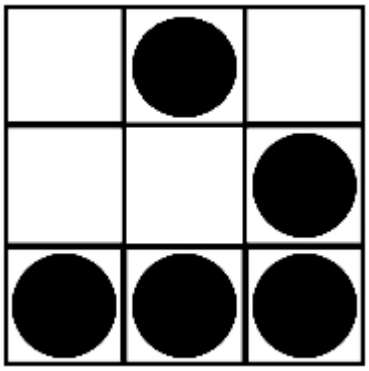
How I Learned to Program

http://rdegges.com/how-i-learned-to-program



Programming is, without a doubt, the most mentally rewarding thing I've ever done. Programming taught me that life should be fun, filled with creativity, and lived to the fullest. Programming taught me that anything is possible; I can do anything I want using only my mind.

Programming also taught me that learning is fun. It showed me that the more you know, the more power you have. Programming showed me that a life filled with learning is a life worth living. Programming revealed to me who I am inside, and has continuously helped me work towards my goals.

I feel extremely lucky to have had the means and opportunity to learn programming early in my life. While my methods are certainly not optimal for everyone, they worked well for me.

I have no regrets.

So I figured I'd share my methods with you, in hopes that a beginner will read this and get some value out of it.

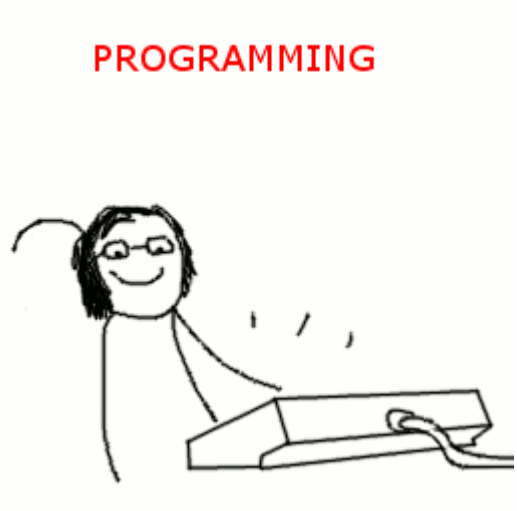
If you don't want to read all this, the important takeaway here is to, above all else, **have fun**.

Install Linux on Your Box

While in my own life, I actually learned quite a bit about computers through video games on MSDOS computers--my real learning started the first day I installed a linux operating system on my home computer.

It doesn't matter whether or not you use Windows on your laptop, or if you have a Macbook Air--if you want to learn to program well, you need to use linux. Sure, there are a ton of great programmers out there using other systems, but you cannot beat linux as a learning machine.

Despite what you may think, programmers don't just "program". Programming as you probably think of it is nothing more than input and output. You type things, and stuff happens. This is incorrect.



Programming is a way of life.

Programmers are people obsessed with knowledge. Programmers use this obsession to fuel a life of learning, discovery, and creation. That is the true definition of a programmer.

A big reason to use linux for your day to day work is that it helps you passively learn about programming as you use it. On Windows, if you want to copy a file from one box to another, you drag and drop. On linux, if you want to copy a file from one box to another, you use scp or rsync. Learning how to use the command line teaches you basic technical logic and problem solving skills.

Another important skill you passively acquire by using linux is self sufficiency. Unlike many other lines of work, programming does not require you to memorize a million things, or repeatedly do the same thing over and over again; instead, programming requires intense self motivation and determination.

Even the best programmers typically have no idea what they're doing when they start a new project. If I could summarize one thing I do more than anything else as a programmer, it would be **research**. Programmers must know how to lookup information, and how to process and use that information in a useful manner. This skill is typically acquired over long periods of time--but linux can help.

Using linux will require you to actively seek out solutions to problems. If you don't know how to setup an SSH tunnel--you **will** learn. Using linux will drive you to discover new things you never would have thought of while using Mac or Windows. As you slowly become a better and better linux user, you will coincidentally become a better and better programmer and pragmatist. You'll learn how to go about solving problems. How to hunt down errors. How to use your combined knowledge to create new things and make your life (and others' lives) easier.

Furthermore, since linux (as well as a majority of its applications) is open source, you're in a great position to learn more about programming culture. At one point or another, I can almost guarantee you will:

- Find a bug in an application you use.
- You'll search for an answer online.
- You'll find either a ticket system or a forum for the software you're using.
- You'll submit a ticket about the bug, or post on the forum stating your problem.
- Interact with other users like yourself to help resolve the issue.

While this may not sound cool now, just wait. Once you've done the above, you'll really be acquainted with the tech community. Finding problems, discussing them with others, and solving problems is what makes the technical community thrive.

If everything was perfect and there were no problems to solve in the world--life would be boring. Getting out there and fixing stuff--fighting chaos--that makes life worth living. So enjoy it!

Linux can help teach you these things, and more.

Have an Intense Desire

Why do you want to program? What is motivating you? What drives you? Unless you desperately want to learn programming, you will fail.

