotspot](https://rules.sonarsource.com/cpp/RSPEC-5824)

### [Changing directories improperly when using "chroot" is security-sensitive](https://rules.sonarsource.com/cpp/RSPEC-5802)

[Security Hotspot](https://rules.sonarsource.com/cpp/RSPEC-5802)

### [Using publicly writable directories is security-sensitive](https://rules.sonarsource.com/cpp/RSPEC-5443)

[Security Hotspot](https://rules.sonarsource.com/cpp/RSPEC-5443)

### [Using clear-text protocols is security-sensitive](https://rules.sonarsource.com/cpp/RSPEC-5332)

[Security Hotspot](https://rules.sonarsource.com/cpp/RSPEC-5332)

### [Expanding archive files without controlling resource consumption is security-sensitive](https://rules.sonarsource.com/cpp/RSPEC-5042)

[Security Hotspot](https://rules.sonarsource.com/cpp/RSPEC-5042)

### [Using weak hashing algorithms is security-sensitive](https://rules.sonarsource.com/cpp/RSPEC-4790)

[Security Hotspot](https://rules.sonarsource.com/cpp/RSPEC-4790)

### [Using pseudorandom number generators (PRNGs) is security-sensitive](https://rules.sonarsource.com/cpp/RSPEC-2245)

[Security Hotspot](https://rules.sonarsource.com/cpp/RSPEC-2245)

### ["#undef" should be used with caution](https://rules.sonarsource.com/cpp/RSPEC-959)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-959)

### [Function names should be used either as a call with a parameter list or with the "&" operator](https://rules.sonarsource.com/cpp/RSPEC-936)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-936)

### [Functions should not be defined with a variable number of arguments](https://rules.sonarsource.com/cpp/RSPEC-923)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-923)

### [The comma operator, "&&", and "||" should not be overloaded](https://rules.sonarsource.com/cpp/RSPEC-919)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-919)

### [A cast shall not remove any const or volatile qualification from the type of a pointer or reference](https://rules.sonarsource.com/cpp/RSPEC-859)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-859)

### [The return value of "std::move" should be used in a function](https://rules.sonarsource.com/cpp/RSPEC-6352)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-6352)

### [Cognitive Complexity of coroutines should not be too high](https://rules.sonarsource.com/cpp/RSPEC-6194)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-6194)

### [Use discriminated unions or "std::variant"](https://rules.sonarsource.com/cpp/RSPEC-6147)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-6147)

### [Multiple mutexes should not be acquired with individual locks](https://rules.sonarsource.com/cpp/RSPEC-5524)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-5524)

### [Pointers or references obtained from aliased smart pointers should not be used as function parameters](https://rules.sonarsource.com/cpp/RSPEC-5507)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-5507)

### ["try\_lock", "lock" and "unlock" should not be directly used for mutexes](https://rules.sonarsource.com/cpp/RSPEC-5506)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-5506)

### [Appropriate arguments should be passed to UNIX/POSIX functions](https://rules.sonarsource.com/cpp/RSPEC-5488)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-5488)

### [Appropriate arguments should be passed to stream functions](https://rules.sonarsource.com/cpp/RSPEC-5485)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-5485)

### ["Forwarding references" parameters should be used only to forward parameters](https://rules.sonarsource.com/cpp/RSPEC-5425)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-5425)

### [Non-const global variables should not be used](https://rules.sonarsource.com/cpp/RSPEC-5421)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-5421)

### [Functions that throw exceptions should not be used as hash functions](https://rules.sonarsource.com/cpp/RSPEC-5409)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-5409)

### [Blocking functions should not be called inside critical sections](https://rules.sonarsource.com/cpp/RSPEC-5314)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-5314)

### [Return value of "setuid" family of functions should always be checked](https://rules.sonarsource.com/cpp/RSPEC-5308)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-5308)

### [Size of variable length arrays should be positive](https://rules.sonarsource.com/cpp/RSPEC-5283)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-5283)

### [Argument of "printf" should be a format string](https://rules.sonarsource.com/cpp/RSPEC-5281)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-5281)

### ["mktemp" family of functions templates should have at least six trailing "X"s](https://rules.sonarsource.com/cpp/RSPEC-5280)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-5280)

### [Logical operators should not be confused with bitwise operators](https://rules.sonarsource.com/cpp/RSPEC-5263)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-5263)

### [Header guards should be followed by according "#define" macro](https://rules.sonarsource.com/cpp/RSPEC-5259)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-5259)

### [Template parameters should be preferred to "std::function" when configuring behavior at compile time](https://rules.sonarsource.com/cpp/RSPEC-5213)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-5213)

### [The addresses of standard library functions should not be taken](https://rules.sonarsource.com/cpp/RSPEC-5180)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-5180)

### [Macros should not be used to define constants](https://rules.sonarsource.com/cpp/RSPEC-5028)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-5028)

### [Memory should not be managed manually](https://rules.sonarsource.com/cpp/RSPEC-5025)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-5025)

### [Lambdas that capture "this" should capture everything explicitly](https://rules.sonarsource.com/cpp/RSPEC-5019)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-5019)

### ["void \*" should not be used in typedefs, member variables, function parameters or return type](https://rules.sonarsource.com/cpp/RSPEC-5008)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-5008)

### [The "Rule-of-Zero" should be followed](https://rules.sonarsource.com/cpp/RSPEC-4963)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-4963)

### ["nullptr" should be used to denote the null pointer](https://rules.sonarsource.com/cpp/RSPEC-4962)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-4962)

### ["default" clauses should be first or last](https://rules.sonarsource.com/cpp/RSPEC-4524)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-4524)

### [A conditionally executed single line should be denoted by indentation](https://rules.sonarsource.com/cpp/RSPEC-3973)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-3973)

### [Conditionals should start on new lines](https://rules.sonarsource.com/cpp/RSPEC-3972)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-3972)

### [Cognitive Complexity of functions should not be too high](https://rules.sonarsource.com/cpp/RSPEC-3776)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-3776)

### [Exceptions should not be thrown in "noexcept" functions](https://rules.sonarsource.com/cpp/RSPEC-3743)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-3743)

### [Member variables should not be "protected"](https://rules.sonarsource.com/cpp/RSPEC-3656)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-3656)

### [When the "Rule-of-Zero" is not applicable, the "Rule-of-Five" should be followed](https://rules.sonarsource.com/cpp/RSPEC-3624)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-3624)

### [Default capture should not be used](https://rules.sonarsource.com/cpp/RSPEC-3608)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-3608)

### [Standard groupings should be used with digit separators](https://rules.sonarsource.com/cpp/RSPEC-3543)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-3543)

### [Special member function should not be defined unless a non standard behavior is required](https://rules.sonarsource.com/cpp/RSPEC-3490)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-3490)

### [Standard namespaces should not be modified](https://rules.sonarsource.com/cpp/RSPEC-3470)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-3470)

### [Destructors should not be called explicitly](https://rules.sonarsource.com/cpp/RSPEC-3432)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-3432)

### ["static" base class members should not be accessed via derived types](https://rules.sonarsource.com/cpp/RSPEC-3252)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-3252)

### [Control characters should not be used in literals](https://rules.sonarsource.com/cpp/RSPEC-2479)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-2479)

### [Exception specifications should not be used](https://rules.sonarsource.com/cpp/RSPEC-2303)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-2303)

### [The sign of an unsigned variable should not be tested](https://rules.sonarsource.com/cpp/RSPEC-1768)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-1768)

### [Pre-defined macros should not be defined, redefined or undefined](https://rules.sonarsource.com/cpp/RSPEC-1761)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-1761)

### ["explicit" should be used on single-parameter constructors and conversion operators](https://rules.sonarsource.com/cpp/RSPEC-1709)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-1709)

### [Constructors and destructors should only use defined methods and fields](https://rules.sonarsource.com/cpp/RSPEC-1699)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-1699)

### [Control flow statements "if", "for", "while", "switch" and "try" should not be nested too deeply](https://rules.sonarsource.com/cpp/RSPEC-134)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-134)

### [Inherited functions should not be hidden](https://rules.sonarsource.com/cpp/RSPEC-1242)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-1242)

### [C-style memory allocation routines should not be used](https://rules.sonarsource.com/cpp/RSPEC-1231)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-1231)

### [Methods should not be empty](https://rules.sonarsource.com/cpp/RSPEC-1186)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-1186)

### [Pure "virtual" functions should not override non-pure "virtual" functions](https://rules.sonarsource.com/cpp/RSPEC-1017)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-1017)

### ["using namespace" directives should not be used in header files](https://rules.sonarsource.com/cpp/RSPEC-1003)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-1003)

