Ankush Singh

**WorkShop_8**

Post Lab Experience Screenshots

```
C:\Users\singh\WorkShop_8>cppcheck -h
Cppcheck - A tool for static C/C++ code analysis

Syntax:
    cppcheck [OPTIONS] [files or paths]

If a directory is given instead of a filename, *.cpp, *.cxx, *.cc, *.c++, *.c,
*.ipp,
*.ixx, *.tpp, and *.txx files are checked recursively from the given directory.

Options:
    --addon=<addon>
                         Execute addon. i.e. --addon=misra. If options must be
                         provided a json configuration is needed.
    --addon-python=<python interpreter>
                         You can specify the python interpreter either in the
                         addon json files or through this command line option.
                         If not present, Cppcheck will try "python3" first and
                         then "python".
    --cppcheck-build-dir=<dir>
                         Cppcheck work folder. Advantages:
                          * whole program analysis
                          * faster analysis; Cppcheck will reuse the results if
                            the hash for a file is unchanged.
                          * some useful debug information, i.e. commands used t
o
                            execute clang/clang-tidy/addons.
    --check-config       Check cppcheck configuration. The normal code
                         analysis is disabled by this flag.
    --check-level=<level>
                         Configure how much checking you want:
                          * normal: Cppcheck uses some compromises in the check
ing so
                            the checking will finish in reasonable time.
                          * exhaustive: deeper analysis that you choose when yo
u can
                            wait.
                         The default choice is 'normal'.
    --check-library      Show information messages when library files have
                         incomplete info.
```

```
                           {column}    column number
                           {info}      location info
                           {code}      show the real code
                           \t         insert tab
                           \n         insert newline
                           \r         insert carriage return
                         Example format (gcc-like):
                         '{file}:{line}:{column}: note: {info}\n{code}'
    -U<ID>               Undefine preprocessor symbol. Use -U to explicitly
                         hide certain #ifdef <ID> code paths from checking.
                         Example: '-UDEBUG'
    -v, --verbose        Output more detailed error information.
                         Note that this option is not mutually exclusive with -
-quiet.
    --version            Print out version number.
    --xml                Write results in xml format to error stream (stderr).

Example usage:
  # Recursively check the current folder. Print the progress on the screen and
  # write errors to a file:
  cppcheck . 2> err.txt

  # Recursively check ../myproject/ and don't print progress:
  cppcheck --quiet ../myproject/

  # Check test.cpp, enable all checks:
  cppcheck --enable=all --inconclusive --library=posix test.cpp

  # Check f.cpp and search include files from inc1/ and inc2/:
  cppcheck -I inc1/ -I inc2/ f.cpp

For more information:
    https://files.cppchecksolutions.com/manual.pdf

Many thanks to the 3rd party libraries we use:
 * tinyxml2 -- loading project/library/ctu files.
 * picojson -- loading compile database.
 * pcre -- rules.
 * qt -- used in GUI

C:\Users\singh\WorkShop_8>
```

Above screenshot, making sure that the tool is installed and displaying the information for how user can use this tool.

Ankush Singh

**WorkShop_8**

Post Lab Experience Screenshots

```
C:\Users\singh\WorkShop_8>git clone https://github.com/dpilger26/NumCpp.git
Cloning into 'NumCpp'...
remote: Enumerating objects: 85425, done.
remote: Counting objects: 100% (6389/6389), done.
remote: Compressing objects: 100% (942/942), done.
remote: Total 85425 (delta 5476), reused 6268 (delta 5423), pack-reused 79036 (from 1)
Receiving objects: 100% (85425/85425), 116.79 MiB | 8.85 MiB/s, done.
Resolving deltas: 100% (73596/73596), done.
Updating files: 100% (3022/3022), done.
```

