Life After Code in Place

Congratulations on completing Code in Place! It has been an honor and a pleasure to have you as part of our learning community this year. This may be the end of the course, but it's also just the beginning of your programming journey. There are so many areas available for you to explore now and so many problems worth solving, and it's now just a matter of you deciding what you want to do next. We know that's a daunting question, so we've put together this document of some free and accessible resources that we think are particularly great. Of course, this isn't an exhaustive list, and there are plenty of ways to deeply understand something that we aren't listing here.

Completing CS 106A

Code in Place is loosely based on Stanford's introductory programming course, CS 106A. Here's the course webpage from the last time Chris and Mehran taught the class. The syllabus from that class differed slightly from what we learned in Code in Place, so if you want to follow through with the rest of CS 106A, you'll need to do the following:

- Install Pycharm. CS 106A has students program on their own computers, rather than on Ed, and some of the libraries that the class uses aren't fully supported by Ed.
- Review lectures 12 and 13, which cover building graphical interfaces and animations in Python
- Follow on from lecture 19 onwards.
- Additionally, you're welcome to do the assignments and section problems to practice the material.

CS 106B

The sequel class to CS 106A at Stanford is CS 106B. In this class, you further develop your toolkit by deepening your understanding of how a computer represents data and learning about additional problem-solving techniques and structures. The class is taught in C++, one of the most popular programming languages in history.

Course website: http://web.stanford.edu/class/cs106b/

Old recorded lectures: https://see.stanford.edu/Course

Continuing with Python

After CS 106A, you might wish to continue your journey of learning Python. There are countless resources online for learning Python, but three we'd recommend in particular are:

- Real Python has a wide variety of articles at different levels of expertise about Python, as well a full series of introductory articles.
- Automate the Boring Stuff With Python covers a wide variety of interesting applications of Python, such as scraping the web, working with PDFs, and updating spreadsheets.
- Python Tutor allows you to write programs and visualize your computer's memory as it executes.

Area Specific Paths

Some areas have a list of **core topics** you need to understand to engage with meaningfully with thohse disciplines. After you develop a solid understanding of these core topics, you can pick and choose whatever subareas you find interesting and focus there.

Data Science, Artificial Intelligence and Machine Learning

Core

- Probability and Statistics (CS 109)
- Data Science with NumPy and Matplotlib (CS 102 and Harvard Data Science)
- Linear Algebra and Multivariable Calculus (Mathematics for Machine Learning, 3Blue1Brown's Youtube Series on Linear Algebra, 3Blue1Brown's Youtube Series on Multivariable Calculus)

AI/ML

- Machine Learning (CS 231N, CS 229 on Coursera)
- Natural Language Processing (CS 124, CS 224N)
- Computer Vision (CS 231N)
- Reinforcement Learning (CS 234)

General Resources

- Kaggle has a wide catalogue of interesting datasets, as well as a pre-setup programming environment with data science and processing libraries, which is really helpful when you're looking for an interesting project to work on.
- deeplearning.ai has a full fledged AI curriculum, designed by leaders of AI in education and industry.

Computer Systems

Core

- Computer Architecture and Systems (CS 107 website and videos)
- Principles of Systems (CS 110)

General Resources

- Miguel Grinberg has a series of blog postson how to use Python to write devices that can connect to the internet and physical world.

Graphics, Game Design & Virtual Reality

Core

- Linear Algebra and Multivariable Calculus (Mathematics for Machine Learning, 3Blue1Brown's Youtube Series on Linear Algebra, 3Blue1Brown's Youtube Series on Multivariable Calculus)
- Core Physics, like Mechanics and Differential Equations

Areas

- 2D/3D Graphics, rendering, animation and geometry (CS 248)
- Virtual Reality (EE 267)
- Animation and Simulation (CS 348C)
- Introduction to Game Design and Development (CS 146)

General Resources

- The Pygame library allows you to start building graphical games very quickly, using the tools that you learned about in Code in Place.

Web Development

Web development is how we make websites and online applications to do useful things. It consists of two major areas.

Frontend programming deals with everything related to what a website user can see and interact with such as the design, style, menus, text, images, etc.

Backend programming deals with everything that has to do with all the logic and internal working of a website that is not typically visible to a user. This is stuff like storing data in databases, making a server, authentication, creating users, generating dynamic pages, etc.