### [Account validity should be verified when authenticating users with PAM](https://rules.sonarsource.com/cpp/RSPEC-5832)

[Vulnerability](https://rules.sonarsource.com/cpp/RSPEC-5832)

### [Lines starting with "#" should contain valid preprocessing directives](https://rules.sonarsource.com/cpp/RSPEC-977)

[Bug](https://rules.sonarsource.com/cpp/RSPEC-977)

### ["#include" directives should be followed by either <filename> or "filename" sequences](https://rules.sonarsource.com/cpp/RSPEC-956)

[Bug](https://rules.sonarsource.com/cpp/RSPEC-956)

### [Non-standard characters should not occur in header file names in "#include" directives](https://rules.sonarsource.com/cpp/RSPEC-955)

[Bug](https://rules.sonarsource.com/cpp/RSPEC-955)

### [Non-empty statements should change control flow or have at least one side-effect](https://rules.sonarsource.com/cpp/RSPEC-905)

[Bug](https://rules.sonarsource.com/cpp/RSPEC-905)

### [Unary minus should not be applied to an unsigned expression](https://rules.sonarsource.com/cpp/RSPEC-876)

[Bug](https://rules.sonarsource.com/cpp/RSPEC-876)

### [Objects with integer type should not be converted to objects with pointer type](https://rules.sonarsource.com/cpp/RSPEC-860)

[Bug](https://rules.sonarsource.com/cpp/RSPEC-860)

### [Variables should be initialized before use](https://rules.sonarsource.com/cpp/RSPEC-836)

[Bug](https://rules.sonarsource.com/cpp/RSPEC-836)

### [String literals with different prefixes should not be concatenated](https://rules.sonarsource.com/cpp/RSPEC-817)

[Bug](https://rules.sonarsource.com/cpp/RSPEC-817)

### [Only escape sequences defined in the ISO C standard should be used](https://rules.sonarsource.com/cpp/RSPEC-796)

[Bug](https://rules.sonarsource.com/cpp/RSPEC-796)

### ["std::bit\_cast" should be used instead of union type-punning](https://rules.sonarsource.com/cpp/RSPEC-6232)

[Bug](https://rules.sonarsource.com/cpp/RSPEC-6232)

### ["std::cmp\_\*" functions should be used to compare unsigned values with negative values](https://rules.sonarsource.com/cpp/RSPEC-6214)

[Bug](https://rules.sonarsource.com/cpp/RSPEC-6214)

### [Call to "std::is\_constant\_evaluated" should not be gratuitous](https://rules.sonarsource.com/cpp/RSPEC-6169)

[Bug](https://rules.sonarsource.com/cpp/RSPEC-6169)

### [Heterogeneous sorted containers should only be used with types that support heterogeneous comparison](https://rules.sonarsource.com/cpp/RSPEC-6021)

[Bug](https://rules.sonarsource.com/cpp/RSPEC-6021)

### ["#pragma pack" should be used correctly](https://rules.sonarsource.com/cpp/RSPEC-5501)

[Bug](https://rules.sonarsource.com/cpp/RSPEC-5501)

### [Enums should be consistent with the bit fields they initialize](https://rules.sonarsource.com/cpp/RSPEC-5491)

[Bug](https://rules.sonarsource.com/cpp/RSPEC-5491)

### [Class members should not be initialized with dangling references](https://rules.sonarsource.com/cpp/RSPEC-5269)

[Bug](https://rules.sonarsource.com/cpp/RSPEC-5269)

### [Array values should not be replaced unconditionally](https://rules.sonarsource.com/cpp/RSPEC-4143)

[Bug](https://rules.sonarsource.com/cpp/RSPEC-4143)

### [Integral operations should not overflow](https://rules.sonarsource.com/cpp/RSPEC-3949)

[Bug](https://rules.sonarsource.com/cpp/RSPEC-3949)

### ["case" ranges should not be empty](https://rules.sonarsource.com/cpp/RSPEC-3935)

[Bug](https://rules.sonarsource.com/cpp/RSPEC-3935)

### [All branches in a conditional structure should not have exactly the same implementation](https://rules.sonarsource.com/cpp/RSPEC-3923)

[Bug](https://rules.sonarsource.com/cpp/RSPEC-3923)

### ["extern" shouldn't be used on member definitions](https://rules.sonarsource.com/cpp/RSPEC-3726)

[Bug](https://rules.sonarsource.com/cpp/RSPEC-3726)

### [Declaration specifiers should not be redundant](https://rules.sonarsource.com/cpp/RSPEC-3689)

[Bug](https://rules.sonarsource.com/cpp/RSPEC-3689)

### [Function declarations that look like variable declarations should not be used](https://rules.sonarsource.com/cpp/RSPEC-3468)

[Bug](https://rules.sonarsource.com/cpp/RSPEC-3468)

### ["sizeof" should not be called on pointers](https://rules.sonarsource.com/cpp/RSPEC-3135)

[Bug](https://rules.sonarsource.com/cpp/RSPEC-3135)

### ["const" references to numbers should not be made](https://rules.sonarsource.com/cpp/RSPEC-2813)

[Bug](https://rules.sonarsource.com/cpp/RSPEC-2813)

### [Unary prefix operators should not be repeated](https://rules.sonarsource.com/cpp/RSPEC-2761)

[Bug](https://rules.sonarsource.com/cpp/RSPEC-2761)

### ["=+" should not be used instead of "+="](https://rules.sonarsource.com/cpp/RSPEC-2757)

[Bug](https://rules.sonarsource.com/cpp/RSPEC-2757)

### [Values of different "enum" types should not be compared](https://rules.sonarsource.com/cpp/RSPEC-2753)

[Bug](https://rules.sonarsource.com/cpp/RSPEC-2753)

### [Conditionally executed code should be reachable](https://rules.sonarsource.com/cpp/RSPEC-2583)

[Bug](https://rules.sonarsource.com/cpp/RSPEC-2583)

### [Null pointers should not be dereferenced](https://rules.sonarsource.com/cpp/RSPEC-2259)

[Bug](https://rules.sonarsource.com/cpp/RSPEC-2259)

### [Single-bit named bit fields should not be of a signed type](https://rules.sonarsource.com/cpp/RSPEC-2216)

[Bug](https://rules.sonarsource.com/cpp/RSPEC-2216)

### [Values should not be uselessly incremented](https://rules.sonarsource.com/cpp/RSPEC-2123)

[Bug](https://rules.sonarsource.com/cpp/RSPEC-2123)

### ["sizeof(sizeof(...))" should not be used](https://rules.sonarsource.com/cpp/RSPEC-1913)

[Bug](https://rules.sonarsource.com/cpp/RSPEC-1913)

### [Related "if/else if" statements should not have the same condition](https://rules.sonarsource.com/cpp/RSPEC-1862)

[Bug](https://rules.sonarsource.com/cpp/RSPEC-1862)

### [Identical expressions should not be used on both sides of a binary operator](https://rules.sonarsource.com/cpp/RSPEC-1764)

[Bug](https://rules.sonarsource.com/cpp/RSPEC-1764)

### [All code should be reachable](https://rules.sonarsource.com/cpp/RSPEC-1763)

[Bug](https://rules.sonarsource.com/cpp/RSPEC-1763)

### [Loops with at most one iteration should be refactored](https://rules.sonarsource.com/cpp/RSPEC-1751)

[Bug](https://rules.sonarsource.com/cpp/RSPEC-1751)

### [The original exception object should be rethrown](https://rules.sonarsource.com/cpp/RSPEC-1679)

[Bug](https://rules.sonarsource.com/cpp/RSPEC-1679)

### [Variables should not be self-assigned](https://rules.sonarsource.com/cpp/RSPEC-1656)

[Bug](https://rules.sonarsource.com/cpp/RSPEC-1656)

### [Condition-specific "catch" handlers should not be used after the ellipsis (catch-all) handler](https://rules.sonarsource.com/cpp/RSPEC-1046)

[Bug](https://rules.sonarsource.com/cpp/RSPEC-1046)

### [Handlers in a single try-catch or function-try-block for a derived class and some or all of its bases should be ordered most-derived-first](https://rules.sonarsource.com/cpp/RSPEC-1045)

[Bug](https://rules.sonarsource.com/cpp/RSPEC-1045)

### [Exception classes should be caught by reference](https://rules.sonarsource.com/cpp/RSPEC-1044)

[Bug](https://rules.sonarsource.com/cpp/RSPEC-1044)

### [Setting capabilities is security-sensitive](https://rules.sonarsource.com/cpp/RSPEC-5849)

[Security Hotspot](https://rules.sonarsource.com/cpp/RSPEC-5849)

