**Brendon Samson**

**Title of the Book and Chapter**

97 Things every Programmer Should Know, Chapter 1: Act with Prudence

**What are the 3 things I Learned today:**

1. **Before** – Not knowing the consequences when choosing what is quick to do than what is right.

**After** – Having known the idea of Technical Debt, if doing the quicker method is unavoidable you should make sure to change it properly on the next iteration to maintain the readability of your code and to make sure that it would not cause problems.

1. **Before** – Doing anything and/or everything just willy-nilly without minding the consequences.

**After** – Always make a plan and discuss with your team the course of your icon to avoid errors.

1. **Before** – Not keeping promises to be done to oneself, team, or clients.

**After** – It is important to keep promises, no matter to whom you have made it to, to avoid misunderstandings that could lead to more problems.

**Title of the Book and Chapter**

97 Things every Programmer Should Know, Chapter 2: Apply Functional Programming Principles

**What are the 3 things I Learned today:**

1. **Before** – Coding just for the sake of achieving the desired result not minding how it was achieved.

**After** – Implementing and practicing Functional programming is very important because it does not only make your code clean it also helps other people, or even your future self, how to understand your code. Breaking down your code into multiple understandable and readable functions will make it easier for you to debug if an error would occur.

1. **Before** – Not knowing what Referential transparency is.

**After** – Referential transparency is achieved by implementing Functional Programming which is a desirable property because it means that the function returns the same result, given the same input, whenever or wherever it is called.

1. **Before** – Reusing locally declared variables for the sake of convenience.

**After** – Knowing that mutable variables are not advised since it could cause defects on your code. Implementing Functional Programming is also the solution for this problem since you will be using smaller functions that acts based on the arguments passed into them.

**Title of the Book and Chapter**

97 Things every Programmer Should Know, Chapter 3: Ask "What Would the User Do?" (You Are not the User)

**What are the 3 things I Learned today:**

1. **Before** – As a programmer, putting oneself in the shoes of the clients/user or assuming what the clients/user wants to happen.

**After** – We cannot always assume that everyone else have the same thought process as ourselves since everybody have different experiences that would affect their train of thought.

1. **Before** – Assuming how the users act.

**After** – Be an observer. Ask simple questions from your user and listen and observe on how they do things then you can ask yourself questions about their process.

1. **Before** – Deciding which would be convenient for the users yourself.

**After** – Everyone is unique, not everyone thinks the same way you do. Always consult your users then observe their actions before deciding which course of action to take/implement.