These resources will generally cover both frontend and backend programming:

- CS 193X
- Mozilla: Learn Web Development
- The Flask Mega Tutorial
- CS 50 Web by Harvard University

Django and Flask are two Python libraries you can use to develop web applications.

Mobile App Development

iPhone App Development

- Apple's Swift Resources

- CS 193P

Android App Development

- Google's Kotlin Resources

Other introductory syllabi

- CS 50x
- Open Source Society University

Other Programming Languages

Python is a wonderful language, but there are many other wonderful languages you might be interested in learning.

- HTML, CSS and Javascript are the best tools for developing internet-based application. Mozilla’s resources are a fantastic introduction.
- C and C++ are two of the most commonly used programming languages, and are great for programmers who want more direct control over what their computers are doing. You can learn C++ from a combination of CS 106B and CS 106L, and you could learn C here.
- Rust is a more recent language that also affords programmers very low-level control of their computers.
- Java and Go are great choices to build systems that must handle large amounts of data.
- Haskell, Scala, and OCaml are programming languages that promote a style of programming known as functional programming, which often is enormously helpful in processing data quickly.

Many programming languages rely on very similar constructs to Python: they also have variables, loops and functions, and often your task when you learn a new programming language is just to understand the syntax of how those common tools are expressed. Learn X in Y minutes is a great tool for a quick review of differences in syntax between different languages.

Tech Interviews

Interviews for tech internships and jobs are kind of their own skill that really gets better with practice. These are some good resources to practice for tech job interviews. Remember, you don't want to be memorising these answers. Instead you want to develop your computational thinking so that you can figure out these answers on the spot!

- Cracking the Coding Interview
- Hackerrank

Suggestions from your Section Leaders

These are a collection of various resources contributed by your wonderful Section Leaders. They aren’t structured in any particular way but they might serve as a useful reference for you!

- Marisa H has a podcast, Blossoming Technologist, that is a good resource for anyone interested in getting into the tech industry. It contains interviews with technologists to discuss various tech careers and skills. Season 2 is coming out soon and will dive into some specific programming languages too! The podcast can be found on anchor.fm, and on all podcasting platforms.
- Zheng Y has an educational CS YouTube channel, which you should check out if you're curious how apps like Google Maps work behind the scenes to calculate best paths!
- Two Minute Papers summarizes in fun and digestible ways all the cool stuff that CS researchers have been recently working on: from new fluid simulators to new AI that destroys people in games somehow even more than the last one. *Thanks to Zheng Y for the recommendation!*
- PyCon is one of the largest Python conferences in the world, and tickets are relatively affordable. *Thanks to Nayeon S for the recommendation!*
- Cory Shaefer has a YouTube channel which is a fantastic resource for people both struggling with the programming basics and learning new skills. He mainly uses Python. He has short videos on very specific concepts, like loops. *Thanks to Gretel U for the recommendation!*
- Microsoft has an Introduction to Python series which is broken down into small lessons on specific topics, so you can look up specifics that you are struggling with or just browse around to find new things to learn. *Thanks to Gretel U for the recommendation!*

- Statquest by Josh Starmer breaks down statistics and machine learning topics in brief videos and makes them accessible. *Thanks to Gretel U for the recommendation!*
- The O'Reilly book Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems uses really robust python libraries to teach machine learning and comes with a GitHub repository. *Thanks to Gretel U for the recommendation!*
- MIT's Missing Semester develops your familiarity with common programming tools. *Thanks to Mark F for the recommendation!*
- Raymond Hettinger has some fantastic talks on the design and philosophy of Python programs. We'd recommend The Mental Game of Python , Transforming Code into Beautiful, Idiomatic Python and Best practices for beautiful intelligible code. *Thanks to Mark F for the recommendation!*
- Pycoders has a weekly newsletter outlining interesting topics related to Python. *Thanks to Trunojoyo (Atun) A for the recommendation!*
- Open Source Society University: Bioinformatics is an open-source curriculum on bioinformatics composed of online courses. It lists many available courses on computational biology. *Thanks to Arseniy S for the recommendation!*
- Processing is an open-source MIT-developed environment for creative computation, focused mainly on visual art, but also deals with sound. *Thanks to Michael M for the recommendation!*
- No Starch Press is selling a bundle of fantastic Python booksfor an incredibly discounted price. *Thanks to Geraldine B for the recommendation!*
- Think Python is a fantastic, free, reference textbook for the Python programmer, new or old. It includes explanations, examples, problems and, via github, solutions. *Thanks to Steve G for the recommendation!*