Q. How should you start learning programming?

This question has been answered literally 10s of thousands of times on Quora, but I'll give my opinion since this was an A2A.

No matter the route you take, it doesn't really matter as long as you practice a ton. Look through the popular languages and decide which one you want to start with. I'd go with one of the following: Java, Swift, Python, C, C++, smalltalk, PHP. It doesn't matter that much, and before you get a job doing it you'll probably want to have bounced around a little bit.

Instead of giving step by step instructions, I'm going to give you a few pointers. Please pay attention to these, don't just be like "yeah whatever." Following these tips will make you grow way faster, and if you actually follow these tips completely, you'll probably be the fastest learning programmer in history. **Nobody follows these rules until way later than they should have, and everyone regrets it a bit.** It's not a huge deal because everyone does it, but you can seriously make a huge difference. You could be job-ready in 3 months easy if you practice every day and do this stuff. If you don't, I'd say you're probably looking at about 1–3 years of practice at least before anyone hires you.

So without further ado, and in somewhat of a loose order of most to least important, my guide to learning how to program:

- **Practice way more than you study/read.** Don't just read how to do things, try them. You'll find that most of the time, there's complexities that aren't mentioned in the text, and you'll learn way more. Just because you think you know how to make a class in C++ doesn't mean you even know how to get the code to compile. **This is by far the most important point.**
- **Don't learn a programming language, learn to program.** Sure, you'll use a programming language and get proficient with it, but it's just a tool. If you pick up a book from start to finish on a language, you'll have learned so much stuff you are never going to use. If you insist on using a textbook, use one that emphasizes examples, or teaches you a programming practice like web development or machine learning. Most programming language focused textbooks aren't good as anything but a reference. Furthermore, it's really important not to be tied to any one language, especially as a beginner. There's a ton of great material out there that is taught in a certain language, but is universally important. You should be able to learn from examples that are written in a language you've never used.
- **Try random stuff.** Try new frameworks, packages, languages, ideas, etc. just to see how they work. Never made an app? Try it. Never used a front end JS framework? Try it. Buy an arduino. Get a book on object oriented design. Try using a NoSQL database for your next project, or try using a hosted database on AWS.

- **Don’t take days off.** Taking time off takes you ‘out of the zone.’ You want these problems in the back of your mind at all times. Taking days off forces you to get back in the zone, and remember things. I come up with solutions to problems or cool ideas all the time, whether I’m driving home from work or at the gym.
- **Make programming a hobby.** Learn to enjoy it. This comes with straying away from reading textbooks, and practicing more. Make cool shit. Get experience. Don’t say “I heard that’s a bad idea.” Say “that’s a bad idea, this is what happened to me when I tried that.” Nobody gives a shit about what you read in a textbook, I promise. Well at least not nearly as much as they care about what you experienced firsthand.
- **Finish projects 100%.** Don’t say ‘eh it’s pretty much done.’ and move on. Finish it. Publish it. Tell your parents and friends to use it. You’ll never know how much work was actually left. The very end of a project can introduce massive holes in your code. Even moreso, once you’ve launched it, your users will find plenty more. Get it published and get people using it.
- **Network.** Quora has introduced me to a lot of cool programmers that have tons of knowledge to give me. I’ve also worked with some kickass engineers/developers that gave me a ton of priceless knowledge. Work in a team as much as possible, it will teach you how companies do things at scale. Version control, task management, code documentation, code readability, loose coupling of objects, etc. are all way more important when working on projects with others.
- **Read tech blogs and info sites.** Quora is a great one. Techcrunch, Techmeme, Medium, Stackoverflow, Linkedin, and facebook are the ones I use the most. My newsfeeds on all of those are filled with the newest info on the tech industry.