### [Using "strncpy" or "wcsncpy" is security-sensitive](https://rules.sonarsource.com/cpp/RSPEC-5816)

[Security Hotspot](https://rules.sonarsource.com/cpp/RSPEC-5816)

### [Using "strncat" or "wcsncat" is security-sensitive](https://rules.sonarsource.com/cpp/RSPEC-5815)

[Security Hotspot](https://rules.sonarsource.com/cpp/RSPEC-5815)

### [Using "strcat" or "wcscat" is security-sensitive](https://rules.sonarsource.com/cpp/RSPEC-5814)

[Security Hotspot](https://rules.sonarsource.com/cpp/RSPEC-5814)

### [Using "strlen" or "wcslen" is security-sensitive](https://rules.sonarsource.com/cpp/RSPEC-5813)

[Security Hotspot](https://rules.sonarsource.com/cpp/RSPEC-5813)

### [Using "strcpy" or "wcscpy" is security-sensitive](https://rules.sonarsource.com/cpp/RSPEC-5801)

[Security Hotspot](https://rules.sonarsource.com/cpp/RSPEC-5801)

### [Setting loose POSIX file permissions is security-sensitive](https://rules.sonarsource.com/cpp/RSPEC-2612)

[Security Hotspot](https://rules.sonarsource.com/cpp/RSPEC-2612)

### [#include directives in a file should only be preceded by other preprocessor directives or comments](https://rules.sonarsource.com/cpp/RSPEC-954)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-954)

### [Loops should not have more than one "break" or "goto" statement](https://rules.sonarsource.com/cpp/RSPEC-924)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-924)

### [Unused type declarations should be removed](https://rules.sonarsource.com/cpp/RSPEC-897)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-897)

### [Comma operator should not be used](https://rules.sonarsource.com/cpp/RSPEC-878)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-878)

### [The unary "&" operator should not be overloaded](https://rules.sonarsource.com/cpp/RSPEC-877)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-877)

### ["bool" expressions should not be used as operands to built-in operators other than =, &&, ||, !, ==, !=, unary &, and the conditional operator](https://rules.sonarsource.com/cpp/RSPEC-872)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-872)

### ["enum" members other than the first one should not be explicitly initialized unless all members are explicitly initialized](https://rules.sonarsource.com/cpp/RSPEC-841)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-841)

### [If a function has internal linkage then all re-declarations shall include the static storage class specifer](https://rules.sonarsource.com/cpp/RSPEC-833)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-833)

### [Functions should not be declared at block scope](https://rules.sonarsource.com/cpp/RSPEC-824)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-824)

### [Bit fields should be declared with appropriate types](https://rules.sonarsource.com/cpp/RSPEC-814)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-814)

### [Coroutines should not take const references as parameters](https://rules.sonarsource.com/cpp/RSPEC-6391)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-6391)

### [Thread local variables should not be used in coroutines](https://rules.sonarsource.com/cpp/RSPEC-6367)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-6367)

### [Use symmetric transfer to switch execution between coroutines](https://rules.sonarsource.com/cpp/RSPEC-6365)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-6365)

### [rvalue reference members should not be copied accidentally](https://rules.sonarsource.com/cpp/RSPEC-6236)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-6236)

### ["std::string\_view" and "std::span" parameters should be directly constructed from sequences](https://rules.sonarsource.com/cpp/RSPEC-6231)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-6231)

### [Comparision operators ("<=>", "==") should be defaulted unless non-default behavior is required](https://rules.sonarsource.com/cpp/RSPEC-6230)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-6230)

### ["std::chrono" components should be used to operate on time](https://rules.sonarsource.com/cpp/RSPEC-6229)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-6229)

### ["std::enable\_if" should not be used](https://rules.sonarsource.com/cpp/RSPEC-6195)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-6195)

### ["std::source\_location" should be used instead of "\_\_FILE\_\_", "\_\_LINE\_\_", and "\_\_func\_\_" macros](https://rules.sonarsource.com/cpp/RSPEC-6190)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-6190)

### [Function template parameters should be named if reused](https://rules.sonarsource.com/cpp/RSPEC-6189)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-6189)

### [Redundant comparison operators should not be defined](https://rules.sonarsource.com/cpp/RSPEC-6186)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-6186)

### ["std::bit\_cast" should be used to reinterpret binary representation instead of "std::memcpy"](https://rules.sonarsource.com/cpp/RSPEC-6181)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-6181)

### ["[[likely]]" and "[[unlikely]]" should be used instead of compiler built-ins](https://rules.sonarsource.com/cpp/RSPEC-6180)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-6180)

### ["starts\_with" and "ends\_with" should be used for prefix and postfix checks](https://rules.sonarsource.com/cpp/RSPEC-6178)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-6178)

### [Designated initializers should be used in their C++ compliant form](https://rules.sonarsource.com/cpp/RSPEC-6172)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-6172)

### ["std::jthread" should be used instead of "std::thread"](https://rules.sonarsource.com/cpp/RSPEC-6168)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-6168)

### [Elements in a container should be erased with "std::erase" or "std::erase\_if"](https://rules.sonarsource.com/cpp/RSPEC-6165)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-6165)

### [Mathematical constants should not be hardcoded](https://rules.sonarsource.com/cpp/RSPEC-6164)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-6164)

### [Transparent comparator should be used with associative "std::string" containers](https://rules.sonarsource.com/cpp/RSPEC-6045)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-6045)

### ["emplace" should be prefered over "insert" with "std::set" and "std::unordered\_set"](https://rules.sonarsource.com/cpp/RSPEC-6033)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-6033)

### [Unnecessary expensive copy should be avoided when using auto as a placeholder type](https://rules.sonarsource.com/cpp/RSPEC-6032)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-6032)

### [The right template argument should be specified for std::forward](https://rules.sonarsource.com/cpp/RSPEC-6031)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-6031)

### ["try\_emplace" should be used with "std::map" and "std::unordered\_map"](https://rules.sonarsource.com/cpp/RSPEC-6030)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-6030)

### [Exception specifications should be treated as part of the type](https://rules.sonarsource.com/cpp/RSPEC-6029)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-6029)

### ["auto" should be used for non-type template parameter](https://rules.sonarsource.com/cpp/RSPEC-6026)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-6026)

### ["std::optional" member function "value\_or" should be used](https://rules.sonarsource.com/cpp/RSPEC-6023)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-6023)

### ["std::byte" should be used when you need byte-oriented memory access](https://rules.sonarsource.com/cpp/RSPEC-6022)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-6022)

### [Inline variables should be used to declare global variables in header files](https://rules.sonarsource.com/cpp/RSPEC-6018)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-6018)

### ["[\*this]" should be used to capture the current object by copy](https://rules.sonarsource.com/cpp/RSPEC-6016)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-6016)

### ["std::uncaught\_exception" should not be used](https://rules.sonarsource.com/cpp/RSPEC-6015)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-6015)

### [Objects should not be created solely to be passed as arguments to functions that perform delegated object creation](https://rules.sonarsource.com/cpp/RSPEC-6011)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-6011)

### ["std::filesystem::path" should be used to represent a file path](https://rules.sonarsource.com/cpp/RSPEC-6010)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-6010)

### [Fold expressions should be used instead of recursive template instantiations](https://rules.sonarsource.com/cpp/RSPEC-6008)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-6008)

### ["as\_const" should be used to make a value constant](https://rules.sonarsource.com/cpp/RSPEC-6006)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-6006)

### [Structured binding should be used](https://rules.sonarsource.com/cpp/RSPEC-6005)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-6005)

### [Emplacement should be prefered when insertion creates a temporary with sequence containers](https://rules.sonarsource.com/cpp/RSPEC-6003)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-6003)

### ["std::visit" should be used to switch on the type of the current value in a "std::variant"](https://rules.sonarsource.com/cpp/RSPEC-6000)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-6000)

### ["bind" should not be used](https://rules.sonarsource.com/cpp/RSPEC-5995)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-5995)

### [Use "make\_unique" and "make\_shared" to construct "unique\_ptr" and "shared\_ptr"](https://rules.sonarsource.com/cpp/RSPEC-5950)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-5950)

### [C-style array should not be used](https://rules.sonarsource.com/cpp/RSPEC-5945)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-5945)

### ["auto" should be used to avoid repetition of types](https://rules.sonarsource.com/cpp/RSPEC-5827)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-5827)

### [Integer literals should not be cast to bool](https://rules.sonarsource.com/cpp/RSPEC-5820)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-5820)

### [Member functions that don't mutate their objects should be declared "const"](https://rules.sonarsource.com/cpp/RSPEC-5817)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-5817)

### [Functions having rvalue reference arguments should "std::move" those arguments](https://rules.sonarsource.com/cpp/RSPEC-5500)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-5500)

