**Chapter 13 Writing Less Code**

* Knowing when not to code is one of the most important skill a programmer can learn.
* Reusing libraries saves time and keep codebase lean and mean.
* Programmers often tend to overestimate how many features are truly essential to their project.
* Not all programs need to be fast and able to handle any input. If you follow the requirements, sometimes you will came out with a simpler problem that require less code.
* Requirements interfere with each other in subtle ways. This means that solving half the problem might only take less coding effort.
* In keeping your codebase small and lightweight you should:
  + create generic “utility” code to remove duplicated code.
  + remove unused code or unless features.
  + keep project compartmentalized into disconnected sub projects.
  + be conscious of the “weight” of codebase.
* Another way of keeping your codebase simple is to be familiar with the libraries. Libraries offers a lot of ready made code that may solve your problem, this helps in minimizing the coding effort and the time of thinking about how to implement the code.
* Use Unix Tools instead of Coding
  + There are problems that needs few lines to implement and sometimes this problem can be easily solved by using Unix Tools, so do not hesitate to use this methods.

In summary,

* Each line of code needs to be tested, documented, and maintained. The heavier it gets, the harder it is to develop in.
* Eliminating nonessential features helps in keeping code light.
* Be familiar with libraries.