Above screenshot, cloned the NumCpp using command 'git clone https://github.com/dpilger26/NumCpp.git'

```
C:\Users\singh\WorkShop_8\NumCpp>cd .git/hooks

C:\Users\singh\WorkShop_8\NumCpp\.git\hooks>dir
 Volume in drive C is Windows
 Volume Serial Number is CCCE-BF85

 Directory of C:\Users\singh\WorkShop_8\NumCpp\.git\hooks

10/20/2024  10:42 AM    <DIR>          .
10/20/2024  10:42 AM               478 applypatch-msg.sample
10/20/2024  10:42 AM               896 commit-msg.sample
10/20/2024  10:42 AM             4,726 fsmonitor-watchman.sample
10/20/2024  10:42 AM               189 post-update.sample
10/20/2024  10:42 AM               424 pre-applypatch.sample
10/20/2024  10:42 AM             1,643 pre-commit.sample
10/20/2024  10:42 AM               416 pre-merge-commit.sample
10/20/2024  10:42 AM             1,374 pre-push.sample
10/20/2024  10:42 AM             4,898 pre-rebase.sample
10/20/2024  10:42 AM               544 pre-receive.sample
10/20/2024  10:42 AM             1,492 prepare-commit-msg.sample
10/20/2024  10:42 AM             2,783 push-to-checkout.sample
10/20/2024  10:42 AM             2,308 sendemail-validate.sample
10/20/2024  10:42 AM             3,650 update.sample
              14 File(s)         25,821 bytes
               1 Dir(s)  167,585,443,840 bytes free

C:\Users\singh\WorkShop_8\NumCpp\.git\hooks>
```

Above Screenshot, going in to the .git/hooks and making sure that we have the files in the directory.

Ankush Singh

**WorkShop_8**

Post Lab Experience Screenshots

```
C:\Users\singh\WorkShop_8\NumCpp\.git\hooks>copy pre-commit.sample pre-commit
        1 file(s) copied.

C:\Users\singh\WorkShop_8\NumCpp\.git\hooks>dir
 Volume in drive C is Windows
 Volume Serial Number is CCCE-BF85

 Directory of C:\Users\singh\WorkShop_8\NumCpp\.git\hooks

10/20/2024  10:54 AM    <DIR>          .
10/20/2024  10:42 AM               478 applypatch-msg.sample
10/20/2024  10:42 AM               896 commit-msg.sample
10/20/2024  10:42 AM             4,726 fsmonitor-watchman.sample
10/20/2024  10:42 AM               189 post-update.sample
10/20/2024  10:42 AM               424 pre-applypatch.sample
10/20/2024  10:42 AM             1,643 pre-commit
10/20/2024  10:42 AM             1,643 pre-commit.sample
10/20/2024  10:42 AM               416 pre-merge-commit.sample
10/20/2024  10:42 AM             1,374 pre-push.sample
10/20/2024  10:42 AM             4,898 pre-rebase.sample
10/20/2024  10:42 AM               544 pre-receive.sample
10/20/2024  10:42 AM             1,492 prepare-commit-msg.sample
10/20/2024  10:42 AM             2,783 push-to-checkout.sample
10/20/2024  10:42 AM             2,308 sendemail-validate.sample
10/20/2024  10:42 AM             3,650 update.sample
              15 File(s)         27,464 bytes
               1 Dir(s)  167,580,680,192 bytes free

C:\Users\singh\WorkShop_8\NumCpp\.git\hooks>
```

Above screenshot, used the command ‘copy pre-commit.sample pre-commit’ and in the screenshot we have file name ‘pre-commit’