When I started coding, it was because I had an intense desire to hack video games. Back when I was a kid, video games were my life. I would rush home from school and spend as long as I could on the computer playing old classics. Some of my fondest memories are playing epic Starcraft matches against my brother (we'd have no rush 2 hour games where we'd max out our units and have battles that would cause our GPUs to quake in terror).

More than anything, I wanted to hack the shit out of those games. I wanted to **dominate** them. I wanted to**enslave** my computer and have it do my bidding.

While my old motivations are silly to me now, back then I felt them intensely. I'd dream about it at night, constantly think about it during the day, and obsess about it while I was on the computer in the afternoon.

When I set my mind towards learning to program, I knew I would make it happen. I knew that no matter what happened in my life, I'd either learn to program or I'd die trying. It was what I can only describe as a glorious feeling. It was similar to that feeling you have where you want something so bad, so intensely, that you feel it with every muscle in your body.

Regardless of the fact that I had absolutely no idea what I was doing--that I knew absolustely no technical people whatsoever--that I had no resources--and that I had zero guidance--I found a way. I ruthlessly read through internet tutorials on random webpages. I spend hundreds of hours scouring random forums looking for bits of information.

The most important thing, however, is that because I wanted it so bad, it felt easy. I've always been an all-or-nothing type of person, and I think that this helped me break through the initial barriers and eventually become a half decent programmer.

Build Small Command Line Programs

A lot of people now-a-days seem to be learning programming by diving head first into web development. While this may work for some people, it seems pretty damn crazy to me. Not only are web technologies complex and vast (building a modern website requires a ton of separate skills that take years to mature), but they're frustrating and discouraging for new developers.

Maybe I'm old school (I'm only 23 :x), but there is nothing more satisfying (and educational!) than writing a ton of simple, command line programs.

I can't even begin to express how useful this was in my programming education. For the longest time (after I gained a basic understanding of programming) I'd rush home from school, sit down at my computer, and spend 5 or 6 hours writing a simple command line utility. I'd write tons of things:

- A simple program that takes in filenames as input, and stores those files all in an organized directory hierarchy depending on the file type.
- An IRC bot that logs all channel activity to a text file.
- A simple program that downloads all the images on a given web page.
- A tool to convert base 10 numbers to any other base on the CLI.
- A provisioning script that installs all my OS customizations: wallpapers, themes, etc.
- A basic screenshot program that automatically uploads screenshot to an image hosting website, and copies the resulting URL into the clipboard for instant copy-paste fun.
- And a million other things.

I got so much value out of these small exercises. Each one was simple enough to be written in several hours (no more), and each one taught me stuff: either a new language, library, or strategy. There's no doubt in my mind that I gained a good portion of my programming knowledge through building these apps.

The other benefit to building these small apps is confidence. Each app I built was a huge personal accomplishment. I felt proud of each one. I'd maintain all of them, and periodically rewrite the code using all the new strategies I'd learn. This taught me basic iterative programming (making improvements over time), and how to really contribute to the open source world.

If you're a new programmer, I doubt there's anything better (or more fun) than writing a ton of small command line utilities. Don't believe me? Try it, and tell me you aren't addicted after the first one!

Write, Write, Write

Writing is a bit controversial. When I started programming, nerds had a reputation for being bad at everything except computers. For a while, I assumed that since I was good at computers, I was naturally worse at everything else: writing included.

That's bullshit.

I've come to realize over the years that programmers are, in particular, excellent writers. The ability to think logically and solve problems is a great writing asset. It's particularly hard to explain your thoughts in writing. In my opinion, having strong programming skills makes this much easier, since as a programmer, you're used to arguing with logic and use it everyday in your work.

Through the process of writing a lot, you'll greatly improve your reasoning ability, and consequently, become a better programmer.

