97 Things every programmer should know

**Chapter 1 Act with Prudence**

Before-

As I was in my first year college, it's hobby of me to just let things go, like I will do this task tommorow, and same thing on the next day and so on, that every little I skip, adds up and it take more time to do it than making it before, also I don't write my task or a "To do list" because of my lazyness or procrastinate things that are not valuable, like playing, or scrolling in my facebook newsfeed, even though I know that my doing is wrong, I still do it because of me being "I can still make it, the deadline is too long, I can still make it in 3-2 days before the deadline" and when the day comes, I start cramming because of too many stack work loads that I didn't even do.

After-

Sometime later, I do realize that I can't live in always in that situation, also this chapter helps me to realize, how crucial this thing can be in my way of working, I should start to list all my priorities more, make the task as soon as possible so I can also have time to check it if it has bugs or errors, also I can ask my colleague to critized my work so I can make it better and also help me improve of skills.

**Chapter 2 Apply Functional Programming Principles**

Before-

For honesty, I don't quite understand how to write a good code or a clean code, as of for me if it's running then that is good, as long as it did what the task instruct it to do, it does not matter for me if how long the code is or short, my main focus is, if this code can run and solve the problem.

After-

Functional Programming is making code a much better, smoother, and lessen the code, also it is easier to debug or know the cause of the error, because you just reuse the other functions, as for me, I can't clearly know how to use this type of programming, but in this method, I hope to gather more knowledge so I can use it, in my future work.

**Chapter 3 Ask "What Would the User Do?" (You Are not the User)**

Before-

I tend to ask myself or put myself as a user, a not computer literate, or new to this kind technology, but in the end my speculations are wrong.

After-

Reading this chapter give me an idea how to exactly know what the user would do in program that design to solve problems, help them to make a task that takes an hour into a minutes of work..etc, watching a user do his/her initiative in using software application, is better than making myself simulate as the user.