### [Capture by reference in lambdas used locally](https://rules.sonarsource.com/cpp/RSPEC-5495)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-5495)

### [Size of bit fields should not exceed the size of their types](https://rules.sonarsource.com/cpp/RSPEC-5494)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-5494)

### ["std::move" should only be used where moving can happen](https://rules.sonarsource.com/cpp/RSPEC-5415)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-5415)

### [Classes should not contain both public and private data members](https://rules.sonarsource.com/cpp/RSPEC-5414)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-5414)

### [GNU attributes should be used correctly](https://rules.sonarsource.com/cpp/RSPEC-5297)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-5297)

### [Unevaluated operands should not have side effects](https://rules.sonarsource.com/cpp/RSPEC-5279)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-5279)

### [Size argument of memory functions should be consistent](https://rules.sonarsource.com/cpp/RSPEC-5278)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-5278)

### [Return value of "nodiscard" functions should not be ignored](https://rules.sonarsource.com/cpp/RSPEC-5277)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-5277)

### [Implicit casts should not lower precision](https://rules.sonarsource.com/cpp/RSPEC-5276)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-5276)

### ["std::move" should only be added when necessary](https://rules.sonarsource.com/cpp/RSPEC-5274)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-5274)

### [Appropriate size arguments should be passed to "strncat" and "strlcpy"](https://rules.sonarsource.com/cpp/RSPEC-5273)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-5273)

### [Moved-from objects should not be relied upon](https://rules.sonarsource.com/cpp/RSPEC-5272)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-5272)

### [Keywords shall not be used as macros identifiers](https://rules.sonarsource.com/cpp/RSPEC-5266)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-5266)

### [Incomplete types should not be deleted](https://rules.sonarsource.com/cpp/RSPEC-5265)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-5265)

### [Dereferenced null pointers should not be bound to references](https://rules.sonarsource.com/cpp/RSPEC-5262)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-5262)

### ["else" statements should be clearly matched with an "if"](https://rules.sonarsource.com/cpp/RSPEC-5261)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-5261)

### [Function pointers should not be used as function parameters](https://rules.sonarsource.com/cpp/RSPEC-5205)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-5205)

### [Function parameters should not be of type "std::unique\_ptr<T> const &"](https://rules.sonarsource.com/cpp/RSPEC-4998)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-4998)

### [Include directives should not rely on non-portable search strategy](https://rules.sonarsource.com/cpp/RSPEC-4263)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-4263)

### [Methods should not have identical implementations](https://rules.sonarsource.com/cpp/RSPEC-4144)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-4144)

### ["#include" paths should be portable](https://rules.sonarsource.com/cpp/RSPEC-3806)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-3806)

### ["#import" should not be used](https://rules.sonarsource.com/cpp/RSPEC-3805)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-3805)

### [Atomic types should be used instead of "volatile" types](https://rules.sonarsource.com/cpp/RSPEC-3687)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-3687)

### [String literals should not be immediately followed by macros](https://rules.sonarsource.com/cpp/RSPEC-3685)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-3685)

### ["reinterpret\_cast" should not be used](https://rules.sonarsource.com/cpp/RSPEC-3630)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-3630)

### ["switch" statements should cover all cases](https://rules.sonarsource.com/cpp/RSPEC-3562)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-3562)

### [Methods returns should not be invariant](https://rules.sonarsource.com/cpp/RSPEC-3516)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-3516)

### [Printf-style format strings should be used correctly](https://rules.sonarsource.com/cpp/RSPEC-3457)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-3457)

### [Conditional operators should not be nested](https://rules.sonarsource.com/cpp/RSPEC-3358)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-3358)

### [Member data should be initialized in-class or in a constructor initialization list](https://rules.sonarsource.com/cpp/RSPEC-3230)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-3230)

### ["this" should not be compared with null](https://rules.sonarsource.com/cpp/RSPEC-2815)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-2815)

### [The "delete" operator should only be used for pointers](https://rules.sonarsource.com/cpp/RSPEC-2808)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-2808)

### [Multiline blocks should be enclosed in curly braces](https://rules.sonarsource.com/cpp/RSPEC-2681)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-2681)

### [Increment should not be used to set boolean variables to 'true'](https://rules.sonarsource.com/cpp/RSPEC-2668)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-2668)

### [Boolean expressions should not be gratuitous](https://rules.sonarsource.com/cpp/RSPEC-2589)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-2589)

### [Standard C++ headers should be used](https://rules.sonarsource.com/cpp/RSPEC-2305)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-2305)

### [Parameters should be passed in the correct order](https://rules.sonarsource.com/cpp/RSPEC-2234)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-2234)

### ["static" members should be accessed statically](https://rules.sonarsource.com/cpp/RSPEC-2209)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-2209)

### [Obsolete POSIX functions should not be used](https://rules.sonarsource.com/cpp/RSPEC-1911)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-1911)

### [Two branches in a conditional structure should not have exactly the same implementation](https://rules.sonarsource.com/cpp/RSPEC-1871)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-1871)

### [Unused assignments should be removed](https://rules.sonarsource.com/cpp/RSPEC-1854)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-1854)

### [Structures should not have too many fields](https://rules.sonarsource.com/cpp/RSPEC-1820)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-1820)

### ["switch" statements should not have too many "case" clauses](https://rules.sonarsource.com/cpp/RSPEC-1479)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-1479)

### [Classes should not have too many methods](https://rules.sonarsource.com/cpp/RSPEC-1448)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-1448)

### [Sections of code should not be commented out](https://rules.sonarsource.com/cpp/RSPEC-125)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-125)

### [Pass by reference to const should be used for large input parameters](https://rules.sonarsource.com/cpp/RSPEC-1238)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-1238)

### [Assignment operators should return non-"const" references](https://rules.sonarsource.com/cpp/RSPEC-1236)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-1236)

### [Polymorphic base class destructor should be either public virtual or protected non-virtual](https://rules.sonarsource.com/cpp/RSPEC-1235)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-1235)

### [Lambdas should not have too many lines](https://rules.sonarsource.com/cpp/RSPEC-1188)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-1188)

### [Generic exceptions should not be caught](https://rules.sonarsource.com/cpp/RSPEC-1181)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-1181)

### [Unused function parameters should be removed](https://rules.sonarsource.com/cpp/RSPEC-1172)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-1172)

### [Unused functions and methods should be removed](https://rules.sonarsource.com/cpp/RSPEC-1144)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-1144)

### [Try-catch blocks should not be nested](https://rules.sonarsource.com/cpp/RSPEC-1141)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-1141)

### [Track uses of "FIXME" tags](https://rules.sonarsource.com/cpp/RSPEC-1134)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-1134)

### [Deprecated attributes should include explanations](https://rules.sonarsource.com/cpp/RSPEC-1123)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-1123)

### [Assignments should not be made from within sub-expressions](https://rules.sonarsource.com/cpp/RSPEC-1121)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-1121)

### [Generic exceptions should never be thrown](https://rules.sonarsource.com/cpp/RSPEC-112)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-112)

### [Variables should not be shadowed](https://rules.sonarsource.com/cpp/RSPEC-1117)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-1117)

### [Redundant pairs of parentheses should be removed](https://rules.sonarsource.com/cpp/RSPEC-1110)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-1110)

### [Inheritance tree of classes should not be too deep](https://rules.sonarsource.com/cpp/RSPEC-110)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-110)

### [Nested blocks of code should not be left empty](https://rules.sonarsource.com/cpp/RSPEC-108)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-108)

### [Functions should not have too many parameters](https://rules.sonarsource.com/cpp/RSPEC-107)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-107)

### [Unused "private" fields should be removed](https://rules.sonarsource.com/cpp/RSPEC-1068)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-1068)

### [Collapsible "if" statements should be merged](https://rules.sonarsource.com/cpp/RSPEC-1066)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-1066)

### [Unused labels should be removed](https://rules.sonarsource.com/cpp/RSPEC-1065)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-1065)

### [Virtual functions should be declared with the "virtual" keyword](https://rules.sonarsource.com/cpp/RSPEC-1016)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-1016)

### [Parameters in an overriding virtual function shall either use the same default arguments as the function they override, or else shall not specify any default arguments](https://rules.sonarsource.com/cpp/RSPEC-1006)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-1006)

### [Header files should not contain unnamed namespaces](https://rules.sonarsource.com/cpp/RSPEC-1000)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-1000)

### [The "sizeof" and "alignof" operator should not be used with operands of a "void" type](https://rules.sonarsource.com/cpp/RSPEC-2665)

[Bug](https://rules.sonarsource.com/cpp/RSPEC-2665)

