# **97 Things Every Programmer Should Know**

## **Chapter 14: Code Reviews**

1. **Before -** I only do code reviews if my code doesn’t work.

**After -** Doing code reviews can increase the quality of code and decrease the defect rate. This is because the code will be evaluated by multiple programmers or individuals, resulting in comprehensive feedback from different perspectives.

1. **Before** - I think code review is a good practice before deploying the project or system publicly.

**After** - Many programmers tend to dislike code reviews because they have had some bad experiences. They find it time-consuming to read, check, and correct mistakes in the code.

1. **Before** - When thinking about code review, it is all about reading and checking some mistakes with the program code.

**After** - The purpose of code review is to share knowledge and establish common coding guidelines. Rather than solely focusing on finding errors, we should aim to learn and understand the code. This way, other programmers who review the code will be able to understand it more easily and make improvements as needed.

## **Chapter 15: Coding with Reason**

1. **Before -** We do coding, because it is needed in order to produce a software or web application.

**After -** Everything has a reason. This statement “Automated tools are preferable, but not always possible” is quite interesting. It is true that everyone prefers automated tools, that’s why a lot of people are invested with IoT or Internet of Things as this make their life more convenient. But is not solely possible to rely with automated tools. There are some code issues that needs to be reviewed and tested manually by human.

1. **Before** - Readability is important when coding a program.

**After** - Having a clear and well-formatted code makes it easier to under, review, and maintain. By using proper spacing, and adopting a consistent format or pattern, the code becomes more readable and maintainable. This allow programmers to easily navigate the code and quickly find the functions that needed to be reviewed and tested.

1. **Before** - When making a function, make sure it only has a single task to solve or purpose.

**After** - You should perform a specific task or single task, and should not be added by additional responsibilities or logic as it makes the code more readable and easier to maintain. By breaking down a large function into smaller, it is become easier to understand what the code is doing and to identify any potential issues.