LEARN TO CODE ALGORITHM: THE EASIEST/INCLUSIVE RECIPE FOR NEWBIES

Kgotso Koete, January 2017

No more lies...

This learning to code trend conceals a lot of nuances. There is hard and easy content, and there is the hard or the easy way to learn. When the dominant narrative is that computer programming is easy, people are often talking about the easy way to learn the easy part. The opposite can be said for those who say it's hard or that it must come naturally.

The 3 biggest issues with programming advice that people give is:

1) Advice is not empathetic and forgets that programmers were once human too.
2) The advice often lists too many resources without giving the newcomer a method to prioritize - what's the point of overwhelming people.
3) Advice often does not consider that people have different cognitive strengths and weaknesses and life circumstances. This requires different kinds of resources that accommodate different situations. Programmers have the curse of knowledge, so they give advice having forgotten how they learned to code in the first place, and sometimes forgetting about the privileges that they enjoy which other may not have access to.

We don't need another list of '30 resources to help you learn to programme in language x'. While those are helpful, what we need a simple recipe that newbies can use and adjust. This is a simple 3 step *learn to code recipe:* 1 know your cognitive preferences, 2 try and switch various resources till you find a good fit, 3 specialize later, and adjust the recipe.

***

HERE IS THE 3 STEP RECIPE

These steps may seem like a lot, but you can through all of this in a matter of months because you will be switching between resources anyway. Don't worry about the # and the {}, this is a primer on how you will be implementing your algorithms in pseudocode (more on this later). Besides, I just could not help but write fake code :).

[1] Figure out your inclinations and learning **preferences. //** Do you prefer videos vs text, theory vs practical, detail vs visual?

[2] Learn to code the easier way by solving for a platform that will **minimize learning friction,** don't solve for content/language yet. Your starting language does not matter in the medium/long-term, you will learn to switch languages once you understand language structure/feature anyway.

# **TRY - SWITCH - ADJUST:** While you are still learning newbie fundamentals, do the following:
{
    # **[A] PRACTICE:** While you are not stuck on basic syntax issues, practice by using your preferred learning method:
    {
        Try a:
        A theoretical varsity course: A University Course like Harvard's CS50 on EdX (I WOULD START WITH CS50) or MIT 600.1x (EdX).
        Or switch to a:
        A practical online course: If you don't like theory, then do Free Code Camp, graduate and work on real projects for non-profits (I am yet to try this, I will only start in Feb 2017).
        Or switch to a:

A book: If you really prefer text instead of videos, then read a book like *Learning Python The Hard Way (for example)*. You can crowdsource advice on step [A], by following step [B] below.

}

[B] ASK: When you get stuck because of syntax issues, then:

{

Try:

Crowd source answers: Google or ask a questions in the relevant community channel/page on Stack Overflow (your new bestie), Reddit, Quora, YouTube: In that order.

Or switch to a:

Cheat Sheet: You can look for other people's code for the same problem on GitHub or Geek for Geeks.

Or switch to a:

Quick and dirty course: If you feel you really need to brush up, then do a Code Academy or Udacity course in the language giving you problems.

Or switch to a:

A visual programming course: If it is really bad and computer code is still hieroglyphics for you, then do a visual programming course from UC Berkeley in a fun language called BYOB. I would not spend too much time on this.

}

[C] CONNECT: While you learn, it's good to accumulate more knowledge outside of your computer and meet people (outside of the internet):

{

Try:

Meetups: Meeting other developers through events from Meetup.com for networks such as Code & Coffee or user groups (Linux User Group or Python User Group).

Try:

Podcasts: Code Newbie by Saron Yitbarek is cool and newbie friendly show you can listen to often.

Try:

Blogs: Medium, classic blogs such as Norvig's, and Hackerdom if you want a more pungent geeky flavor. If the cultural paradigms on these blogs make you feel excluded, revert to the step right above this one.

}

}

[3] When you are ready to start solving for deeper/specialized content (web or Machine learning etc.) + related language (JavaScript or Python respectively), then:

{

Rinse, wash, and repeat loop **[1]** and then **[2]** above for next phase of deeper/specialized learning.

Tweak this algorithm for new needs/circumstances/skills.

// Remember, learning a new language is very manageable when you have the basics. Programming languages may have different capabilities, but they all share similar structures/ features.

}

If this recipe looks like English to you even with the indentations, the # and {}, then it will be easy to talk to machines. Or else, just keep trying different combinations and switching resources from the options above and I promise it will make sense soon.

The key to this recipe is switching between [A], [B], and [C] whenever you are stuck. I did the Harvard CS50 course and I thought I would never be able to programme. I then switched to Code Academy to learn the basics. I then did the MIT equivalent of CS50 and did just fine. Don't let cognitive friction make you feel stupid.

Feel free to suggest how to improve this algorithm so that newbies are never lied to again. Fork and share as much as you like.

Regards
Kgotso Koete