HW8: Static Analysis

6.3 Writeup

Write a short summary to reflect on your analysis of the results in this step:

> This step taught me how to run cppcheck commands on various files.
> I also learned that there are different categories (or types) of problems.

Your write-up must also answer these questions:

- What are other options you explored in step 6.1? What do they do?

`--check-config`	check cppcheck configuration [1,2]
`--enable=all`	enable all checks [1,2]
`--language=c++`	modify the type of language [2]
`--quiet`	only print something when there is an error [1]
`--version`	print out version information [1,2]

Source(s): [1] https://linux.die.net/man/1/cppcheck

[2] https://www.youtube.com/watch?v=hb4OKInN-5M

- In step 6.1, what are four unique problems that you found?

```
hc219417@property-of-hanban:/mnt/c/Users/hanna/OneDrive/Desktop/CS3560/hw/hw8-static-analysis-s22-23-march-madness/source/cppcheck$ cppcheck --quiet --enable=all gui/
```

[useStlAlgorithm]

```
gui/checkthread.cpp:402:23: style: Consider using std::transform algorithm instead of a raw loop. [useStlAlgorithm]

            callstack.emplace_back(path.file.toStdString(), path.info.toStdString(), path.line, path.column);

                      ^
```

[ConfigurationNotChecked]

```
gui/helpdialog.cpp:59:0: information: Skipping configuration 'FILESDIR' since the value of 'FILESDIR' is unknown. Use -D if you want to check it. You can use -U to skip it explicitly. [ConfigurationNotChecked]

    const QString filesdir = FILESDIR;

^
```

[clarifyCondition]

```
gui/test/data/benchmark/simple.cpp:2184:38: style: Boolean result is used in bitwise operation. Clarify expression with parentheses. [clarifyCondition]

            else if (usage._modified & !usage._write)

                                     ^
```

[knownConditionTrueFalse]

```
gui/test/data/benchmark/simple.cpp:2357:17: style: Condition 'used1' is always true [knownConditionTrueFalse]

            if (used1 && used2)

                ^

gui/test/data/benchmark/simple.cpp:2354:21: note: Assignment 'used1=true', assigned value is 1

            used1 = true;

                    ^

gui/test/data/benchmark/simple.cpp:2357:17: note: Condition 'used1' is always true

            if (used1 && used2)

                ^
```

- In step 6.2, what are four unique problem *types* you found?

hc219417@property-of-hanban:/mnt/c/Users/hanna/OneDrive/Desktop/CS3560/hw/hw8-static-analysis-s22-23-march-madness/source/doxygen/src$ cppcheck --quiet --enable=all xmlgen.cpp

[noExplicitConstructor]

```
qcstring.h:120:5: style: Class 'QCString' has a constructor with 1 argument that is not
explicit. [noExplicitConstructor]

    QCString( std::string &&s) : m_rep(std::move(s)) {}

    ^
```

[missingOverride]

```
dirdef.h:116:21: style: The function 'definitionType' overrides a function in a base class but
is not marked with a 'override' specifier. [missingOverride]

    virtual DefType definitionType() const = 0;

                    ^

definition.h:99:21: note: Virtual function in base class

    virtual DefType definitionType() const = 0;

                    ^

dirdef.h:116:21: note: Function in derived class

    virtual DefType definitionType() const = 0;

                    ^
```

[passedByValue]

```
doxygen.h:61:63: performance: Function parameter 'ts' should be passed by const reference.
[passedByValue]

  LookupInfo(const Definition *d,const MemberDef *td,QCString ts,QCString rt)

                                                              ^
```

[stlFindInsert]

```
classdef.h:507:29: performance: Searching before insertion is not necessary. [stlFindInsert]

      accessors.insert(s.str());

                            ^
```

7.3 Writeup

Write a short summary to reflect on your analysis of the results in this step:

> This step showed me that there are multiple ways to generate documentation. However, I prefer using doxygen for just documentation purposes and cppcheck for static analysis.

Also, answer the following questions in your write-up:

- What shell command can generate the differences between the `doxyfile` in Cppcheck project and `doxyfile` in Doxygen project?

  ```
  doxygen -g
  doxygen Doxyfile
  ```

- Why are there `*.doc` files in `source/doxygen/doc` folder? What are the contents of the doc files?

  word processing document format (Microsoft Word)

- Is there a way to turn off the LaTeX generation for `source/doxygen/doc`? If so, how?

  Yes, under `Configuration options related to the LaTeX output` in the Doxyfile change `GENERATE_LATEX` by setting it equal to `NO` (the default value is: `YES`).

