LEARNING PROGRAMMING IS A LONG RUN NOT A SPRINT

I've decided to share my experience as a programmer with the fresh students here. I'm in programming for more than one year now. There are many things that I was using in the past for learning enhancement, and most of them I use today. Each of the students that take the Dataquest course have their own story, they all have a unique approach. Everyone is different and live in different circumstances, and for those reasons it's possible that this essay won't provide good directions for everybody but there is a chance that it will help you.

The most important thing that is mentioned here, is the sentence that is placed as a headline. IT IS very very VERY important! Please read until the end of this paragraph and then take a 10-minute break. Just to think about it. If you are a beginner, or maybe you studied programming for some time but feel burned out, this truth is crucial for changes. It's impossible to learn programming in one week or one month. It's just like running: if you know that you have to run 1000 km, you don't sprint. Never! To succeed in the long run, you have to be smart about it. You can't do anything that is overkill. You shouldn't even think: how long does it take? You have to find a way to see the whole programming mission as something uplifting, interesting and exciting. Without that, you will fail.

I decided to share my experience in a few short stories and examples because I believe that it's easier to understand a process (in this case my learning process) as a whole if it's divided into smaller sections.

Before I start, I would like to share a little bit of my background – it will make it easier to understand some of the examples given later. I finished my regular education many years ago. I work at a bank and, at the same time, in wholesale's gaming industry. The reason I started learning programming is very simple: I see it as a great opportunity for doing creative stuff and earning good money while doing it. The goal wasn't easy because working and studying at the same time can be overwhelming. But it doesn't have to be.

How to study when there is little time to do it?

It's a tricky question but it must be answered. If you want to learn anything, you need these few crucial things:
- Time
- Energy
- Motivation.

It would be good to learn at least 5-6 days per week. You probably think that it's impossible for you. In my opinion, in many cases it is more than possible. The issue lies elsewhere: people are not aware how much time they waste daily. To see that for yourself, open the settings on your smartphone and navigate to battery settings. On most smartphones, you can find some sort of chart or data set that shows how long you're using each app daily. In my case, I was playing games and reading funny stuff online for 3-4 h per day, almost every day! I realized that if I cut this time in half, I will have at least 2h for learning every day. That's a lot. That's 40h per month! 480 h per year.

We've established that there's definitely time available somewhere in your life. But how to actually find it?

In my case, the idea was: set up both jobs in such a way that would allow me to spend as little time as possible at work. Also, set all things in such a way as to spend the smallest amount of energy while working.

I have two priorities:
1. Get to know Python programming: this year is dedicated to this mission.
2. Earn enough at my current jobs to survive.

How to do that without being fired? I followed an ancient Japanese idea: when you work 20% of the time, you are doing things in the most effective and profitable way for the company. The rest of the time is mostly ineffective. Try to focus on the things you make during this 20% and make them as best as you can. Also, look for something more, something new, a challenge, or an opportunity, that can be added to this 20%, and, as a result, you will have 30% of epic work with legendary outcomes. I'm aware that this might not be possible at every job. If that's the case for you, maybe the first step is not to start programming but to get a job that will better correspond with your learning desires?

The second thing I did was to reorganize my work-time: I realized that I can do my 30% of epic work within a shorter span of time. Because most of my earnings depend on the number of sales deals I make, I realized that I can do all of my work in just a few hours (instead of pretending to work for 8 hours). I also decided to work half-time. This decision came with sacrificing part of my earnings. But learning Python is a priority.

My actions created the right conditions for me to learn: I have more time now. Less money as well, sure, but until I can pay my rent and buy some food, that's fine. Why? Because I'd rather sacrifice 12-16 months to learn something that is interesting and gives me an opportunity for a better life, than earn twice as much as I earn now and keep doing something that doesn't bring me any real satisfaction.

How to find the energy to learn?