### ["nonnull" pointers should not be set to null](https://rules.sonarsource.com/cpp/RSPEC-2637)

[Bug](https://rules.sonarsource.com/cpp/RSPEC-2637)

### ["for" loop counters should not have essentially floating type](https://rules.sonarsource.com/cpp/RSPEC-2193)

[Bug](https://rules.sonarsource.com/cpp/RSPEC-2193)

### [Line continuation characters '\' should not be followed by trailing whitespace](https://rules.sonarsource.com/cpp/RSPEC-1916)

[Bug](https://rules.sonarsource.com/cpp/RSPEC-1916)

### [Using hardcoded IP addresses is security-sensitive](https://rules.sonarsource.com/cpp/RSPEC-1313)

[Security Hotspot](https://rules.sonarsource.com/cpp/RSPEC-1313)

### [Pointer and reference parameters should be "const" if the corresponding object is not modified](https://rules.sonarsource.com/cpp/RSPEC-995)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-995)

### [The three expressions of a "for" statement should only be concerned with loop control](https://rules.sonarsource.com/cpp/RSPEC-886)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-886)

### [Literal suffix "L" for long integers shall be upper case](https://rules.sonarsource.com/cpp/RSPEC-818)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-818)

### [Use type-erased "coroutine\_handle" when applicable](https://rules.sonarsource.com/cpp/RSPEC-6372)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-6372)

### [Use conditional suspension to resume current coroutine](https://rules.sonarsource.com/cpp/RSPEC-6366)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-6366)

### ["auto" should be used to store a result of functions that conventionally return an iterator or a range](https://rules.sonarsource.com/cpp/RSPEC-6234)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-6234)

### ["std::has\_single\_bit" should be used to test if an integer is a power of two](https://rules.sonarsource.com/cpp/RSPEC-6228)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-6228)

### [Empty class members should be marked as "[[no\_unique\_address]]"](https://rules.sonarsource.com/cpp/RSPEC-6226)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-6226)

### ["std::to\_address" should be used to convert iterators to raw pointers](https://rules.sonarsource.com/cpp/RSPEC-6225)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-6225)

### ["[[nodiscard]]" attributes on types should include explanations](https://rules.sonarsource.com/cpp/RSPEC-6222)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-6222)

### [STL constrained algorithms with range parameter should be used when iterating over the entire range](https://rules.sonarsource.com/cpp/RSPEC-6197)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-6197)

### ["std::span" should be used for a uniform sequence of elements contiguous in memory](https://rules.sonarsource.com/cpp/RSPEC-6188)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-6188)

### [Operator spaceship "<=>" should be used to define comparable types](https://rules.sonarsource.com/cpp/RSPEC-6187)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-6187)

### ["std::midpoint" and "std::lerp" should be used for midpoint computation and linear interpolation](https://rules.sonarsource.com/cpp/RSPEC-6179)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-6179)

### ["contains" should be used to check if a key exists in a container](https://rules.sonarsource.com/cpp/RSPEC-6171)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-6171)

### [Free functions should be preferred to member functions when accessing a container in a generic context](https://rules.sonarsource.com/cpp/RSPEC-6024)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-6024)

### [The "\_t" and "\_v" version of type traits should be used instead of "::type" and "::value"](https://rules.sonarsource.com/cpp/RSPEC-6020)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-6020)

### ["if constexpr" should be preferred to overloading for metaprogramming](https://rules.sonarsource.com/cpp/RSPEC-6017)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-6017)

### ["static\_assert" with no message should be used over "static\_assert" with empty or redundant message](https://rules.sonarsource.com/cpp/RSPEC-6013)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-6013)

### [Redundant class template arguments should not be used](https://rules.sonarsource.com/cpp/RSPEC-6012)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-6012)

### ["std::string\_view" should be used to pass a read-only string to a function](https://rules.sonarsource.com/cpp/RSPEC-6009)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-6009)

### ["if","switch", and range-based for loop initializer should be used to reduce scope of variables](https://rules.sonarsource.com/cpp/RSPEC-6004)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-6004)

### ["std::scoped\_lock" should be used instead of "std::lock\_guard"](https://rules.sonarsource.com/cpp/RSPEC-5997)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-5997)

### [Multicharacter literals should not be used](https://rules.sonarsource.com/cpp/RSPEC-5978)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-5978)

### [Classes should explicitly specify the access level when specifying base classes](https://rules.sonarsource.com/cpp/RSPEC-5965)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-5965)

### ["std::initializer\_list" constructor should not overlap with other constructors](https://rules.sonarsource.com/cpp/RSPEC-5964)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-5964)

### [Threads should not be detached](https://rules.sonarsource.com/cpp/RSPEC-5962)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-5962)

### [Loop variables should be declared in the minimal possible scope](https://rules.sonarsource.com/cpp/RSPEC-5955)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-5955)

### ["shared\_ptr" should not be taken by rvalue reference](https://rules.sonarsource.com/cpp/RSPEC-5954)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-5954)

### [Inheriting constructors should be used](https://rules.sonarsource.com/cpp/RSPEC-5952)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-5952)

### [Return type of functions shouldn't be const qualified value](https://rules.sonarsource.com/cpp/RSPEC-5951)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-5951)

### [Macros should not be used as replacement to "typdef" and "using"](https://rules.sonarsource.com/cpp/RSPEC-5825)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-5825)

### [Concise syntax should be used for concatenatable namespaces](https://rules.sonarsource.com/cpp/RSPEC-5812)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-5812)

### [STL algorithms and range-based for loops should be preferred to traditional for loops](https://rules.sonarsource.com/cpp/RSPEC-5566)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-5566)

### ["using" should be preferred for type aliasing](https://rules.sonarsource.com/cpp/RSPEC-5416)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-5416)

### ["constexpr" functions should not be declared "inline"](https://rules.sonarsource.com/cpp/RSPEC-5408)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-5408)

### ["^" should not be confused with exponentiation](https://rules.sonarsource.com/cpp/RSPEC-5381)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-5381)

### [Pointer and reference local variables should be "const" if the corresponding object is not modified](https://rules.sonarsource.com/cpp/RSPEC-5350)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-5350)

### [Format strings should comply with ISO standards](https://rules.sonarsource.com/cpp/RSPEC-5293)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-5293)

### [Functions which do not return should be declared as "noreturn"](https://rules.sonarsource.com/cpp/RSPEC-5271)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-5271)

### [Macros should not be redefined](https://rules.sonarsource.com/cpp/RSPEC-3744)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-3744)

### ['extern "C"' should not be used with namespaces](https://rules.sonarsource.com/cpp/RSPEC-3732)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-3732)

### ["auto" should not be used as a storage class specifier](https://rules.sonarsource.com/cpp/RSPEC-3731)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-3731)

### ["#include\_next" should not be used](https://rules.sonarsource.com/cpp/RSPEC-3730)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-3730)

### [String literals should not be concatenated implicitly](https://rules.sonarsource.com/cpp/RSPEC-3728)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-3728)

### [Reference types should not be qualified with "const" or "volatile"](https://rules.sonarsource.com/cpp/RSPEC-3708)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-3708)

### [Partial specialization syntax should not be used for function templates](https://rules.sonarsource.com/cpp/RSPEC-3691)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-3691)

### [Alternative operators should not be used](https://rules.sonarsource.com/cpp/RSPEC-3659)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-3659)

### [Types and variables should be declared in separate statements](https://rules.sonarsource.com/cpp/RSPEC-3646)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-3646)

### [Scoped enumerations should be used](https://rules.sonarsource.com/cpp/RSPEC-3642)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-3642)

### ["const" and "volatile" should not be used in "enum" declarations](https://rules.sonarsource.com/cpp/RSPEC-3636)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-3636)

### [Jump statements should not be redundant](https://rules.sonarsource.com/cpp/RSPEC-3626)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-3626)

### ["static" should not be used in unnamed namespaces](https://rules.sonarsource.com/cpp/RSPEC-3609)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-3609)

### ["final" classes should not have "virtual" functions](https://rules.sonarsource.com/cpp/RSPEC-3576)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-3576)

### [Redundant lambda return types should be omitted](https://rules.sonarsource.com/cpp/RSPEC-3574)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-3574)

### [Declarations of functions defined outside of the class should not be marked as "inline"](https://rules.sonarsource.com/cpp/RSPEC-3548)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-3548)

### [Allocation and deallocation functions should not be explicitly declared "static"](https://rules.sonarsource.com/cpp/RSPEC-3541)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-3541)

### [Access specifiers should not be redundant](https://rules.sonarsource.com/cpp/RSPEC-3539)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-3539)

### [The "register" storage class specifier should not be used](https://rules.sonarsource.com/cpp/RSPEC-3522)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-3522)