Ankush Singh

**WorkShop_8**

Post Lab Experience Screenshots

```
C:\Users\singh\WorkShop_8\NumCpp\test>cd pytest

C:\Users\singh\WorkShop_8\NumCpp\test\pytest>cd src

C:\Users\singh\WorkShop_8\NumCpp\test\pytest\src>dir
 Volume in drive C is Windows
 Volume Serial Number is CCCE-BF85

 Directory of C:\Users\singh\WorkShop_8\NumCpp\test\pytest\src

10/20/2024  10:42 AM    <DIR>          .
10/20/2024  10:42 AM    <DIR>          ..
10/20/2024  10:42 AM             2,433 BindingsIncludes.hpp
10/20/2024  10:42 AM             2,224 CMakeLists.txt
10/20/2024  10:42 AM             2,454 Constants.cpp
10/20/2024  10:42 AM            19,712 Coordinates.cpp
10/20/2024  10:42 AM             4,799 Core.cpp
10/20/2024  10:42 AM             9,145 DataCube.cpp
10/20/2024  10:42 AM             3,290 DateTime.cpp
10/20/2024  10:42 AM             2,021 Filter.cpp
10/20/2024  10:42 AM           157,417 Functions.cpp
10/20/2024  10:42 AM             3,050 ImageProcessing.cpp
10/20/2024  10:42 AM             1,776 Integrate.cpp
10/20/2024  10:42 AM             2,472 Linalg.cpp
10/20/2024  10:42 AM               163 Logger.cpp
10/20/2024  10:42 AM           167,986 NdArray.cpp
10/20/2024  10:42 AM             1,465 NumCppPy.cpp
10/20/2024  10:42 AM             9,725 Polynomial.cpp
10/20/2024  10:42 AM    <DIR>          pybind11
10/20/2024  10:42 AM            31,096 Random.cpp
10/20/2024  10:42 AM             2,123 Roots.cpp
10/20/2024  10:42 AM             9,849 Rotations.cpp
10/20/2024  10:42 AM            33,514 Special.cpp
10/20/2024  10:42 AM             1,720 Utils.cpp
10/20/2024  10:42 AM            10,383 Vector.cpp
              22 File(s)        478,817 bytes
               3 Dir(s)  167,523,143,680 bytes free
```

```
C:\Users\singh\WorkShop_8\NumCpp>git status
On branch master
Your branch is ahead of 'origin/master' by 3 commits.
  (use "git push" to publish your local commits)

Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   test/pytest/src/Vector.cpp

no changes added to commit (use "git add" and/or "git commit -a")

C:\Users\singh\WorkShop_8\NumCpp>
```

Above Screenshots, we can see there are .cpp files and another screenshot we can see the modified any .cpp file.

```
C:\Users\singh\WorkShop_8\NumCpp>git add test/pytest/src/Vector.cpp

C:\Users\singh\WorkShop_8\NumCpp>git commit -m 'minor changes'
error: pathspec 'changes'' did not match any file(s) known to git

C:\Users\singh\WorkShop_8\NumCpp>git commit -m "minor changes"
============================================
Hello World
============================================
test/pytest/src/Vector.cpp:2: trailing whitespace.
+// Adding a Comment (modified any .cpp file (Vector.cpp))

C:\Users\singh\WorkShop_8\NumCpp>
```

Above screenshot, we can see that the hook file is successfully connected with git.

Ankush Singh

**WorkShop_8**

Post Lab Experience Screenshots

```
C:\Users\singh\WorkShop_8\NumCpp>git commit -m "minor changes"
===============================================
Hello World
===============================================
Cppcheck - A tool for static C/C++ code analysis

Syntax:
    cppcheck [OPTIONS] [files or paths]

If a directory is given instead of a filename, *.cpp, *.cxx, *.cc, *.c++, *.c, *.ipp,
*.ixx, *.tpp, and *.txx files are checked recursively from the given directory.

Options:
    --addon=<addon>
                         Execute addon. i.e. --addon=misra. If options must be
                         provided a json configuration is needed.
    --addon-python=<python interpreter>
                         You can specify the python interpreter either in the
                         addon json files or through this command line option.
                         If not present, Cppcheck will try "python3" first and
                         then "python".
    --cppcheck-build-dir=<dir>
                         Cppcheck work folder. Advantages:
                          * whole program analysis
                          * faster analysis; Cppcheck will reuse the results if
                            the hash for a file is unchanged.
                          * some useful debug information, i.e. commands used to
                            execute clang/clang-tidy/addons.
    --check-config       Check cppcheck configuration. The normal code
                         analysis is disabled by this flag.
    --check-level=<level>
                         Configure how much checking you want:
                          * normal: Cppcheck uses some compromises in the checking so
                            the checking will finish in reasonable time.
                          * exhaustive: deeper analysis that you choose when you can
                            wait.
                         The default choice is 'normal'.
    --check-library      Show information messages when library files have
                         incomplete info.
    --checkers-report=<file>
                         Write a report of all the active checkers to the given file.
    --clang=<path>       Experimental: Use Clang parser instead of the builtin Cppcheck
                         parser. Takes the executable as optional parameter and
                         defaults to `clang`. Cppcheck will run the given Clang
                         executable, import the Clang AST and convert it into
                         Cppcheck data. After that the normal Cppcheck analysis is
                         used. You must have the executable in PATH if no path is
                         given.
    --config-exclude=<dir>
                         Path (prefix) to be excluded from configuration
                         checking. Preprocessor configurations defined in
                         headers (but not sources) matching the prefix will not
                         be considered for evaluation.
    --config-excludes-file=<file>
                         A file that contains a list of config-excludes
    --disable=<id>       Disable individual checks.
                         Please refer to the documentation of --enable=<id>
                         for further details.
    --dump               Dump xml data for each translation unit. The dump
                         files have the extension .dump and contain ast,
```

