I am an Engineer here based in Silicon Valley. Here is what my usual day looks like.

* 6AM : Wake Up
* 6-7AM : Minor workout(20 mins) and Getting ready to work. Nothing to do with Phone yet except my morning playlist in the background.
* 7-7:30AM : Breakfast time. I spend 15mins usually for making and having.
* 7:30AM - 9AM : Commute to work, Read while in commute, Walk to work. I get off one station ahead.
* 9AM : Engineering time. I start with Mails.
* 9-12PM : Mostly with code. A good day is when there are less or no meetings. I ensure I walk 300 steps every hour.
* 12-1PM : Lunch with some work related reading stuff.
* 1-3PM : Hope for more coding(Standing desk time)
* 3-5PM : Usually spent with bug fixes and meetings I want to have. Mind isn't as fresh at this time.
* 5PM-6PM : Try to plan for next day and get off.
* Until 7:30PM : Workout + get back to home
* Until 9PM : My TV show + making dinner part.
* 9-10PM : May be work or learn something
* 10PM : No matter what, back to bed and possibly read to pass out fast. This sometime goes past 11PM.

Bottom line : Silicon Valley plays no role in a typical Engineers day. Could be same anywhere.

Most companies don’t actually want “Computer Programmers”. It doesn’t take a ton of skill or expertise to hack out some code to automate a task or provide a feature.

What companies usually actually want are “Software Engineers”. There’s a huge difference between code that works, and code that works *and is also maintainable*.

I conducted an interview yesterday, and the candidate and I did a pair programming exercise, working through a trivial problem. He solved the problem easily, and his solution was fast and simple, but I still rejected him. Why did I do that?

It was apparent that he considered the audience of his code to be the machine that executes it. While that particular reader is important, it’s also vital that a software professional realize that *people* will be reading their code, possibly for years to come. And those people will need to make changes to it.

Want details? The candidate failed to extract identical lines of logic into a method, and call the method multiple times. Yes, it was only a few lines of code, but he just copied and pasted it in the editor. In addition, some of the variable names were just single letters, not descriptive at all. And when I asked him to show me that it worked, he just hand-entered test values, instead of taking a couple of seconds to make a simple test method that can be called with different inputs.

**There you have it. Nothing else is a better indicator.**

His solution includes a hack? It does so *for good reason.*

His algorithm isn’t as fast as another one? He chose it *for good reason.*

He took longer than someone else might have taken? He did so *for good reason.*

Nothing screams “amateur” *more* than a programmer who approaches a problem blindly and implements his solution thoughtlessly and haphazardly.

A military general who leads every army on nothing other than hope and prayers, however often he is victorious, is not a general any soldier wants leading his charge.