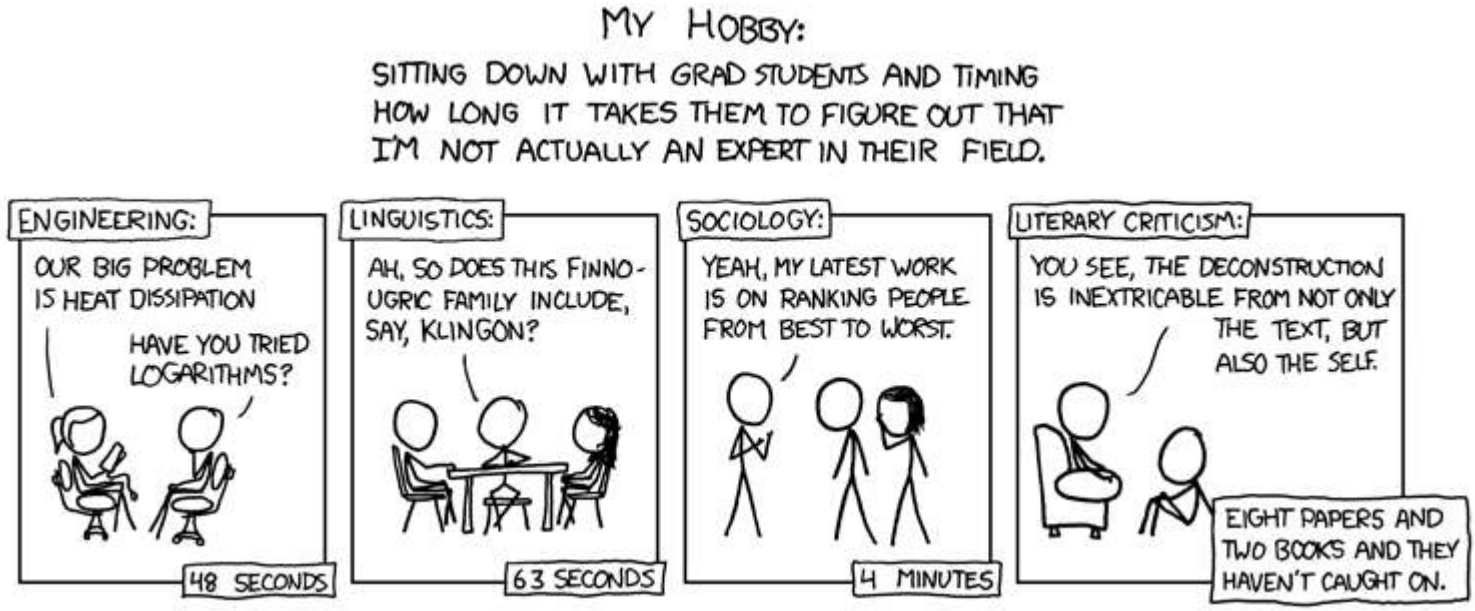
As a programmer, having a blog is a great way to practice writing. It's a great way to keep track of the things you learn, and help ensure you are always making progress. Through the process of writing about things, techincal things in particular, you'll greatly increase your knowledge on the subject.

For instance, if you're writing a CLI app that orders pizza through Dominos, it would be hard to write about that without going into detail describing the technology you're using, how the Domino's API works, etc. By taking the time to write about your experiences working with their API, you'll consequently learn a lot more than you would if you didn't.

Writing can be amazingly helpful when used to describe technical stuff, as it really simplifies and clarifies the root of the problem--forcing you to think of the problem in the simplest way possible. I can't tell you how many times I've worked on a really tough problem, then taken the time to write about it and realized that I vastly overcomplicated the issue.

One of my biggest regrets is that over the years I threw away a vast majority of my articles. Over time I rewrote my website frequently, mismanaged servers, and slowly lost most of my writing. This blog you're reading now exists primarily as the result of my decision to save all my future writings and provide a home for them that I won't mistakely lose. Don't make the same mistake I did!

Join an Online Community



The internet is a big place. Programming is a big field. While it is certainly possible to become an excellent programmer by yourself, completely isolated--it is much easier to do it with the help of friends.

When I started programming I was lucky enough to meet some amazing like-minded programmers online using IRC. The people I met were some of the smartest, passionate, and most motivated individuals I've ever met in my life. We're still friends today!

Having other insanely passionate and driven friends kept me motivated, and helped push me to be the best I could. We wrote articles for one another to share things we learned--we critiqued each other's code. We talked about projects we were working on, and what the best way to implement them was.

Having a group of people with the same passion and drive as yourself cannot be understated.

Finding a group like this, on the otherhand, is extremely difficult. I highly recommend using IRC (as a lot of bright people seem to use it), and gradually joining new channels and chatting with people who share similar interests. If, in the offchance you're like me, and want to hang out in the same circle, you're invited to check out #heapify.

Have Fun

Programming is fun. Programming is really, insanely fun. Just writing about it makes me feel happy inside. It's hard to contain my excitement.

The most important part of learning to program is to always **HAVE FUN**! Regardless of whether you're just getting into programming, or whether you've been a programmer for a long time: having fun is the most important thing you can do.
Let's say you're just starting to learn python (Dive Into Python is still one of the best python books ever written, by the way)--don't start by writing some boring project. Write something new! Something that you will find useful. Have fun with it, and challenge yourself.
If your sole motivation for working on a project is to get it done, you're cheating yourself. Part of being a good programmer is building stuff that **YOU** find cool. There is plenty of dreary software in the world, what the world needs is more **AWESOME** software. And the only way to make awesome software is to have fun building it!
I could literally go on ranting about how much fun programming is indefinitely. But instead, I want to challenge **YOU** (ya, you!). Think of something you'd really love to build: maybe it's a file sharing site, maybe a video editor--whatever excites you and gives you that warm fuzzy feeling inside. Got it?
OK, now go build it!

Regardless of where you are in your programming career: always have fun, and keep pushing yourself!