## You should learn Python!

It will make your life better!

Having a basic knowledge of a programming language is really helpful. In addition to research projects, I've written programs to troubleshoot wifi, help with taxes, figure out the best time to hold review sessions, compare auto loans and other financial products, and tons of other stuff that would've been impossible or a gigantic pain otherwise.

In one meeting, we wanted some data about CSUN students. The person from IT told us it would take weeks to compile all of it. It's not that she was wrong: her staff would do a proper project. But we just needed a rough estimate. So, in a few minutes I hacked together something to download information from the relevant websites and assemble what we needed.

Why Python

I recommend Python because it's easy to get started using and extremely versatile. Even better, for virtually everything you want to do, someone who knows what they’re doing has already written a library to help you do it.

For example, if I want to analyze data, I use a library called pandas ( <https://pandas.pydata.org/pandas-docs/stable/index.html> ). To scrape content from websites, I use BeautifulSoup (<https://www.crummy.com/software/BeautifulSoup/bs4/doc/>). Networkx ( <https://networkx.github.io/> ) is fantastic for analyzing social networks,.

Alternatively, if you do not like things to be easy and enjoy tearing out your hair over problems which you don’t understand, well, JavaScript is the programming language you’re looking for....

## How to learn

You could take a class. Don’t do that. There are tons of free tutorials out there. For example: <https://docs.python-guide.org/intro/learning/> The CSUN library also has useful online materials. I'd recommend looking for materials which teach you through Jupyter notebooks (an interactive programming tool). This tutorial looks promising: <https://www.dataquest.io/blog/jupyter-notebook-tutorial/>

However, my main advice to you is that once you've got the barest of basics, think of something you want to do and teach yourself through solving a problem. Programming is a practical skill. It can sound really hard when someone describes it, but it is usually easy once you actually try it out. (Think of how hard dribbling a basketball would sound if someone described all the steps). Everything will be much easier to understand that way. I didn't really understand how to use decorator functions, until I needed a way for a bunch of different functions which use timestamps to modify the timezones.

Getting started

If you don't want to install things on your computer, there are a number of free services which will create a mini server running python for you to play with. I use Binder for projects I want to share: [https://mybinder.readthedocs.io/en/latest/ (Links to an external site.)Links to an external site.](https://mybinder.readthedocs.io/en/latest/) It's actually easier to use than the documentation suggests. Once it starts up, you'll have a Jupyter notebook to play with or try out some of the tutorials you found elsewhere.

If you want to use it on your computer, there are two things you should do right off the bat.

(1) Install the python interpreter on your computer. This just lets you run programs written in python.

(2) Install Jupyter notebooks. This gives you an interactive way of developing stuff. The notebook runs in your webbrowser and is easy to use. Virtually everything I do in python starts in a notebook and gradually migrates into a more formal program structure as needed.

(3) However, my main advice to you is that once you've got the barest of basics, think of something you want to do and teach yourself as you solve a problem. Everything will be much easier to understand that way.

## Other languages

Convinced to learn a bit of programming, but not buying my Python-sales-pitch? Ok. Fine. Here’re some thoughts on other languages which I use:

PHP: My first love. I still use it for the server-side of web-apps. It is not useful for data-analysis or small on-the-fly projects. If you are creating websites, you need to use a framework. I like Laravel (https://laravel.com/)

JavaScript: JavaScript is evil and hateful. Get ready to have fun with all sorts of unintuitive problems created by ‘features’ of this language (e.g., variable hoisting, closures, and functions nested inside functions held inside variables nested in other functions). That said, if you're going to do anything on the web, you need to learn JavaScript. Fortunately, there are some decent frameworks which make life a lot easier. Everyone (myself included) seems to start off with jQuery; it does make things much easier. Nowadays, I like Vue for building reactive webpages.

Java: If you're writing real software, you should probably learn Java. If I remember correctly, it is the language all of our computer science department's graduates know.

C: This is hardcore stuff. Where interpreted languages like Python hide what the computer is actually doing, when you write in C, you’re in charge of everything. That includes keeping your computer from turning into goo. If you want to make cool gadgets or robots, get yourself an Arduino and learn a little C.

C++: Just like it says in the name, it’s the C programming language, plus (plus) things like abstraction to make life easier.

Note that you can actually use Java, C, and C++ inside of Python when you need them. That’s pretty much the only reason I use them ⎯if I need something to run super fast or otherwise need a very high level of control over a process.