Ankush Singh

**WorkShop_8**

Post Lab Experience Screenshots

```
    --dump                Dump xml data for each translation unit. The dump
                          files have the extension .dump and contain ast,
                          tokenlist, symboldatabase, valueflow.
    -D<ID>                Define preprocessor symbol. Unless --max-configs or
                          --force is used, Cppcheck will only check the given
                          configuration when -D is used.
                          Example: '-DDEBUG=1 -D__cplusplus'.
    -E                    Print preprocessor output on stdout and don't do any
                          further processing.
    --enable=<id>         Enable additional checks. The available ids are:
                           * all
                                   Enable all checks. It is recommended to only
                                   use --enable=all when the whole program is
                                   scanned, because this enables unusedFunction.
                           * warning
                                   Enable warning messages
                           * style
                                   Enable all coding style checks. All messages
                                   with the severities 'style', 'warning',
                                   'performance' and 'portability' are enabled.
                           * performance
                                   Enable performance messages
                           * portability
                                   Enable portability messages
                           * information
                                   Enable information messages
                           * unusedFunction
                                   Check for unused functions. It is recommended
                                   to only enable this when the whole program is
                                   scanned.
                           * missingInclude
                                   Warn if there are missing includes.
                          Several ids can be given if you separate them with
                          commas. See also --std
    --error-exitcode=<n> If errors are found, integer [n] is returned instead of
                          the default '0'. '1' is returned
                          if arguments are not valid or if no input files are
                          provided. Note that your operating system can modify
                          this value, e.g. '256' can become '0'.
    --errorlist           Print a list of all the error messages in XML format.
    --exitcode-suppressions=<file>
                          Used when certain messages should be displayed but
                          should not cause a non-zero exitcode.
    --file-filter=<str>   Analyze only those files matching the given filter str
                          Can be used multiple times
                          Example: --file-filter=*bar.cpp analyzes only files
                                   that end with bar.cpp.
    --file-list=<file>    Specify the files to check in a text file. Add one
                          filename per line. When file is '-,' the file list will
                          be read from standard input.
    -f, --force           Force checking of all configurations in files. If used
                          together with '--max-configs=', the last option is the
                          one that is effective.
    -h, --help            Print this help.
    -I <dir>              Give path to search for include files. Give several -I
                          parameters to give several paths. First given path is
                          searched for contained header files first. If paths are
                          relative to source files, this is not needed.
    --includes-file=<file>
                          Specify directory paths to search for included header
                          files in a text file. Add one include path per line.
```