Okay, suppose that you are not fully drained out of energy because of work. Even if you don't have to work very hard at your job, the amount of energy of a human being is limited. Also, most likely, we will have to learn after work - in the evening or during the night. That can be difficult. But is it really? What is so hard about? From my experience, it's the brain fatigue. It doesn't matter that my job is easy now. Many different things have to be done every day: you have to buy groceries, take care of your family, run an errand. And just like that, the whole day goes by and the only thing you want to do when you get home is to watch a movie or read a book, and relax. How to deal with that? The answer is easy: you need to rest. In my case, I needed to change my whole daily (and nightly) schedule. I based my new schedule on an idea used by NASA astronauts. I wake up at 9 AM (eat some food and start my workday - I work remotely). After 2 PM I do the buy-groceries-drive-somewhere-run-an-errand part. I return home, eat something and go to sleep for 2h, most often between 7 PM and 9 PM. I try not to sleep longer, because if I do, I wake up feeling drowsy. Two hours is good nap time. After waking up I don't study right away. First, I dedicate 1-2h for something fun like watching a movie, or having a snack. It creates a positive flow and wakes me up. After that I can start learning, and finish as late as 3-4 AM when I go to sleep again. I have, in total, 7-8h of sleep per day: 6 hours from 3 AM to 9 AM plus the 2h evening nap which makes 8h in total.

Motivation is the third most important ingredient.

This topic is broad and I had trouble dividing it into reasonable sections. In my opinion, the most crucial parts are:

- Never learn.
- Never work.
- Take care of your mind and your brain.

Crazy, isn't it? But it will make sense, I promise.
First of all, if you have time and space for learning already, there's another thing you will need: a proper perspective. You have to truly see programming as something interesting. It may sound ridiculous because how can you see something as interesting if you don't know anything about it? Everybody's life path is different. In my case, I wanted to create a few programs, and I was willing to know more about how to do it. Also, I was wondering what are the mechanics "behind the curtain" that drive programs. That was a big part of what hooked me on Python.

Let's get back to our main paradigms:
- Never learn.
- Newer work on programming.
- Take care of your mind and emotions.

To me, learning and working sound like pressure, stress and no fun at all. I don't like those concepts. Both of them are related to effort. Both of them drain your energy and weaken your motivation.

Instead of learning, do exciting things! Just realize that when something is exciting, time stops, it doesn't exist anymore, it's not necessary - surely you had this experience! You can do exciting things without effort. And usually, you can do them longer than you do the studying, and much more effectively than you do work. If you understand the depth of this idea, then you will NEVER learn and work again. Most likely you will achieve great success at your job too! It will be achieved without achieving - that's a great paradox and a secret to living a good life.

But what to do when for some reason you don't feel excited about discovering Python? Take a break and see what's going on in your mind.
Because I don't want to make this essay very long, I will point out only a few most common problems that may be a reason for not being excited and my personal solutions to them.

Problem 1.
You may be scared of one particular programming lesson.
Solution: Don't try to force yourself to learn it! Just set it aside and let yourself be with the fear. Try to welcome it with openness and a smile. Beyond it lies a false belief that you need to succeed at all cost. Once you understand your fear, you will realize that you can handle the lesson with ease.

Problem 2.
You may stop feeling that learning programming is exciting because you started treating it like work or school.

Solution: As I explained before, that kind of mindset is wrong. But it's difficult to let go completely. It's rooted deeply into our social habits and the education system that often corrupts the idea of discovering things. Give yourself some time and contemplate the difference between working/learning and doing exciting things. Think what's more effective and which attitude will bring you joy instead of stress.

Problem 3.
You were too excited about your programming explorations and spent too much time learning without a break.
Solution: Take a break. Remember that it's not a sprint. If you are so excited that you can spend 6-8h programming, next time try to spend 1-2h less (remembering that overkill has consequences). If something is so exciting that it's impossible to let go, after finishing a project, give yourself a few days off. Never judge yourself for taking this break! You deserve it!

