**Personal CCD Cheat Sheet**

1. **Use meaningful variable and function names**

* Choose descriptive names that accurately convey the purpose and meaning.

1. **Keep functions and methods small and focused**

* Functions should do one thing and they should do it well.
* Functions should have descriptive names.
* Functions with clear responsibilities; More manageable.

1. **Comment where necessary**

* Always try to explain yourself in code. If it’s not possible, take your time to write a good comment.
* Don’t add obvious noise.
* Comment out code not in-use.
* Use as an explanation of intent.
* Use as a warning of consequences.

1. **Write modular and reusable code**

* Reusable code should have no side effects.
* Break down complex tasks into smaller clear, concise logic.
* Aim to keep lines of code short and readable by avoiding excessive nesting or complexity.
* Don't repeat yourself.

1. **Meaningful whitespace and formatting**

* Use consistent indentation and spacing.
* Formatting to improve readability and maintainability.
* Use whitespace to separate logical sections.

1. **Handling errors**

* Don’t mix error handling and code.
* Write the try-catch­finally statement first, it will help you structure your code
* Use Exceptions instead of returning error codes.
* Don’t return null, don’t pass null either.
* Throw exceptions with context.

1. **Test your code**

* Test-driven development (TDD) can help you write more reliable code.
* Keep your tests as clean as your production code, they should be easily readable.
* Use a coverage tool.
* Tests should be easy to run.
* One assert per test.

1. **Refactor regularly**

* Refactor the existing code before adding new functionality in a way so that the change can easily be made.
* Only refactor in small steps with working code in-between so that you can keep track.
* Remove duplication and simplify complex logic.
* Improve naming and organization as needed.

1. **Use version control effectively**

* Git to manage changes to your codebase effectively.
* Commit small changes with descriptive commit messages.
* Use branches and merge to collaborate with others.