  `GENERATE_LATEX = YES` → `GENERATE_LATEX = NO`

- There is this `source/doxygen/doc/Doxyfile_chm` file, what is the purpose of this file? What is it supposed to create if you run doxygen against it?

  `chm` is the extension used by Windows help files and contains help documentation compiled and saved in a compressed HTML format.

  According to the generation rules at the bottom of the Doxyfile_chm file, running doxygen will produce an HTML_chm file (`GENERATE_LATEX = NO, HTML_OUTPUT = chm`) named `index.chm` (`CHM_FILE = index.chm`) along with three additional HTML index files: `index.hhp`, `index.hhc`, and `index.hhk` (`GENERATE_HTMLHELP = YES`), no tree-like index structure (`GENERATE_TREEVIEW = NO`), anavigation index consisting of multiple levels of tabs that are statically embedded in every HTML page as opposed to a main index with vertical navigation menus (`HTML_DYNAMIC_MENUS = NO`), the HTML help compiler (`hhc.exe`) will be located at `HTML_HELP_COMPILER` (`HHC_LOCATION = "$(HTML_HELP_COMPILER)"`), and a binary table of contents

(`BINARY_TOC = YES`). The Doxyfile will be included as part of a configuration file (`@INCLUDE = Doxyfile`).

Source: https://www.doxygen.nl/manual/config.html

- Which software has the highest documentation rate? Is there a way to measure this quantity?

  According to one online community, they recommend Cppcheck for most people, though they had pros and cons for both Cppcheck and Doxygen.

	Cppcheck	Doxygen
Pros	● fast ● updated regularly ● different output formats	● free ● cross-platform ● generates documentation from comments
Cons	● custom c++ parser ● limited ability to find bugs ● false positives ● sole focus on bugs	● i18n support is poor ● PDF output is very problematic ● no recursive inclusion

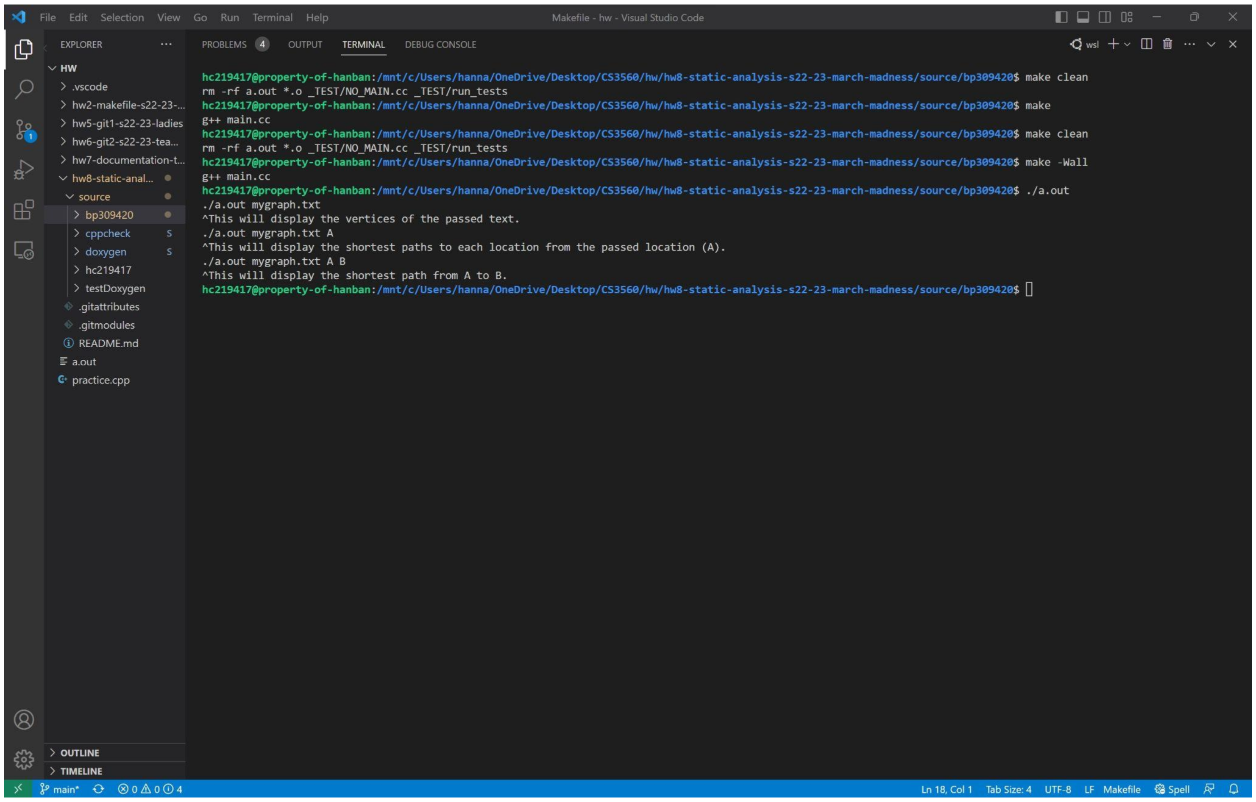
Source: https://www.slant.co/versus/839/12123/~cppcheck_vs_doxygen