Problem 4.
You had a busy day at work or after work, had to do something with a great amount of effort, or an intellectual, an emotional engagement that made you feel wasted and burnt out.
Solution: forget about programming today. Remember that it's a long-distance run and you can't disregard your work, your social needs, or any other non-programming things completely. Sometimes they have to come before programming. That's alright, there is no fault in that. The only mistake you can do is to see it as such - if you blame yourself for not focusing on programming enough, that means you’re trying to learn with too much effort. And remember that learning effortlessly means learning efficiently. If there is a lot going on in your life, take it easy with programming.

Problem 5.
You don't understand one particular lesson and feel powerless.
Solution: Try to find a solution on the web and if you don't find any, ask the community on the Dataquest platform. This community is more open and friendly than others. In many cases the one on Stack Overflow - at least in my opinion.

Problem 6.
You had trouble understanding one particular programming lesson, found the solution on Stack Overflow, and you completely couldn't understand it.
Solution: This is a funny one. There are people out there that can create an inadequately complex answer for a very simple problem. It's like looking for an instruction on how to use matches and finding a 300-page book on the subject. If you encounter a long and complex answer for a specific problem that you know is a simple one - or just assume that it is - then look further and skip the

long and bizarre answers. Sometimes a longer answer is better because it provides a more professional or a faster execution of a program but quite often it actually indicates someone's lack of understanding. And don't worry, in most cases you can find at least 5 answers. Look for the most elegant and easy one.

Secret extra tips and tricks:
Problem: I need motivation, I don't feel that I enjoy exploring programming anymore.
Solution: Use everything that can help you get back in the game. What that could be exactly depends on a person. Personally, I'd recommend:

Wasting some time:
In my case, sometimes motivation comes after allowing myself to waste some time on my phone. Feeling that I have freedom of choice is important! But while wasting my time I am fully aware of what I'm doing. I'm aware that my programming mission won't develop any further until I get back to it. I'm aware that I want to change my life and that the game I'm playing on my phone isn't a solution. There is a point where I just don't want to play on my phone any longer and I get back to programming.

Doing a drink:
Yes! In my case, a glass of whiskey and a bowl full of pistachios is a good motivator. If you feel that it's helping you, use it. However, if you start to feel that it's reducing your efficiency, set the drink aside.

Doing something else:
When you study one thing for a long time, it might become boring. It's a good idea to find something else that is also interesting and important from the field of Python. After some time you can get back to the other thing.

When you've had enough of reading:
Instead of reading, write code. If you have enough of both, watch a tutorial on Youtube. There are many different ways of exploring Python. Choose whatever feels most exciting at the time.

Changing your style:
The only certain and constant thing in life is change. Every day something else may be more useful and more effective in your learning process. It's a good idea to be self-aware. Meditation can help with that. The most important thing is to stop whatever you are doing at the time, and in this passive state of simple observation, try to see what doesn't work and what needs to be changed. Understanding yourself is crucial.

Reducing the noise:
Many times I had trouble concentrating because of noise. I started using earplugs and it helped me immensely. I realized it wasn't only about reducing the noise but also about hearing my own breathing. It creates a sort of meditating rhythm which makes it easier to follow one thought, one task at a time. In many, many cases it allowed me to learn 80% more effectively!

Getting very well acquainted with the basics:
You should pay full attention to the fundamentals, e.g. loops, dictionaries, pep-8. Without those it will be harder to plan and write a good program later on. When you finish a lesson from Dataquest, read it again. Ask yourself: do you understand all that you have read and then written? If not, delete the code and write it again - do that until you feel that you can do the task with ease.

Another important thing: if you want to learn something that goes beyond a Dataquest lesson but is somehow related to it, it's a good idea to do so. You will feel more satisfied and empowered with your extra knowledge and, as a bonus, you can share your experience with the Dataquest community and help others.

Experiencing Python may be fun but it may be a nightmare too. The key to success is following the learning method I have described here. I hope that it will help you turn programming into an exciting experience instead of a stressful obligation.