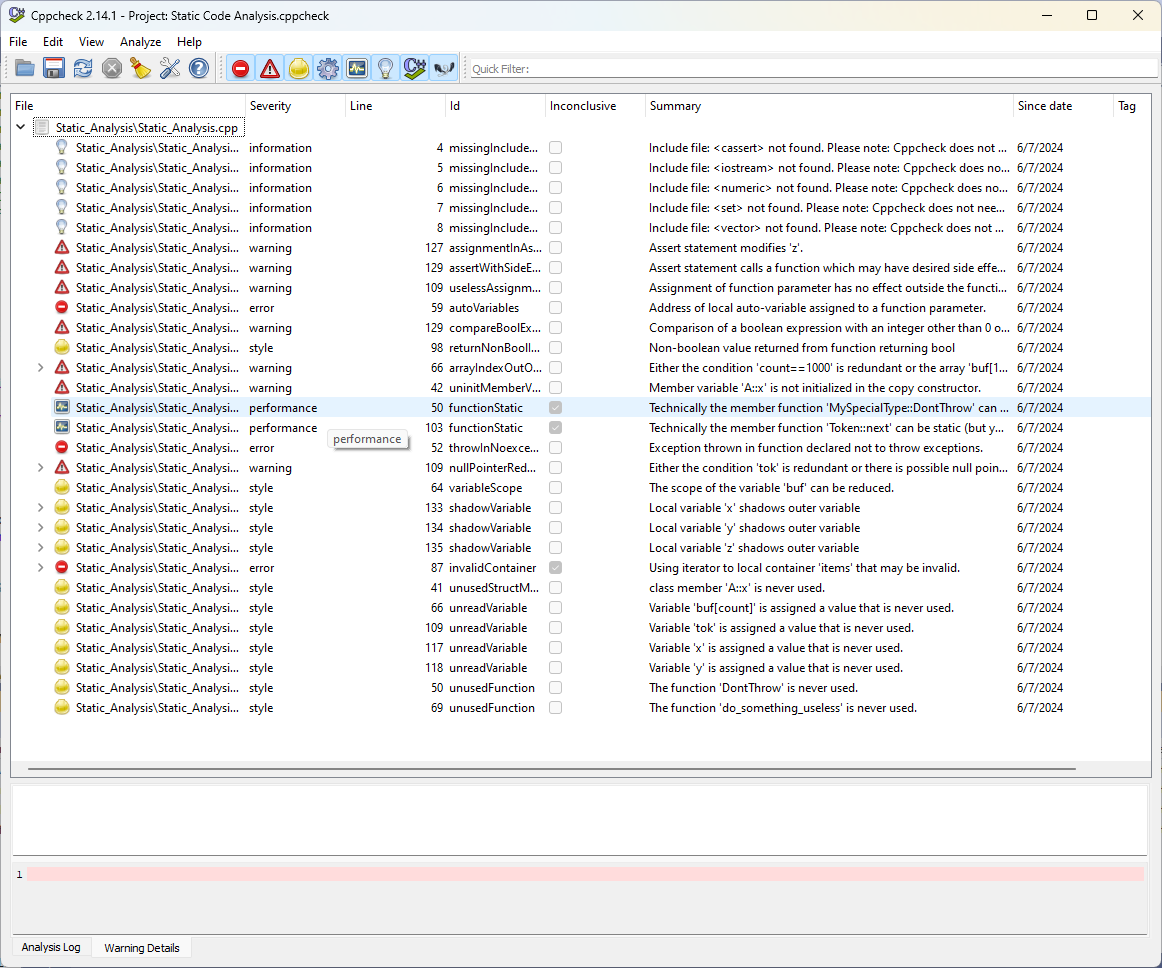
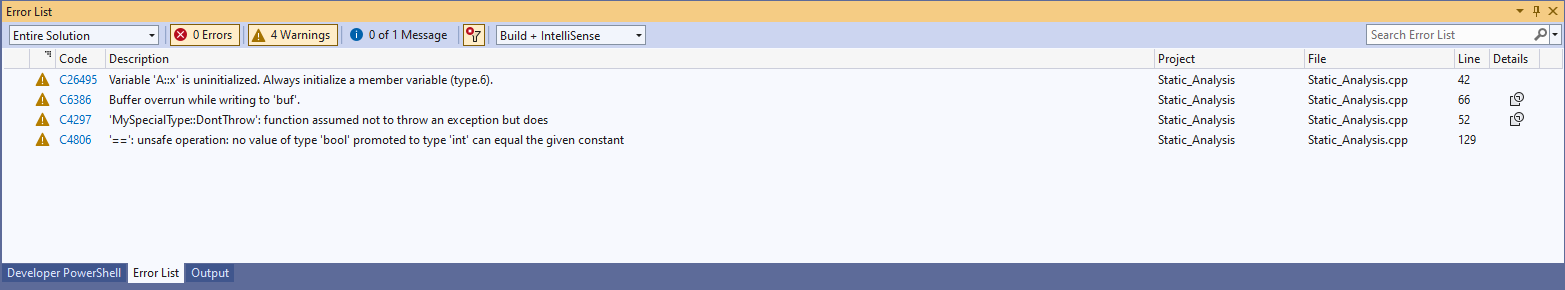
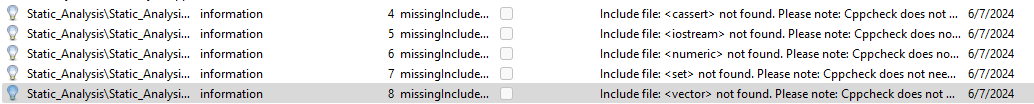
Brian Engel

