**Chapter 1 Code Should Be Easy to Understand**

The Fundamental Theorem of Readability is the measure of time-til-understanding of code.This is what you will want to minimize.

For a code to be considered as readable other people must be able to:

* make changes to it,
* spot bugs, and
* understand how it interact with the rest of your code.

Having a readable code is important even if you’re in one-man project. Because at some time in the future, may be after 6 months you will have to review your code again and you’ll feel so unfamiliar with your code. And you never know-someone might join your project.

The less code you write to solve a problem, the better. But fewer lines isn’t always better if there will be a lack of readability like:



this code is harder to understand than this:



also a line comment can make you understand the code more quickly.



Two goals fewer lines of code and minimizing the time-till-understanding.

The efficiency of the code will not conflict on how readable it is.

Having a code that is readable requires extra work to think about whether an imaginary outsider would find your code easy to read. But as soon as you get used in this kind of writing your code you will be a better coder with fewer bugs.

**Part 1 Surface-Level Improvements**

Is about picking good names, writing good comments and formatting your code neatly.