

# Final Considerations



## **Donald Knuth's literate programming movement**

computer scientist,  
mathematician,  
professor emeritus  
at Stanford University  
...and a lot more

*“The most important function of computer code  
is to communicate the programmer's intent to a  
human reader.”*

OPTIMIZATION IS SACRIFICING CLARITY & MAINTAINABILITY IN THE NAME OF PERFORMANCE

*“We should forget about small efficiencies, say about 97% of the time: premature optimization is the root of all evil.”*

DO NOT SACRIFICE CODE CLARITY BY OPTIMIZING BEFORE YOU KNOW THAT YOU NEED TO

The author also continues, saying: “Yet we should not pass up our opportunities in that critical 3%.”

The message is: Avoid sacrificing code clarity for **\*\*negligible\*\*** performance improvement.

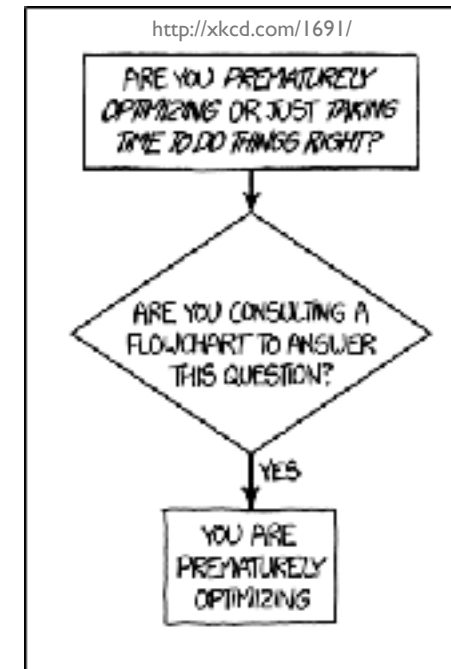


## Rob Pike's 5 Rules of Programming

Bell Labs  
Unix Team  
UTF-8  
Go Language  
...and a lot more

# 1

- You can't tell where a program is going to spend its time. Bottlenecks occur in surprising places, so don't try to second guess and put in a speed hack until you've proven that's where the bottleneck is.



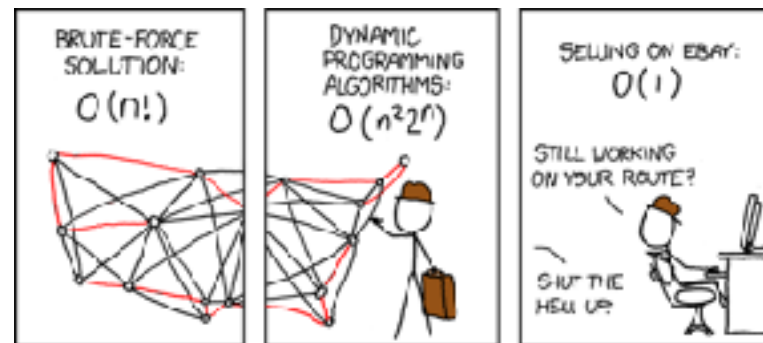
## 2



- **Measure. Don't tune for speed until you've measured, and even then don't unless one part of the code overwhelms the rest.**

### 3

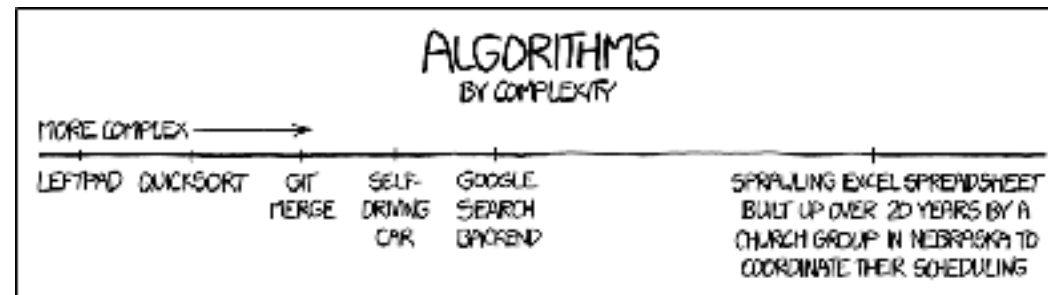
- Fancy algorithms are slow when  $n$  is small, and  $n$  is usually small. Fancy algorithms have big constants. Until you know that  $n$  is frequently going to be big, don't get fancy.





## 4

- Fancy algorithms are buggier than simple ones, and they're much harder to implement. Use simple algorithms as well as simple data structures.



# 5

- **Data dominates.** If you've chosen the right data structures and organized things well, the algorithms will almost always be self-evident. Data structures, not algorithms, are central to programming.