I agree with the author on the point that time and focus contribute most when it comes to skill building, and it is never too old to learn. My three roommates at college started to play guitar at the same time. However only one of them kept playing when we graduated and he was able to be on stage with some of the guitar masters at that time. Three years of practice every day makes him one of the best guitar players in my college from zero background. Not only instrument studying, our time and effort put in any skill would always pay off.

However, as the author suggests, talent also plays an important role in skill building. First we should acknowledge that someone is born to learn some skill. It may cost a lot more time and effort for the ordinary people to catch up with those genius. Unfair it is, but it is not an excuse for us to give up. Those who wants to compete against the genius with excessive efforts may or may not beat their rivals in the end, but they would definitely become an expert.

In the field of computer science and programming, practice also makes perfect. Programming language, as the fundamental tool of computer science, is pretty easy to learn and understand, compared with those mathematic tools in other field. It makes sense to me that no one can become a good coder, being able to translate problems in real world to codes, without actually working on some big projects and putting in a huge amount of time and effort. I think for now I am trying to overcome the suck threshold.

When it comes to software design, the most important thing this article tells us is that we should make sure our program works fine for everyone without much effort. Take the operation system as an example, Window, a complex product. Most users have never read the help document, but form their common sense they would figure out how to make it work with a mouse. And if anything unexpected happens, users would just restart the machine simply and everything goes back on track. The designers should keep this in mind that when a software is designed for millions of people. If a software is designed for experts, hardly would it become a popular one.