### ["override" or "final" should be used instead of "virtual"](https://rules.sonarsource.com/cpp/RSPEC-3471)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-3471)

### [Empty "case" clauses that fall through to the "default" should be omitted](https://rules.sonarsource.com/cpp/RSPEC-3458)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-3458)

### [Namespaces should not be empty](https://rules.sonarsource.com/cpp/RSPEC-3261)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-3261)

### [Forward declarations should not be redundant](https://rules.sonarsource.com/cpp/RSPEC-3231)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-3231)

### [Members should be initialized in the order they are declared](https://rules.sonarsource.com/cpp/RSPEC-3229)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-3229)

### [Declarations should not be empty](https://rules.sonarsource.com/cpp/RSPEC-2754)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-2754)

### [General "catch" clauses should not be used](https://rules.sonarsource.com/cpp/RSPEC-2738)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-2738)

### ["catch" clauses should do more than rethrow](https://rules.sonarsource.com/cpp/RSPEC-2737)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-2737)

### [Exceptions should not be ignored](https://rules.sonarsource.com/cpp/RSPEC-2486)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-2486)

### ["final" classes should not have "protected" members](https://rules.sonarsource.com/cpp/RSPEC-2156)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-2156)

### ["final" should not be used redundantly](https://rules.sonarsource.com/cpp/RSPEC-1990)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-1990)

### [Redundant casts should not be used](https://rules.sonarsource.com/cpp/RSPEC-1905)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-1905)

### [Code annotated as deprecated should not be used](https://rules.sonarsource.com/cpp/RSPEC-1874)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-1874)

### ["#pragma warning (default: ...)" should not be used](https://rules.sonarsource.com/cpp/RSPEC-1762)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-1762)

### [Init-declarator-lists and member-declarator-lists should consist of single init-declarators and member-declarators respectively](https://rules.sonarsource.com/cpp/RSPEC-1659)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-1659)

### [Unused local variables should be removed](https://rules.sonarsource.com/cpp/RSPEC-1481)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-1481)

### ["switch" statements should have at least 3 "case" clauses](https://rules.sonarsource.com/cpp/RSPEC-1301)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-1301)

### [A "while" loop should be used instead of a "for" loop](https://rules.sonarsource.com/cpp/RSPEC-1264)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-1264)

### [Nested code blocks should not be used](https://rules.sonarsource.com/cpp/RSPEC-1199)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-1199)

### [Overriding member functions should do more than simply call the same member in the base class](https://rules.sonarsource.com/cpp/RSPEC-1185)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-1185)

### [Do not check emptiness with a size method when a dedicated function exists](https://rules.sonarsource.com/cpp/RSPEC-1155)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-1155)

### [Empty statements should be removed](https://rules.sonarsource.com/cpp/RSPEC-1116)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-1116)

### ["/\*" and "//" should not be used within comments](https://rules.sonarsource.com/cpp/RSPEC-1103)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-1103)

### [Classes should not be derived from virtual bases](https://rules.sonarsource.com/cpp/RSPEC-1011)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-1011)

### [Track uses of "TODO" tags](https://rules.sonarsource.com/cpp/RSPEC-1135)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-1135)

### [Deprecated code should be removed](https://rules.sonarsource.com/cpp/RSPEC-1133)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-1133)

### [Reserved identifiers and functions in the C standard library should not be defined or declared](https://rules.sonarsource.com/cpp/RSPEC-978)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-978)

### [In the definition of a function-like macro, each instance of a parameter shall be enclosed in parentheses, unless it is used as the operand of # or ##](https://rules.sonarsource.com/cpp/RSPEC-963)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-963)

### [Bit fields should not be used](https://rules.sonarsource.com/cpp/RSPEC-2806)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-2806)

### [Track lack of copyright and license headers](https://rules.sonarsource.com/cpp/RSPEC-1451)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-1451)

### [Octal values should not be used](https://rules.sonarsource.com/cpp/RSPEC-1314)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-1314)

### [Function templates should not be specialized](https://rules.sonarsource.com/cpp/RSPEC-1032)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-1032)

### ["abort", "exit", "getenv" and "system" from <stdlib.h> should not be used](https://rules.sonarsource.com/cpp/RSPEC-990)

[Bug](https://rules.sonarsource.com/cpp/RSPEC-990)

### ["atof", "atoi" and "atol" from <stdlib.h> should not be used](https://rules.sonarsource.com/cpp/RSPEC-989)

[Bug](https://rules.sonarsource.com/cpp/RSPEC-989)

### ["<signal.h>" should not be used](https://rules.sonarsource.com/cpp/RSPEC-987)

[Bug](https://rules.sonarsource.com/cpp/RSPEC-987)

### [Dynamic heap memory allocation should not be used](https://rules.sonarsource.com/cpp/RSPEC-984)

[Bug](https://rules.sonarsource.com/cpp/RSPEC-984)

### [The global namespace should only contain "main", namespace declarations, and "extern" C declarations](https://rules.sonarsource.com/cpp/RSPEC-997)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-997)

### ["<time.h>" should not be used](https://rules.sonarsource.com/cpp/RSPEC-991)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-991)

### ["<stdio.h>" should not be used in production code](https://rules.sonarsource.com/cpp/RSPEC-988)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-988)

### ["offsetof" macro from <stddef.h> should not be used](https://rules.sonarsource.com/cpp/RSPEC-986)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-986)

### ["errno" should not be used](https://rules.sonarsource.com/cpp/RSPEC-985)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-985)

### ["setjmp" and "longjmp" should not be used](https://rules.sonarsource.com/cpp/RSPEC-982)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-982)

### [Function-like macros should not be used](https://rules.sonarsource.com/cpp/RSPEC-960)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-960)

### [Macros should not be #define'd or #undef'd within a block](https://rules.sonarsource.com/cpp/RSPEC-958)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-958)

### [Unions should not be used](https://rules.sonarsource.com/cpp/RSPEC-953)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-953)

### [Array type function arguments should not decay to pointers](https://rules.sonarsource.com/cpp/RSPEC-945)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-945)

### [Object declarations should contain no more than 2 levels of pointer indirection](https://rules.sonarsource.com/cpp/RSPEC-943)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-943)

### [Recursion should not be used](https://rules.sonarsource.com/cpp/RSPEC-925)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-925)

### [Constants of unsigned type should have a "U" suffix](https://rules.sonarsource.com/cpp/RSPEC-854)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-854)

### [Cyclomatic Complexity of coroutines should not be too high](https://rules.sonarsource.com/cpp/RSPEC-6192)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-6192)

### [Functions should not have more than one argument of type "bool"](https://rules.sonarsource.com/cpp/RSPEC-5422)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-5422)

### [using-directives and using-declarations (excluding class scope or function scope using-declarations) shall not be used in header files](https://rules.sonarsource.com/cpp/RSPEC-5318)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-5318)

### [Virtual functions should not have default arguments](https://rules.sonarsource.com/cpp/RSPEC-3719)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-3719)

### [Octal and hexadecimal escape sequences should be terminated](https://rules.sonarsource.com/cpp/RSPEC-2335)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-2335)

### [Flexible array members should not be declared](https://rules.sonarsource.com/cpp/RSPEC-2324)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-2324)

### [Preprocessor directives should not be indented](https://rules.sonarsource.com/cpp/RSPEC-1915)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-1915)

### ["switch" statements should not be nested](https://rules.sonarsource.com/cpp/RSPEC-1821)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-1821)

### [Lambdas should not be used](https://rules.sonarsource.com/cpp/RSPEC-1750)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-1750)

### [Cyclomatic Complexity of functions should not be too high](https://rules.sonarsource.com/cpp/RSPEC-1541)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-1541)

### [Cyclomatic Complexity of classes should not be too high](https://rules.sonarsource.com/cpp/RSPEC-1311)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-1311)

### ["switch" statements should have "default" clauses](https://rules.sonarsource.com/cpp/RSPEC-131)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-131)

### ["if ... else if" constructs should end with "else" clauses](https://rules.sonarsource.com/cpp/RSPEC-126)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-126)

### ["typedef" should be used for function pointers](https://rules.sonarsource.com/cpp/RSPEC-1259)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-1259)

### [Control structures should use curly braces](https://rules.sonarsource.com/cpp/RSPEC-121)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-121)

### [Expressions should not be too complex](https://rules.sonarsource.com/cpp/RSPEC-1067)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-1067)

### ["<cstdio>" should not be used](https://rules.sonarsource.com/cpp/RSPEC-1055)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-1055)

### ["<ctime>" should not be used](https://rules.sonarsource.com/cpp/RSPEC-1052)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-1052)