8.2 Writeup

Write a short summary describing the difference between:

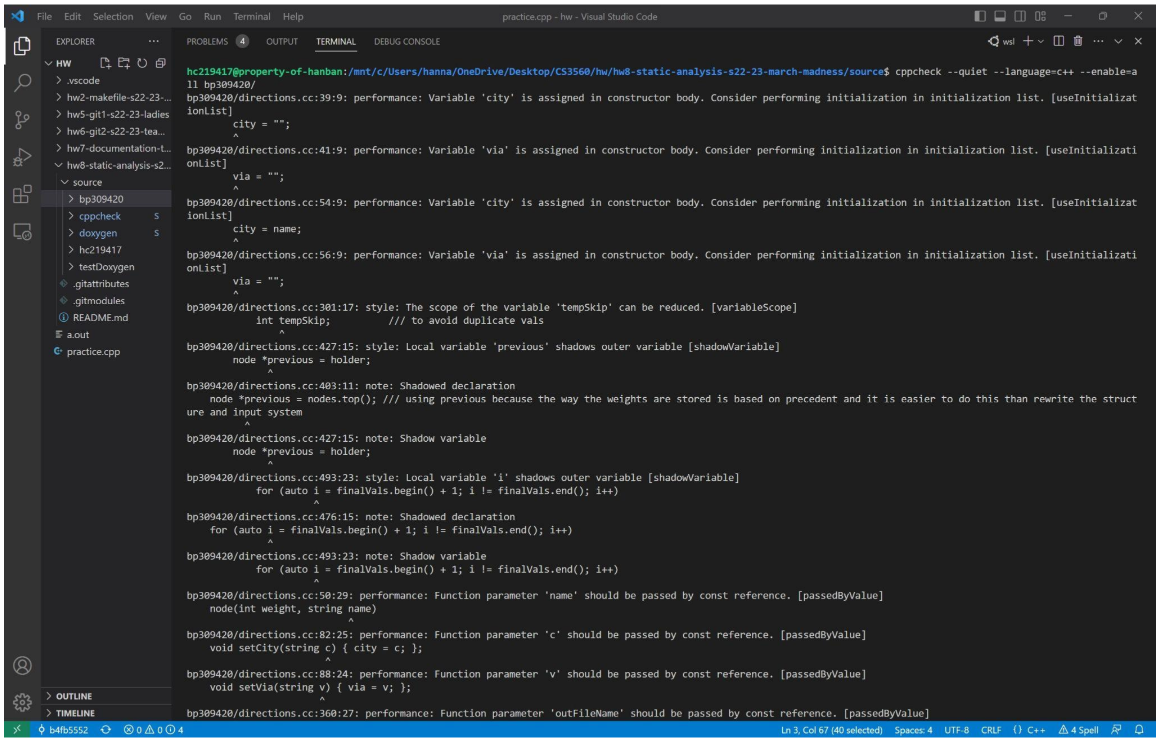
- compiler output (and which compiler/compiler options you used) from step 5
- cppcheck output (and cppcheck options) from step 8.1

  There were no error messages from the terminal in step 5 when compiling (`make`). The only output was when actually running the other student's program (`./a.out`):