Module 5 Activity Static Code Analysis



Not Risk. These are information messages that cppcheck put out for all the include lines. It seems that cppcheck doesn’t need to include standard library headers in order to work correctly.

Risk. Changes value of z instead of comparing value of z in the assert statement.

Risk. Uses a function in assert that changes the value of a variable.

Risk. It doesn’t really do anything except get to the end of linked list and then return 0. Should probably return the end of the linked list.

Risk. Points to a local variable that is freed once the function ends.

Risk. This popped up in both cppcheck and the compiler. my\_function returns a bool (0 or 1) and will never be 3. It appears the function was written with the wrong return type.

Risk. Tries to return a non boolean value.

Risk. This popped up in both cppcheck and the compiler. Tries to write to an array out of bounds.

Risk. This popped up in both cppcheck and the compiler. The constructor needs to initialize variables in the class if they aren’t already initialized.

Not Risk. Can make function static since it isn’t attached to any instances. Can boost performance.

Not Risk. Can make function static since it isn’t attached to any instances. Can boost performance.

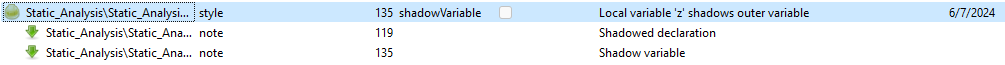
Risk. This popped up in both cppcheck and the compiler. Throwing an exception from a function declared to not throw exceptions can cause unexpected behavior.

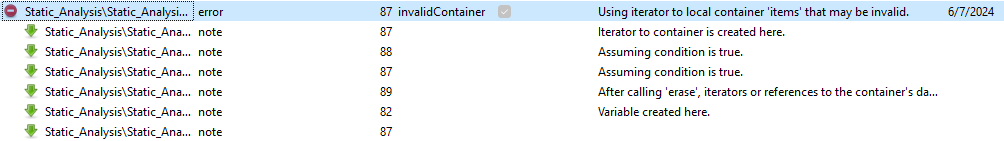
Risk. There needs to be a check that tok is not null before dereferencing it.

Not Risk. It is saying the array can be put in the if loop, but this might have unintended consequences if more is added to this function.

Risk. Can lead to unintended consequences. Variable defined in outer scope and then redefined in inner scope will go back to value in outer scope once the inner scope has finished.

Risk. Can lead to unintended consequences. Variable defined in outer scope and then redefined in inner scope will go back to value in outer scope once the inner scope has finished.

Risk. Can lead to unintended consequences. Variable defined in outer scope and then redefined in inner scope will go back to value in outer scope once the inner scope has finished.

Risk. This has to do with possibly making the iterator invalid by erasing the element before the iterator is updated.

Not Risk. Let us know the variable x in structure a is not used in the code.

Not Risk. Let us know buff[count] is not used after it is assigned.

Not Risk. Let us know tok is not used after it is assigned.

Not Risk. Let us know x is not used after it is assigned.

Not Risk. Let us know y is not used after it is assigned.

Not Risk. Let us know function DontThrow is not used.

Not Risk. Let us know function do\_something\_useless is not used.