### [C libraries should not be used](https://rules.sonarsource.com/cpp/RSPEC-1051)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-1051)

### [Macros used in preprocessor directives should be defined before use](https://rules.sonarsource.com/cpp/RSPEC-966)

[Bug](https://rules.sonarsource.com/cpp/RSPEC-966)

### [Evaluation of the operand to the sizeof operator shall not contain side effects](https://rules.sonarsource.com/cpp/RSPEC-922)

[Bug](https://rules.sonarsource.com/cpp/RSPEC-922)

### [Bitwise operators should not be applied to signed operands](https://rules.sonarsource.com/cpp/RSPEC-874)

[Bug](https://rules.sonarsource.com/cpp/RSPEC-874)

### [Boolean operations should not have numeric operands, and vice versa](https://rules.sonarsource.com/cpp/RSPEC-867)

[Bug](https://rules.sonarsource.com/cpp/RSPEC-867)

### [Pointer conversions should be restricted to a safe subset](https://rules.sonarsource.com/cpp/RSPEC-856)

[Bug](https://rules.sonarsource.com/cpp/RSPEC-856)

### [Function pointers should not be converted to any other type](https://rules.sonarsource.com/cpp/RSPEC-855)

[Bug](https://rules.sonarsource.com/cpp/RSPEC-855)

### [Results of ~ and << operations on operands of underlying types unsigned char and unsigned short should immediately be cast to the operand's underlying type](https://rules.sonarsource.com/cpp/RSPEC-853)

[Bug](https://rules.sonarsource.com/cpp/RSPEC-853)

### [Each operand of the ! operator, the logical && or the logical || operators shall have type bool](https://rules.sonarsource.com/cpp/RSPEC-5359)

[Bug](https://rules.sonarsource.com/cpp/RSPEC-5359)

### [When an array is declared, its size shall either be stated explicitly or defined implicitly by initialization](https://rules.sonarsource.com/cpp/RSPEC-5298)

[Bug](https://rules.sonarsource.com/cpp/RSPEC-5298)

### [User-defined types should not be passed as variadic arguments](https://rules.sonarsource.com/cpp/RSPEC-5270)

[Bug](https://rules.sonarsource.com/cpp/RSPEC-5270)

### [Floating point numbers should not be tested for equality](https://rules.sonarsource.com/cpp/RSPEC-1244)

[Bug](https://rules.sonarsource.com/cpp/RSPEC-1244)

### [Multiple declarations for an identifier in the same namespace shall not straddle a using-declaration for that identifier](https://rules.sonarsource.com/cpp/RSPEC-1002)

[Bug](https://rules.sonarsource.com/cpp/RSPEC-1002)

### [There shall be at most one occurrence of the # or ## operators in a single macro definition](https://rules.sonarsource.com/cpp/RSPEC-967)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-967)

### [Parameters in a function prototype should be named](https://rules.sonarsource.com/cpp/RSPEC-926)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-926)

### ["goto" statement should not be used](https://rules.sonarsource.com/cpp/RSPEC-907)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-907)

### [A loop-control-variable other than the loop-counter which is modified in statement shall have type bool](https://rules.sonarsource.com/cpp/RSPEC-892)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-892)

### [Increment (++) and decrement (--) operators should not be used in a method call or mixed with other operators in an expression](https://rules.sonarsource.com/cpp/RSPEC-881)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-881)

### ["enum" values should not be used as operands to built-in operators other than [ ], =, ==, !=, unary &, and the relational operators <, <=, >, >=](https://rules.sonarsource.com/cpp/RSPEC-873)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-873)

### [C-style and functional notation casts should not be used](https://rules.sonarsource.com/cpp/RSPEC-871)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-871)

### [Operands of "&&" and "||" should be primary (C) or postfix (C++) expressions](https://rules.sonarsource.com/cpp/RSPEC-868)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-868)

### [Limited dependence should be placed on operator precedence](https://rules.sonarsource.com/cpp/RSPEC-864)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-864)

### [Braces should be used to indicate and match the structure in the non-zero initialization of arrays and structures](https://rules.sonarsource.com/cpp/RSPEC-835)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-835)

### [Array declarations should include an explicit size specification](https://rules.sonarsource.com/cpp/RSPEC-834)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-834)

### [Objects or functions with external linkage shall be declared in a header file](https://rules.sonarsource.com/cpp/RSPEC-831)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-831)

### ["typedef" names should be unique identifiers](https://rules.sonarsource.com/cpp/RSPEC-802)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-802)

### [Identifiers should not be longer than 31 characters](https://rules.sonarsource.com/cpp/RSPEC-799)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-799)

### [All uses of the #pragma directive should be documented](https://rules.sonarsource.com/cpp/RSPEC-793)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-793)

### [Assembly language should be encapsulated and isolated](https://rules.sonarsource.com/cpp/RSPEC-784)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-784)

### [Coroutines should not have too many lines of code](https://rules.sonarsource.com/cpp/RSPEC-6184)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-6184)

### [[[nodiscard]] should be used when the return value of a function should not be ignored](https://rules.sonarsource.com/cpp/RSPEC-6007)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-6007)

### [Functions that are not used in a project should be removed](https://rules.sonarsource.com/cpp/RSPEC-5536)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-5536)

### [Local variables should be initialized immediately](https://rules.sonarsource.com/cpp/RSPEC-5523)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-5523)

### [The order for arguments of the same type in a function call should be obvious](https://rules.sonarsource.com/cpp/RSPEC-5419)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-5419)

### [A cast should not convert a pointer type to an integral type](https://rules.sonarsource.com/cpp/RSPEC-5358)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-5358)

### [An object with integral type or pointer to void type shall not be converted to an object with pointer type](https://rules.sonarsource.com/cpp/RSPEC-5357)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-5357)

### [An object with pointer type shall not be converted to an unrelated pointer type, either directly or indirectly](https://rules.sonarsource.com/cpp/RSPEC-5356)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-5356)

### [Non-exception types should not be caught](https://rules.sonarsource.com/cpp/RSPEC-3698)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-3698)

### [Non-exception types should not be thrown](https://rules.sonarsource.com/cpp/RSPEC-3696)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-3696)

### [Binary operators should be overloaded as "friend" functions](https://rules.sonarsource.com/cpp/RSPEC-2807)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-2807)

### [Track parsing failures](https://rules.sonarsource.com/cpp/RSPEC-2260)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-2260)

### [Files should not be too complex](https://rules.sonarsource.com/cpp/RSPEC-1908)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-1908)

### [The ternary operator should not be used](https://rules.sonarsource.com/cpp/RSPEC-1774)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-1774)

### [A "struct" should not have member functions](https://rules.sonarsource.com/cpp/RSPEC-1771)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-1771)

### [Default parameters should not be defined](https://rules.sonarsource.com/cpp/RSPEC-1712)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-1712)

### [Exceptions should not be used](https://rules.sonarsource.com/cpp/RSPEC-1706)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-1706)

### [Rvalue references should not be used](https://rules.sonarsource.com/cpp/RSPEC-1704)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-1704)

### [Functions/methods should not have too many lines](https://rules.sonarsource.com/cpp/RSPEC-138)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-138)

### [Track uses of "NOSONAR" comments](https://rules.sonarsource.com/cpp/RSPEC-1291)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-1291)

### ["::" operator should be used to access global variables and functions](https://rules.sonarsource.com/cpp/RSPEC-1271)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-1271)

### ["for" loop stop conditions should be invariant](https://rules.sonarsource.com/cpp/RSPEC-127)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-127)

### [Statements should be on separate lines](https://rules.sonarsource.com/cpp/RSPEC-122)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-122)

### ["switch case" clauses should not have too many lines of code](https://rules.sonarsource.com/cpp/RSPEC-1151)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-1151)

### [Functions should not contain too many return statements](https://rules.sonarsource.com/cpp/RSPEC-1142)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-1142)

### [Magic numbers should not be used](https://rules.sonarsource.com/cpp/RSPEC-109)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-109)

### [Standard outputs should not be used directly to log anything](https://rules.sonarsource.com/cpp/RSPEC-106)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-106)

### [Files should not have too many lines of code](https://rules.sonarsource.com/cpp/RSPEC-104)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-104)

### [Lines should not be too long](https://rules.sonarsource.com/cpp/RSPEC-103)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-103)

### ["operator=" should check for assignment to self](https://rules.sonarsource.com/cpp/RSPEC-1250)

[Bug](https://rules.sonarsource.com/cpp/RSPEC-1250)

### [Accessible base classes should not be both "virtual" and non-virtual in the same hierarchy](https://rules.sonarsource.com/cpp/RSPEC-1013)

[Bug](https://rules.sonarsource.com/cpp/RSPEC-1013)