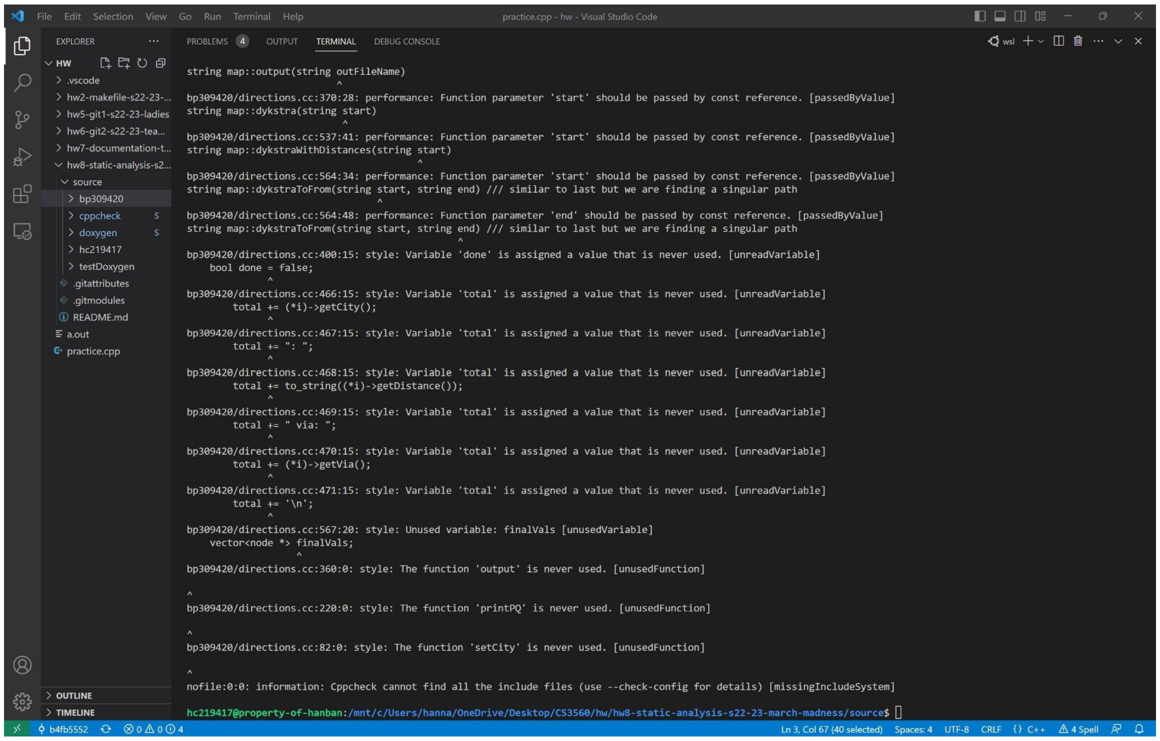


```
hc219417@property-of-hanban:/mnt/c/Users/hanna/OneDrive/Desktop/CS3560/hw/hw8-static-analysis-s22-23-march-madness/source/bp309420$ make clean
rm -rf a.out *.o _TEST/NO_MAIN.cc _TEST/run_tests
hc219417@property-of-hanban:/mnt/c/Users/hanna/OneDrive/Desktop/CS3560/hw/hw8-static-analysis-s22-23-march-madness/source/bp309420$ make
g++ main.cc
hc219417@property-of-hanban:/mnt/c/Users/hanna/OneDrive/Desktop/CS3560/hw/hw8-static-analysis-s22-23-march-madness/source/bp309420$ make clean
rm -rf a.out *.o _TEST/NO_MAIN.cc _TEST/run_tests
hc219417@property-of-hanban:/mnt/c/Users/hanna/OneDrive/Desktop/CS3560/hw/hw8-static-analysis-s22-23-march-madness/source/bp309420$ make -Wall
g++ main.cc
hc219417@property-of-hanban:/mnt/c/Users/hanna/OneDrive/Desktop/CS3560/hw/hw8-static-analysis-s22-23-march-madness/source/bp309420$ ./a.out
./a.out mygraph.txt
^This will display the vertices of the passed text.
./a.out mygraph.txt A
^This will display the shortest paths to each location from the passed location (A).
./a.out mygraph.txt A B
^This will display the shortest path from A to B.
hc219417@property-of-hanban:/mnt/c/Users/hanna/OneDrive/Desktop/CS3560/hw/hw8-static-analysis-s22-23-march-madness/source/bp309420$ 
```