Ankush Singh

**WorkShop_8**

Post Lab Experience Screenshots

```
    --includes-file=<file>
                        Specify directory paths to search for included header
                        files in a text file. Add one include path per line.
                        First given path is searched for contained header
                        files first. If paths are relative to source files,
                        this is not needed.
    --include=<file>
                        Force inclusion of a file before the checked file.
    -i <dir or file>    Give a source file or source file directory to exclude
                        from the check. This applies only to source files so
                        header files included by source files are not matched.
                        Directory name is matched to all parts of the path.
    --inconclusive      Allow that Cppcheck reports even though the analysis is
                        inconclusive.
                        There are false positives with this option. Each result
                        must be carefully investigated before you know if it is
                        good or bad.
    --inline-suppr      Enable inline suppressions. Use them by placing one or
                        more comments, like: '// cppcheck-suppress warningId'
                        on the lines before the warning to suppress.
    -j <jobs>           Start <jobs> threads to do the checking simultaneously.
    --language=<language>, -x <language>
                        Forces cppcheck to check all files as the given
                        language. Valid values are: c, c++
    --library=<cfg>     Load file <cfg> that contains information about types
                        and functions. With such information Cppcheck
                        understands your code better and therefore you
                        get better results. The std.cfg file that is
                        distributed with Cppcheck is loaded automatically.
                        For more information about library files, read the
                        manual.
    --max-configs=<limit>
                        Maximum number of configurations to check in a file
                        before skipping it. Default is '12'. If used together
                        with '--force', the last option is the one that is
                        effective.
    --max-ctu-depth=N   Max depth in whole program analysis. The default value
                        is 2. A larger value will mean more errors can be found
                        but also means the analysis will be slower.
    --output-file=<file> Write results to file, rather than standard error.
    --platform=<type>, --platform=<file>
                        Specifies platform specific types and sizes. The
                        available builtin platforms are:
                         * unix32
                                32 bit unix variant
                         * unix64
                                64 bit unix variant
                         * win32A
                                32 bit Windows ASCII character encoding
                         * win32W
                                32 bit Windows UNICODE character encoding
                         * win64
                                64 bit Windows
                         * avr8
                                8 bit AVR microcontrollers
                         * elbrus-e1cp
                                Elbrus e1c+ architecture
                         * pic8
                                8 bit PIC microcontrollers
                                Baseline and mid-range architectures
                         * pic8-enhanced
```

Ankush Singh

**WorkShop_8**

Post Lab Experience Screenshots

```
                         * pic8-enhanced
                                8 bit PIC microcontrollers
                                Enhanced mid-range and high end (PIC18) architectures
                         * pic16
                                16 bit PIC microcontrollers
                         * mips32
                                32 bit MIPS microcontrollers
                         * native
                                Type sizes of host system are assumed, but no
                                further assumptions.
                         * unspecified
                                Unknown type sizes
    --plist-output=<path>
                        Generate Clang-plist output files in folder.
    --project=<file>    Run Cppcheck on project. The <file> can be a Visual
                        Studio Solution (*.sln), Visual Studio Project
                        (*.vcxproj), compile database (compile_commands.json),
                        or Borland C++ Builder 6 (*.bpr). The files to analyse,
                        include paths, defines, platform and undefines in
                        the specified file will be used.
    --project-configuration=<config>
                        If used together with a Visual Studio Solution (*.sln)
                        or Visual Studio Project (*.vcxproj) you can limit
                        the configuration cppcheck should check.
                        For example: '--project-configuration=Release|Win32'
    -q, --quiet         Do not show progress reports.
                        Note that this option is not mutually exclusive with --verbose.
    -rp=<paths>, --relative-paths=<paths>
                        Use relative paths in output. When given, <paths> are
                        used as base. You can separate multiple paths by ';'.
                        Otherwise path where source files are searched is used.
                        We use string comparison to create relative paths, so
                        using e.g. ~ for home folder does not work. It is
                        currently only possible to apply the base paths to
                        files that are on a lower level in the directory tree.
    --report-progress   Report progress messages while checking a file (single job only).
    --rule=<rule>       Match regular expression.
    --rule-file=<file>  Use given rule file. For more information, see:
                        http://sourceforge.net/projects/cppcheck/files/Articles/
    --std=<id>          Set standard.
                        The available options are:
                         * c89
                                C code is C89 compatible
                         * c99
                                C code is C99 compatible
                         * c11
                                C code is C11 compatible (default)
                         * c++03
                                C++ code is C++03 compatible
                         * c++11
                                C++ code is C++11 compatible
                         * c++14
                                C++ code is C++14 compatible
                         * c++17
                                C++ code is C++17 compatible
                         * c++20
                                C++ code is C++20 compatible (default)
    --suppress=<spec>   Suppress warnings that match <spec>. The format of
                        <spec> is:
                        [error id]:[filename]:[line]
                        The [filename] and [line] are optional. If [error id]
```