### [A variable which is not modified shall be const qualified](https://rules.sonarsource.com/cpp/RSPEC-994)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-994)

### [Preprocessor operators "#" and "##" should not be used](https://rules.sonarsource.com/cpp/RSPEC-968)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-968)

### [Switch statement conditions should not have essentially boolean type](https://rules.sonarsource.com/cpp/RSPEC-920)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-920)

### ["continue" should not be used](https://rules.sonarsource.com/cpp/RSPEC-909)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-909)

### [The loop-counter should be modified by one of: --, ++, -=n, or +=n; where n remains constant for the duration of the loop](https://rules.sonarsource.com/cpp/RSPEC-890)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-890)

### [Signed and unsigned types should not be mixed in expressions](https://rules.sonarsource.com/cpp/RSPEC-845)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-845)

### [typedefs that indicate size and signedness should be used in place of the basic types](https://rules.sonarsource.com/cpp/RSPEC-813)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-813)

### [The first operand of a conditional operator should have type bool](https://rules.sonarsource.com/cpp/RSPEC-812)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-812)

### [The condition of an if-statement and the condition of an iteration-statement shall have type bool](https://rules.sonarsource.com/cpp/RSPEC-811)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-811)

### [Appropriate char types should be used for character and integer values](https://rules.sonarsource.com/cpp/RSPEC-810)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-810)

### [Source code should only use /\* ... \*/ style comments](https://rules.sonarsource.com/cpp/RSPEC-787)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-787)

### [Concept names should comply with a naming convention](https://rules.sonarsource.com/cpp/RSPEC-6221)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-6221)

### [Coroutine names should comply with a naming convention](https://rules.sonarsource.com/cpp/RSPEC-6193)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-6193)

### ["std::cmp\_\*" functions should be used to compare signed and unsigned values](https://rules.sonarsource.com/cpp/RSPEC-6183)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-6183)

### ["nodiscard" attributes on functions should include explanations](https://rules.sonarsource.com/cpp/RSPEC-6166)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-6166)

### ["dynamic\_cast" should be used for downcasting](https://rules.sonarsource.com/cpp/RSPEC-5981)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-5981)

### [Struct should explicitly specify the access level when specifying base classes](https://rules.sonarsource.com/cpp/RSPEC-5966)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-5966)

### ["std::endl" should not be used](https://rules.sonarsource.com/cpp/RSPEC-5946)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-5946)

### [The identifiers used for the parameters in a re-declaration or override of a function shall be identical to those in the declaration](https://rules.sonarsource.com/cpp/RSPEC-5319)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-5319)

### [A loop-control-variable other than the loop-counter shall not be modified within condition or expression](https://rules.sonarsource.com/cpp/RSPEC-5316)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-5316)

### [The loop-counter shall not be modified within condition or statement](https://rules.sonarsource.com/cpp/RSPEC-5313)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-5313)

### [If loop-counter is not modified by -- or ++, then, within condition, the loop-counter shall only be used as an operand to <=, <, > or >=](https://rules.sonarsource.com/cpp/RSPEC-5312)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-5312)

### [A for loop shall contain a single loop-counter which shall not have floating type](https://rules.sonarsource.com/cpp/RSPEC-5311)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-5311)

### [Every switch statement shall have at least one case-clause](https://rules.sonarsource.com/cpp/RSPEC-5309)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-5309)

### [All "if ... else if" constructs shall be terminated with an "else "clause](https://rules.sonarsource.com/cpp/RSPEC-5307)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-5307)

### [An `if ( condition )` construct shall be followed by a compound statement. The else keyword shall be followed by either a compound statement, or another if statement](https://rules.sonarsource.com/cpp/RSPEC-5306)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-5306)

### [The statement forming the body of a "switch", "while", "do {...} while" or "for" statement shall be a compound statement](https://rules.sonarsource.com/cpp/RSPEC-5305)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-5305)

### [C-style casts (other than void casts) and functional notation casts (other than explicit constructor calls) shall not be used](https://rules.sonarsource.com/cpp/RSPEC-5303)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-5303)

### ["auto" should not be used to deduce raw pointers](https://rules.sonarsource.com/cpp/RSPEC-4334)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-4334)

### [Method overloads should be grouped together in the interface](https://rules.sonarsource.com/cpp/RSPEC-4136)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-4136)

### [GNU extensions should not be used](https://rules.sonarsource.com/cpp/RSPEC-3715)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-3715)

### [Raw string literals should be used](https://rules.sonarsource.com/cpp/RSPEC-3628)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-3628)

### ["inline" should not be used redundantly](https://rules.sonarsource.com/cpp/RSPEC-3549)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-3549)

### [Digit separators should be used](https://rules.sonarsource.com/cpp/RSPEC-3542)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-3542)

### [Base class access specifiers should not be redundant](https://rules.sonarsource.com/cpp/RSPEC-3540)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-3540)

### [Inheritance should be "public"](https://rules.sonarsource.com/cpp/RSPEC-3469)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-3469)

### [Methods should not return constants](https://rules.sonarsource.com/cpp/RSPEC-3400)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-3400)

### [Label names should comply with a naming convention](https://rules.sonarsource.com/cpp/RSPEC-3222)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-3222)

### [Enumeration values should comply with a naming convention](https://rules.sonarsource.com/cpp/RSPEC-2343)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-2343)

### [Enumeration names should comply with a naming convention](https://rules.sonarsource.com/cpp/RSPEC-2342)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-2342)

### [Namespace names should comply with a naming convention](https://rules.sonarsource.com/cpp/RSPEC-2304)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-2304)

### [Comment styles "//" and "/\* ... \*/" should not be mixed within a file](https://rules.sonarsource.com/cpp/RSPEC-1917)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-1917)

### ["union" names should comply with a naming convention](https://rules.sonarsource.com/cpp/RSPEC-1878)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-1878)

### ["public", "protected" and "private" sections of a class should be declared in that order](https://rules.sonarsource.com/cpp/RSPEC-1773)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-1773)

### [Constants should come first in equality tests](https://rules.sonarsource.com/cpp/RSPEC-1772)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-1772)

### [Type specifiers should be listed in a standard order](https://rules.sonarsource.com/cpp/RSPEC-1749)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-1749)

### [C++ comments should be used](https://rules.sonarsource.com/cpp/RSPEC-1708)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-1708)

### [Track "TODO" and "FIXME" comments that do not contain a reference to a person](https://rules.sonarsource.com/cpp/RSPEC-1707)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-1707)

### [The prefix increment/decrement form should be used](https://rules.sonarsource.com/cpp/RSPEC-1705)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-1705)

### ["struct" names should comply with a naming convention](https://rules.sonarsource.com/cpp/RSPEC-1642)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-1642)

### [File names should comply with a naming convention](https://rules.sonarsource.com/cpp/RSPEC-1578)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-1578)

### [Macro names should comply with a naming convention](https://rules.sonarsource.com/cpp/RSPEC-1543)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-1543)

### [Comments should not be located at the end of lines of code](https://rules.sonarsource.com/cpp/RSPEC-139)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-139)

### [Functions without parameters should not use "(void)"](https://rules.sonarsource.com/cpp/RSPEC-1270)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-1270)

### [break statements should not be used except for switch cases](https://rules.sonarsource.com/cpp/RSPEC-1227)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-1227)

### [Local variable and function parameter names should comply with a naming convention](https://rules.sonarsource.com/cpp/RSPEC-117)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-117)

### [Field names should comply with a naming convention](https://rules.sonarsource.com/cpp/RSPEC-116)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-116)

### [Lines should not end with trailing whitespaces](https://rules.sonarsource.com/cpp/RSPEC-1131)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-1131)

### [Files should contain an empty newline at the end](https://rules.sonarsource.com/cpp/RSPEC-113)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-113)

### [Tabulation characters should not be used](https://rules.sonarsource.com/cpp/RSPEC-105)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-105)

### [Class names should comply with a naming convention](https://rules.sonarsource.com/cpp/RSPEC-101)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-101)

### [A function should have a single point of exit at the end of the function](https://rules.sonarsource.com/cpp/RSPEC-1005)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-1005)

### ["using-directives" should not be used](https://rules.sonarsource.com/cpp/RSPEC-1001)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-1001)

### [Function names should comply with a naming convention](https://rules.sonarsource.com/cpp/RSPEC-100)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-100)

### [Track comments matching a regular expression](https://rules.sonarsource.com/cpp/RSPEC-5639)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-5639)

### [Track instances of the "#error" preprocessor directive being reached](https://rules.sonarsource.com/cpp/RSPEC-1914)

[Code Smell](https://rules.sonarsource.com/cpp/RSPEC-1914)