# CONTRIBUTION GUIDELINES

## Before contributing

Welcome to [TheAlgorithms/C-Plus-Plus](https://github.com/TheAlgorithms/C-Plus-Plus)! Before submitting pull requests, please make sure that you have \*\*read the whole guidelines\*\*. If you have any doubts about this contribution guide, please open [an issue](https://github.com/TheAlgorithms/C-Plus-Plus/issues/new/choose) and clearly state your concerns.

## Contributing

### Contributor

We are very happy that you consider implementing algorithms and data structures for others! This repository is referred to and used by learners from around the globe. Being one of our contributors, you agree and confirm that:

- You did your own work.

- No plagiarism allowed. Any plagiarized work will not be merged.

- Your work will be distributed under [MIT License](License) once your pull request has been merged.

- You submitted work fulfils or mostly fulfils our styles and standards.

\*\*New implementation\*\* New implementation are welcome!

\*\*Improving comments\*\* and \*\*adding tests\*\* to existing algorithms are much appreciated.

\*\*Issues\*\* Please avoid opening issues asking to be "assigned” to a particular algorithm. This merely creates unnecessary noise for maintainers. Instead, please submit your implementation in a pull request and it will be evaluated by project maintainers.

### Making Changes

#### Code

- Please use the directory structure of the repository.

- File extension for code should be \*.h \*.cpp.

- Don't use \*\*bits/stdc++.h\*\* because this is quite Linux specific and slows down the compilation process.

- Avoid using \*\*struct\*\* and instead use the \*\*class\*\* keyword.

- You can suggest reasonable changes to existing algorithms.

- Strictly use snake\_case (underscore\_separated) in filenames.

- If you have added or modified code, please make sure the code compiles before submitting.

- Our automated testing runs [\_\_cpplint\_\_](https://github.com/cpplint/cpplint) on all pull requests so please be sure that your code passes before submitting.

- Please conform to [doxygen](https://www.doxygen.nl/manual/docblocks.html) standard and document the code as much as possible. This not only facilitates the readers but also generates the correct info on website.

- \*\*Be consistent in use of these guidelines.\*\*

#### Documentation

- Make sure you put useful comments in your code. Do not comment things that are obvious.

- Please avoid creating new directories if at all possible. Try to fit your work into the existing directory structure. If you want to create a new directory, then please check if a similar category has been recently suggested or created by other pull requests.

- If you have modified/added documentation, please ensure that your language is concise and contains no grammar errors.

- Do not update README.md along with other changes, first create an issue and then link to that issue in your pull request to suggest specific changes required to README.md

- The repository follows [Doxygen](https://www.doxygen.nl/manual/docblocks.html) standards and auto-generates the [repo website](https://thealgorithms.github.io/C-Plus-Plus). Please ensure the code is documented in this structure. Sample implementation is given below.

#### Test

- Make sure to add examples and test cases in your main() function.

- If you find any algorithm or document without tests, please feel free to create a pull request or issue describing suggested changes.

- Please try to add one or more `test()` functions that will invoke the algorithm implementation on random test data with expected output. Use `assert()` function to confirm that the tests will pass.

#### Typical structure of a program:

```cpp

/\*\*

\* @file

\* @brief Add one line description here

\* @details

\* This is a multi line

\* description containing links, references,

\* math equations, etc

\* @author [Name](https://github.com/handle)

\* @see related\_file.cpp, another\_file.cpp

\*/

#include

/\*\*

\* @namespace <check from other files in this repo>

\*/

namespace name {

/\*\*

\* Class documentation

\*/

class cls\_name{

private:

int var1; ///< short info of this variable

char \*msg; ///< short info

public:

// other members also documented as below

}

/\*\*

\* Function documentation

\* @tparam T this is a one-line info about T

\* @param param1 on-line info about param1

\* @param param2 on-line info about param2

\* @returns `true` if ...

\* @returns `false` if ...

\*/

template<class T>

bool func(int param1, T param2) {

// function statements here

if(/\*something bad\*/)

return false;

return true;

}

/\*\* Test function \*/

void test() {

/\* some statements \*/

assert(func(...) == ...); // this ensures that the algorithm works as expected

// can have multiple checks

}

/\*\* Main function \*/

int main(int argc, char \*argv[]) {

// code here

return 0;

}

```

#### New File Name guidelines

- Use lowercase words with ``"\_"`` as separator

- For instance

```

MyNewCppClass.CPP is incorrect

my\_new\_cpp\_class.cpp is correct format

```

- It will be used to dynamically create a directory of files and implementation.

- File name validation will run on docker to ensure the validity.

#### New Directory guidelines

- We recommend adding files to existing directories as much as possible.

- Use lowercase words with ``"\_"`` as separator ( no spaces or ```"-"``` allowed )

- For instance

```

SomeNew Fancy-Category is incorrect

some\_new\_fancy\_category is correct

```

- Filepaths will be used to dynamically create a directory of our algorithms.

- Filepath validation will run on GitHub Actions to ensure compliance.

#### Commit Guidelines

- It is recommended to keep your changes grouped logically within individual commits. Maintainers find it easier to understand changes that are logically spilt across multiple commits. Try to modify just one or two files in the same directory. Pull requests that span multiple directories are often rejected.

```

git add file\_xyz.cpp

git commit -m "your message"

```

Examples of commit messages with semantic prefixes:

```

fix: xyz algorithm bug

feat: add xyx algorithm, class xyz

test: add test for xyz algorithm

docs: add comments and explanation to xyz algorithm

```

Common prefixes:

- fix: A bug fix

- feat: A new feature

- docs: Documentation changes

- test: Correct existing tests or add new ones

### Pull Requests

- Checkout our [pull request template](https://github.com/TheAlgorithms/C-Plus-Plus/blob/master/.github/pull\_request\_template.md)

#### cpplint

To see if [\_\_cpplint\_\_](https://github.com/cpplint/cpplint) is already installed, do:

\* `cpplint --version` # currently returns "cpplint 1.4.4"

If cpplint is \_\_\_not\_\_\_ installed then do:

\* `python3 -m pip install cpplint` # If that does not work then try...

\* `py -m pip install cpplint` # If that does not work then try...

\* `pip install cpplint`

Once cpplint is installed, test your file(s) with:

\* `cpplint --filter=-legal my\_file.cpp my\_other\_file.cpp` # Fix any issues and try again.

The [\_\_clang-format\_\_](https://clang.llvm.org/docs/ClangFormat.html) tool can fix whitespace related \_cpplint\_ issues.

\* On Macs only: `brew install clang-format` # Only needs to be installed once.

\* All platforms: `clang-format -i -style="{IndentWidth: 4}" my\_file.cpp`

Most importantly,

- Happy coding!