Ankush Singh

# WorkShop_8

Post Lab Experience Screenshots

```
    --suppress=<spec>    Suppress warnings that match <spec>. The format of
                         <spec> is:
                         [error id]:[filename]:[line]
                         The [filename] and [line] are optional. If [error id]
                         is a wildcard '*', all error ids match.
    --suppressions-list=<file>
                         Suppress warnings listed in the file. Each suppression
                         is in the same format as <spec> above.
    --suppress-xml=<file>
                         Suppress warnings listed in a xml file. XML file should
                         follow the manual.pdf format specified in section.
                         `6.4 XML suppressions` .
    --template='<text>'  Format the error messages. Available fields:
                           {file}              file name
                           {line}              line number
                           {column}            column number
                           {callstack}         show a callstack. Example:
                                                 [file.c:1] -> [file.c:100]
                           {inconclusive:text} if warning is inconclusive, text
                                               is written
                           {severity}          severity
                           {message}           warning message
                           {id}                warning id
                           {cwe}               CWE id (Common Weakness Enumeration)
                           {code}              show the real code
                           \t                 insert tab
                           \n                 insert newline
                           \r                 insert carriage return
                         Example formats:
                         '{file}:{line},{severity},{id},{message}' or
                         '{file}({line}):({severity}) {message}' or
                         '{callstack} {message}'
                         Pre-defined templates: gcc (default), cppcheck1 (old default), vs, edit.
    --template-location='<text>'
                         Format error message location. If this is not provided
                         then no extra location info is shown.
                         Available fields:
                           {file}      file name
                           {line}      line number
                           {column}    column number
                           {info}      location info
                           {code}      show the real code
                           \t         insert tab
                           \n         insert newline
                           \r         insert carriage return
                         Example format (gcc-like):
                         '{file}:{line}:{column}: note: {info}\n{code}'
    -U<ID>               Undefine preprocessor symbol. Use -U to explicitly
                         hide certain #ifdef <ID> code paths from checking.
                         Example: '-UDEBUG'
    -v, --verbose        Output more detailed error information.
                         Note that this option is not mutually exclusive with --quiet.
    --version            Print out version number.
    --xml                Write results in xml format to error stream (stderr).

Example usage:
  # Recursively check the current folder. Print the progress on the screen and
  # write errors to a file:
  cppcheck . 2> err.txt

  # Recursively check ../myproject/ and don't print progress:
  cppcheck --quiet ../myproject/

  # Check test.cpp, enable all checks:
  cppcheck --enable=all --inconclusive --library=posix test.cpp

  # Check f.cpp and search include files from inc1/ and inc2/:
  cppcheck -I inc1/ -I inc2/ f.cpp

For more information:
    https://files.cppchecksolutions.com/manual.pdf

Many thanks to the 3rd party libraries we use:
 * tinyxml2 -- loading project/library/ctu files.
 * picojson -- loading compile database.
 * pcre -- rules.
 * qt -- used in GUI
===========================================
test/pytest/src/Vector.cpp:2: trailing whitespace.
+// Adding a Comment (modified any .cpp file (Vector.cpp))

C:\Users\singh\WorkShop_8\NumCpp>
```

Above screenshots, Displaying the output after running cppcheck -h in pre-commit file.

Ankush Singh

**WorkShop_8**

Post Lab Experience Screenshots

$ pre-commit ×

C: > Users > singh > WorkShop_8 > NumCpp > .git > hooks > $ pre-commit

```
#!/bin/sh
#
# An example hook script to verify what is about to be committed.
# Called by "git commit" with no arguments.  The hook should
# exit with non-zero status after issuing an appropriate message if
# it wants to stop the commit.
#
# To enable this hook, rename this file to "pre-commit".

echo "============================================"
echo "Started static analysis tool for NumCPP!"
echo "============================================"
cppcheck --platform=win64 "C:\Users\singh\WorkShop_8\NumCpp"
echo "============================================"
echo "Finished the code scanning with static analysis tool"
echo "============================================"
```

Above screenshot, editing the pre-commit file to start the static analysis for NumCPP.

