97 things every programmer should know

Chapter 1. **Act with Prudence**

**What are the three things I learned today**

1. Before – I was lazy and I want to make things easier and faster without thinking that bugs and errors may pile up and cause damages in the long run.

After – Now I code carefully and fix bugs as soon as possible and efficient I could.

1. Before – I didn’t learn all about this kind of technical debt

After – I have learned a lot from it.

1. Before – I’ve learned all about risk mitigation and risk management only.

After – But somehow it has relationship with this.

# Chapter 2. Apply Functional Programming Principles

**What are Three Things I learned today**

1. Before – I code functionalities in local variable and I don’t use functional programming and my code goes haywire I can’t properly trace my errors.

After - After learning functional programming my code has become clean and easy to trace errors.

2. Before – I only thought of OOP as the one who can make my code clean.

After - After functional programming it has become more efficient for me to code.

3. Before – I didn’t pay much attention to my professor before when he was discussing functional programming

After - After reading this blog. I’ve become much more efficient at coding.

Chapter 3. **Apply Functional Programming Principles**

What are Three Things I learned today

1. Before – I used to think that what is right and what is wrong for me is what others also think.

After - But then I was wrong about it.

1. Before – I used to use some languages that only IT people knows.

After - And I adjusted the way I speak towards non IT People.

1. Before – I used to code that only applicable to me.

After - I value other person’s perspective. Especially the users.

**Chapter 4. Automate Your Coding Standard**

▼ What are Three Things I learned today

1. Before - I used to code that is only for my standards alone

After - Later on when coding with a team, we need to follow the standards of coding.

2. Before - I used to write variables which in my own standards.

After - I write variables prior to standards to the team.

3. Before - I like to have my own world when it comes to coding

After - It is better to code with the team.

Chapter 5. **Beauty is in Simplicity**

▼ What are Three Things I learned today

1. Before - My code is in haywire before.

After - After reading and analyzing other people's source code on the internet i learned something with great efficiency and how to make coding simple.

2. Before - My code is messy lots of misalignment.

After - I know how to align properly the codes.

3. Before - I can't properly locate errors.

After - I can properly locate errors.

Chapter 6. Before You Refactor

▼ What are Three Things I learned today

1. Before - I used to alter the code every time i see some errors

After - I use backups in my codes in order to test something which is better or properly working.

2. Before - Back then everytime I can't properly locate the errors I start from scratch again which is a bad habit.

After - I've learned that i should use backups instead and trace every errors not to waste it. Because I've coded that project for days.

3. Before - I code and code but i have so many errors.

After - Then i learned to use comments to know that this line of code has already been tested.

Chapter 7. Beware the Share

▼ What are Three Things I learned today

1. Before - Back when i was not in OJT i usually rely on source codes.

After - But after reading this chapter I've learned something that I shoudn't rely on source codes alone.

2. Before - I was lazy to study that's why i rely on source codes.

After - After reading this chapter. I've learned i should study more rather than relying on source codes.

3. Before - I thought source codes were reliable since they're made from experienced professionals.

After - I've learned that just because they're experienced professionals is you can already rely all on source codes. Always analyze the code before copying it.

Chapter 8. The boy scout rule

▼ What are Three Things I learned today

1. Before - I've never thought that boy scout rule also applies in programming but i do make some improvements in someone's code.

After - After reading the chapter, i'm relieved that i did the right thing.

2. Before - I ask my team member to correct his/her mistake before.

After - But then i did the right thing to correct it myself.

3. Before - I only care about my code.

After - Caring for the team's code is much better.