Below is a screenshot of the issues Visual Studio had with the code in this project. There were only three issues found by Visual Studio. This was very surprising to me as the code successfully compiled and ran with no errors. Below is a table of all the issues shown by the analyzer software.

Warning Message Style Error

|  |  |  |  |
| --- | --- | --- | --- |
| **Severity** | **Description/Risk** | **Mitigation** | **Analyzer** |
|  | Buffer overrun, indicates that a writable extent of the specified buffer might be smaller than the index used to write to it. | Make sure the index stays in bounds | VS |
|  | Member variable isn’t initialized by a constructor or by an initializer | Make sure all variables are initialized by the end of a constructor. Add in-class initialization of all member variables | VS, CPP |
|  | Local variable not initialized when it is declared | Declare all local variables when they are initialized | VS |
|  | After assignment, the variable is either assigned another value or goes out of scope. It is possible the unused variable points out a bug | Remove all unused variables from the code | CPP |
|  | Function returns the address of a stack variable, which will cause unintended program behavior, typically in the form of a crash | When pointing to variables make sure to not assign an auto-variable as a function parameter | CPP |
|  | A function is called inside of an assert statement in release code | Functions inside assert statements should be removed as they will not be called in a release build | CPP |
|  | Assignment of function parameter has no effect outside the function. This is a bad code quality | Do not assign parameters that will not be used | CPP |
|  | Unsafe operation: No value of type ‘type’ promoted to type ‘type’ can equal the given constant. Promotion causes bool to be promoted to int. It can never be true | When comparing two types take care to make sure they can be compared and/or of the same type. | CPP |
|  | Access of Memory Location After End of Buffer, reads or writes to a buffer using an index or pointer that references a memory location at the end of a buffer. Occurs when a pointer or its index is incremented to a position after the buffer. | Double check the buffer is as large as specified. If accessing buffer in a loop, make sure there is no danger of writing past the allocated space | CPP |
|  | Function assumed not to throw an exception but does. Function declaration contains *noexcept* specifier, an empty throw exception. | Do not attempt to throw exceptions in functions that are declared noexcept or throw(). Remove the noexcept or throw | CPP |
|  | Improper control of a resource through its lifetime. Resources often have explicit instructions on how to be created, used and destroyed. When code does not follow these instructions, it can lead to unexpected behavior and/or potential exploitable state. Using iterator to local container “items” that be invalid | Proper use of resources must be followed according to guidelines. | CPP |