Ankush Singh

# WorkShop_8

Post Lab Experience Screenshots

```
C:\Users\singh\WorkShop_8\NumCpp>git commit -m "minor changes"
===============================================
Started static analysis tool for NumCPP!
===============================================
Checking C:\Users\singh\WorkShop_8\NumCpp\develop\main.cpp ...
1/30 files checked 0% done
Checking C:\Users\singh\WorkShop_8\NumCpp\examples\GaussNewtonNlls\GaussNewtonNlls.cpp ...
2/30 files checked 3% done
Checking C:\Users\singh\WorkShop_8\NumCpp\examples\InterfaceWithEigen\InterfaceWithEigen.cpp ...
3/30 files checked 3% done
Checking C:\Users\singh\WorkShop_8\NumCpp\examples\InterfaceWithOpenCV\InterfaceWithOpenCV.cpp ...
4/30 files checked 3% done
Checking C:\Users\singh\WorkShop_8\NumCpp\examples\ReadMe\ReadMe.cpp ...
Checking C:\Users\singh\WorkShop_8\NumCpp\examples\ReadMe\ReadMe.cpp: __cpp_structured_bindings...
5/30 files checked 5% done
Checking C:\Users\singh\WorkShop_8\NumCpp\test\cppcheck\CppCheck.cpp ...
6/30 files checked 5% done
Checking C:\Users\singh\WorkShop_8\NumCpp\test\gtest\test_BinaryLogger.cpp ...
C:\Users\singh\WorkShop_8\NumCpp\test\gtest\test_BinaryLogger.cpp:66:5: error: syntax error [syntaxError]
    TEST_F(BinaryLoggerTestSuite, TestBinaryDataLoggerConstructors)
    ^
7/30 files checked 6% done
Checking C:\Users\singh\WorkShop_8\NumCpp\test\gtest\test_Logger.cpp ...
C:\Users\singh\WorkShop_8\NumCpp\test\gtest\test_Logger.cpp:76:5: error: syntax error [syntaxError]
    TEST_F(LoggerTestSuite, TestConsoleLogger)
    ^
8/30 files checked 6% done
Checking C:\Users\singh\WorkShop_8\NumCpp\test\multiple\function.cpp ...
9/30 files checked 6% done
Checking C:\Users\singh\WorkShop_8\NumCpp\test\multiple\main.cpp ...
10/30 files checked 7% done
Checking C:\Users\singh\WorkShop_8\NumCpp\test\pytest\src\Constants.cpp ...
Checking C:\Users\singh\WorkShop_8\NumCpp\test\pytest\src\Constants.cpp: NUMCPP_NO_USE_BOOST...
Checking C:\Users\singh\WorkShop_8\NumCpp\test\pytest\src\Constants.cpp: __cpp_lib_clamp...
Checking C:\Users\singh\WorkShop_8\NumCpp\test\pytest\src\Constants.cpp: __cpp_lib_execution...
Checking C:\Users\singh\WorkShop_8\NumCpp\test\pytest\src\Constants.cpp: __cpp_lib_gcd_lcm...
Checking C:\Users\singh\WorkShop_8\NumCpp\test\pytest\src\Constants.cpp: __cpp_lib_hypot...
Checking C:\Users\singh\WorkShop_8\NumCpp\test\pytest\src\Constants.cpp: __cpp_lib_math_special_functions...
Checking C:\Users\singh\WorkShop_8\NumCpp\test\pytest\src\Constants.cpp: __cpp_lib_parallel_algorithm...
11/30 files checked 7% done
Checking C:\Users\singh\WorkShop_8\NumCpp\test\pytest\src\Coordinates.cpp ...
12/30 files checked 11% done
Checking C:\Users\singh\WorkShop_8\NumCpp\test\pytest\src\Core.cpp ...
13/30 files checked 12% done
Checking C:\Users\singh\WorkShop_8\NumCpp\test\pytest\src\DataCube.cpp ...
14/30 files checked 14% done
Checking C:\Users\singh\WorkShop_8\NumCpp\test\pytest\src\DateTime.cpp ...
15/30 files checked 14% done
```

```
15/30 files checked 14% done
Checking C:\Users\singh\WorkShop_8\NumCpp\test\pytest\src\Filter.cpp ...
16/30 files checked 15% done
Checking C:\Users\singh\WorkShop_8\NumCpp\test\pytest\src\Functions.cpp ...
