# 

# CS 405 5-3 Activity: Static Code Analysis

Raymond Aponte

raymond.aponte@snhu.edu

Southern New Hampshire University

**Process Summary**

All unique messages were discovered by Cppcheck. All messages discovered by Visual Studio were also discovered by Cppcheck.

MESSAGE: assignmentInAssert on line 127. CWE: 398 indicates poor code quality.

* + RISK: Not risk
  + SYSTEM: Cppcheck
  + DESCRIPTION: Variable 'z' is modified inside assert statement. Assert statements are removed from release builds so the code inside assert statement is not executed. If the code is needed also in release builds, this is a bug.
  + MITIGATION: If necessary, ‘z’ should be modified outside of the assert statement.

1. MESSAGE: uselessAssignmentPtrArg on line 109. CWE: 398 indicates poor code quality.
   * RISK: Not risk
   * SYSTEM: Cppcheck
   * DESCRIPTION: Assignment of function parameter has no effect outside the function.
   * MITIGATION: The function parameter may require dereferencing.
2. MESSAGE: autoVariables on line 59. CWE: 562 indicates return of stack variable address.
   * RISK: Risk
   * SYSTEM: Cppcheck
   * DESCRIPTION: Dangerous assignment - the function parameter is assigned the address of a local auto-variable. Local auto-variables are reserved from the stack which is freed when the function ends. So the pointer to a local variable is invalid after the function ends. The program may crash next time the pointer is dereferenced.
   * MITIGATION: Do not return the address of a stack variable. Static analysis tools such as Cppcheck can help spot these instances.
3. MESSAGE: returnNonBoolInBooleanFunction on line 98.
   * RISK: Not risk
   * SYSTEM: Cppcheck
   * DESCRIPTION: Non-boolean value returned from function returning bool.
   * MITIGATION: The data type of the function should be changed or the correct value should be returned.
4. MESSAGE: functionStatic on lines 50 and 103. CWE: 398 indicates poor code quality.
   * RISK: Not risk
   * SYSTEM: Cppcheck
   * DESCRIPTION: The member functions 'MySpecialType::DontThrow' and 'Token::next' can be made into static functions.
   * MITIGATION: Making a function static can bring a performance benefit since no 'this' instance is passed to the function.
5. MESSAGE: nullPointerRedundantCheck on line 109. CWE: 476 indicates NULL pointer dereference.
   * RISK: Risk
   * SYSTEM: Cppcheck
   * DESCRIPTION: Either the condition 'tok' is redundant or there is possible null pointer dereference: tok. This may cause a crash or exit.
   * MITIGATION: Initializing all variables/data stores, implementing input validation to ensure expected values, and sanity-checking all pointers can prevent this type of problem.
6. MESSAGE: shadowVariable on lines 133, 134, and 135. CWE: 398 indicates poor code quality.
   * RISK: Not risk
   * SYSTEM: Cppcheck
   * DESCRIPTION: Local variable 'x' shadows outer variable. The same is stated for variables ‘y’ and ‘z’.
   * MITIGATION: Do not use the same names for different variables.
7. MESSAGE: invalidContainer on line 87. CWE: 664 indicates improper control of a resource through its lifetime.
   * RISK: Risk
   * SYSTEM: Cppcheck
   * DESCRIPTION: Using iterator to local container 'items' that may be invalid. Unexplainable behavior/exploitable states can occur.
   * MITIGATION: If trying to iterate trough a vector, set the iterator to 0 and iterate to the end of the vector, incrementing the iterator each time.
8. MESSAGE: unreadVariable on lines 66, 109, 117, and 118. CWE: 563 indicates assignment to variable without use.
   * RISK: Not risk
   * SYSTEM: Cppcheck
   * DESCRIPTION: Variables ‘buf[count]’, ‘tok’, ‘x’, and ‘y’ are assigned values that are never used. This is poor code quality that could lead to bugs or other weaknesses.
   * MITIGATION: Unused variables should be removed.
9. MESSAGE: assertWithSideEffect on line 129. CWE: 398 indicates poor code quality.
   * RISK: Not risk
   * SYSTEM: Cppcheck
   * DESCRIPTION: Non-pure function: 'my\_function' is called inside assert statement. Assert statements are removed from release builds so the code inside assert statement is not executed. If the code is needed also in release builds, this is a bug.
   * MITIGATION: Assert statements should be removed from the release build.