Cppcheck had a *lot* more output and all sorts of problems listed out for the project itself:



```
hc219417@property-of-hanban:/mnt/c/Users/hanna/OneDrive/Desktop/CS3560/hw/hw8-static-analysis-s22-23-march-madness/source$ cppcheck --quiet --language=c++ --enable=a
ll bp309420/
bp309420/directions.cc:39:9: performance: Variable 'city' is assigned in constructor body. Consider performing initialization in initialization list. [useInitializat
ionList]
        city = "";
        ^
bp309420/directions.cc:41:9: performance: Variable 'via' is assigned in constructor body. Consider performing initialization in initialization list. [useInitializati
onList]
        via = "";
        ^
bp309420/directions.cc:54:9: performance: Variable 'city' is assigned in constructor body. Consider performing initialization in initialization list. [useInitializat
ionList]
        city = name;
        ^
bp309420/directions.cc:56:9: performance: Variable 'via' is assigned in constructor body. Consider performing initialization in initialization list. [useInitializati
onList]
        via = "";
        ^
bp309420/directions.cc:301:17: style: The scope of the variable 'tempSkip' can be reduced. [variableScope]
            int tempSkip;          /// to avoid duplicate vals
                ^
bp309420/directions.cc:427:15: style: Local variable 'previous' shadows outer variable [shadowVariable]
        node *previous = holder;
              ^
bp309420/directions.cc:403:11: note: Shadowed declaration
    node *previous = nodes.top(); /// using previous because the way the weights are stored is based on precedent and it is easier to do this than rewrite the struct
ure and input system
          ^
bp309420/directions.cc:427:15: note: Shadow variable
        node *previous = holder;
              ^
bp309420/directions.cc:493:23: style: Local variable 'i' shadows outer variable [shadowVariable]
            for (auto i = finalVals.begin() + 1; i != finalVals.end(); i++)
                      ^
bp309420/directions.cc:476:15: note: Shadowed declaration
    for (auto i = finalVals.begin() + 1; i != finalVals.end(); i++)
              ^
bp309420/directions.cc:493:23: note: Shadow variable
            for (auto i = finalVals.begin() + 1; i != finalVals.end(); i++)
                      ^
bp309420/directions.cc:50:29: performance: Function parameter 'name' should be passed by const reference. [passedByValue]
    node(int weight, string name)
                            ^
bp309420/directions.cc:82:25: performance: Function parameter 'c' should be passed by const reference. [passedByValue]
    void setCity(string c) { city = c; };
                        ^
bp309420/directions.cc:88:24: performance: Function parameter 'v' should be passed by const reference. [passedByValue]
    void setVia(string v) { via = v; };
                       ^
bp309420/directions.cc:360:27: performance: Function parameter 'outFileName' should be passed by const reference. [passedByValue]
```



```
string map::output(string outFileName)
                          ^
bp309420/directions.cc:370:28: performance: Function parameter 'start' should be passed by const reference. [passedByValue]
string map::dykstra(string start)
                           ^
bp309420/directions.cc:537:41: performance: Function parameter 'start' should be passed by const reference. [passedByValue]
string map::dykstraWithDistances(string start)
                                        ^
bp309420/directions.cc:564:34: performance: Function parameter 'start' should be passed by const reference. [passedByValue]
string map::dykstraToFrom(string start, string end) /// similar to last but we are finding a singular path
                                 ^
bp309420/directions.cc:564:48: performance: Function parameter 'end' should be passed by const reference. [passedByValue]
string map::dykstraToFrom(string start, string end) /// similar to last but we are finding a singular path
                                               ^
bp309420/directions.cc:400:15: style: Variable 'done' is assigned a value that is never used. [unreadVariable]
    bool done = false;
              ^
bp309420/directions.cc:466:15: style: Variable 'total' is assigned a value that is never used. [unreadVariable]
        total += (*i)->getCity();
              ^
bp309420/directions.cc:467:15: style: Variable 'total' is assigned a value that is never used. [unreadVariable]
        total += ": ";
              ^
bp309420/directions.cc:468:15: style: Variable 'total' is assigned a value that is never used. [unreadVariable]
        total += to_string((*i)->getDistance());
              ^
bp309420/directions.cc:469:15: style: Variable 'total' is assigned a value that is never used. [unreadVariable]
        total += " via: ";
              ^
bp309420/directions.cc:470:15: style: Variable 'total' is assigned a value that is never used. [unreadVariable]
        total += (*i)->getVia();
              ^
bp309420/directions.cc:471:15: style: Variable 'total' is assigned a value that is never used. [unreadVariable]
        total += '\n';
              ^
bp309420/directions.cc:567:20: style: Unused variable: finalVals [unusedVariable]
    vector<node *> finalVals;
                   ^
bp309420/directions.cc:360:0: style: The function 'output' is never used. [unusedFunction]

^
bp309420/directions.cc:220:0: style: The function 'printPQ' is never used. [unusedFunction]

^
bp309420/directions.cc:82:0: style: The function 'setCity' is never used. [unusedFunction]

^
nofile:0:0: information: Cppcheck cannot find all the include files (use --check-config for details) [missingIncludeSystem]

hc219417@property-of-hanban:/mnt/c/Users/hanna/OneDrive/Desktop/CS3560/hw/hw8-static-analysis-s22-23-march-madness/source$
```