Checking C:\Users\singh\WorkShop_8\NumCpp\test\pytest\src\Functions.cpp: NUMCPP_NO_USE_BOOST;__cpp_lib_gcd_lcm...
Checking C:\Users\singh\WorkShop_8\NumCpp\test\pytest\src\Functions.cpp: NUMCPP_NO_USE_BOOST;__cpp_lib_math_special_functions...
17/30 files checked 46% done
Checking C:\Users\singh\WorkShop_8\NumCpp\test\pytest\src\ImageProcessing.cpp ...
18/30 files checked 46% done
Checking C:\Users\singh\WorkShop_8\NumCpp\test\pytest\src\Integrate.cpp ...
19/30 files checked 46% done
Checking C:\Users\singh\WorkShop_8\NumCpp\test\pytest\src\Linalg.cpp ...
20/30 files checked 47% done
Checking C:\Users\singh\WorkShop_8\NumCpp\test\pytest\src\Logger.cpp ...
21/30 files checked 47% done
Checking C:\Users\singh\WorkShop_8\NumCpp\test\pytest\src\NdArray.cpp ...
22/30 files checked 80% done
Checking C:\Users\singh\WorkShop_8\NumCpp\test\pytest\src\NumCppPy.cpp ...
23/30 files checked 80% done
Checking C:\Users\singh\WorkShop_8\NumCpp\test\pytest\src\Polynomial.cpp ...
C:\Users\singh\WorkShop_8\NumCpp\test\pytest\src\Polynomial.cpp:207:42: error: syntax error [syntaxError]
        .def("__add__", &Poly1d::operator+)
                                         ^
Checking C:\Users\singh\WorkShop_8\NumCpp\test\pytest\src\Polynomial.cpp: NUMCPP_NO_USE_BOOST;__cpp_lib_math_special_functions...
24/30 files checked 82% done
Checking C:\Users\singh\WorkShop_8\NumCpp\test\pytest\src\Random.cpp ...
25/30 files checked 88% done
Checking C:\Users\singh\WorkShop_8\NumCpp\test\pytest\src\Roots.cpp ...
26/30 files checked 89% done
Checking C:\Users\singh\WorkShop_8\NumCpp\test\pytest\src\Rotations.cpp ...
C:\Users\singh\WorkShop_8\NumCpp\test\pytest\src\Rotations.cpp:204:57: error: syntax error [syntaxError]
        .def("__add__", &rotations::Quaternion::operator+)
                                                         ^
27/30 files checked 91% done
Checking C:\Users\singh\WorkShop_8\NumCpp\test\pytest\src\Special.cpp ...
Checking C:\Users\singh\WorkShop_8\NumCpp\test\pytest\src\Special.cpp: NUMCPP_NO_USE_BOOST;__cpp_lib_math_special_functions...
28/30 files checked 97% done
Checking C:\Users\singh\WorkShop_8\NumCpp\test\pytest\src\Utils.cpp ...
29/30 files checked 97% done
Checking C:\Users\singh\WorkShop_8\NumCpp\test\pytest\src\Vector.cpp ...
C:\Users\singh\WorkShop_8\NumCpp\test\pytest\src\Vector.cpp:251:43: error: syntax error [syntaxError]
        .def("__imul__", &Vec2::operator*=, pb11::return_value_policy::reference)
                                           ^
30/30 files checked 100% done
Active checkers: There was critical errors
===============================================
Finished the code scanning with static analysis tool
===============================================
test/pytest/src/Vector.cpp:2: trailing whitespace.
+// Adding a Comment (modified any .cpp file (Vector.cpp))

C:\Users\singh\WorkShop_8\NumCpp>
```

Above screenshots, we can the output for static analysis after it scans the files. In the screenshot we can also see that tool was able to analyze 30 files. Used the hook for static analyze.

Ankush Singh

**WorkShop_8**

Post Lab Experience Screenshots