**Overview**

This document will present my reflective piece of everything that happened this trimester. What I have learned (WHAT), How I used what I learned throughout this trimester (SO WHAT). How will I use what I learned moving forward (NOW WHAT)

**What I have learned (WHAT)**

I want to start off by saying I am not new to computer science. I am a software engineer, and I've been in the industry for the past 11 years working on various projects and in multiple positions. Some as a lead, others as senior developers. I have worked on multiple technologies and roles, such as a game developer, an engine developer, an app developer, and a web developer.

Going through this unit, I did not expect to learn much, but I was very wrong and pleasantly surprised by what and how much I have learned.

To start, I never really used Python in any of my projects. My python learning journey started last trimester when I had to create a web interface for a student registration application. I did that in Django. I learned a lot but not as much as I have this trimester. Although I already know a lot of OOP concepts, this trimester, I learned how Python implements them and how to use them.

For instance, I learned that Python doesn't have a virtual keyword like C++, C#, Java, and so on. So instead, you override a function like you would in any other language without the use of the virtual/override keyword.

I also learned how Python doesn't have access modifiers. Everything is public in Python. Well, not really. You can mark something private by using the double underscore at the start of the variable name like self.\_\_name. This doesn't make it private in the normal sense of private, but instead, name mangles the variable to another name. (*Name mangling in Python* 2020 geeksforgeeks)

Furthermore, I learned how abstract functions are used in Python and how unit testing is essential when creating any software. Because of this unit, I started doing unit testing in my full-time job. While it is fun and useful, it can be time-consuming, and you get the sense that no tasks were accomplished that day which is not true because unit testing can help find bugs in software much easier.

The other thing that I learned this trimester is UML diagrams. Now to be clear, I have seen UML class diagrams and UML case diagrams in the past, but I never actually made any myself. I was always confused about the different types of arrows used. However, now I finally know what these arrows mean. One thing that I didn't even know existed was Transition, Activity, Sequence, and State diagrams. Now I do, and I fully understand them.

While UML diagrams are fun and useful in general, they are not used outside the academic world. At least not to my knowledge. I have worked at many companies, and none have used UML diagrams. I think the reason for that is that they are super time-consuming and require a dedicated team to do them and update them since the programming team will be busy implementing them.

Additionally, I finished 90% of the Codio activities this trimester and have worked on every Unit activity since week 1. My assignments this trimester were fantastic. I enjoyed both the driverless car system design and the code implementation assessments. I also learned how to read academic papers. Previously I struggled reading academic papers, but this trimester helped me a lot in that regard.

Finally, I progressed on my master's degree thesis. Since I want to create an operating system from scratch on ARM, I did a good amount of self-studying on ARM64 Assembly. As a result, I learned many ARM-specific instructions, and I wrote multiple small applications in ARM64 Assembly.

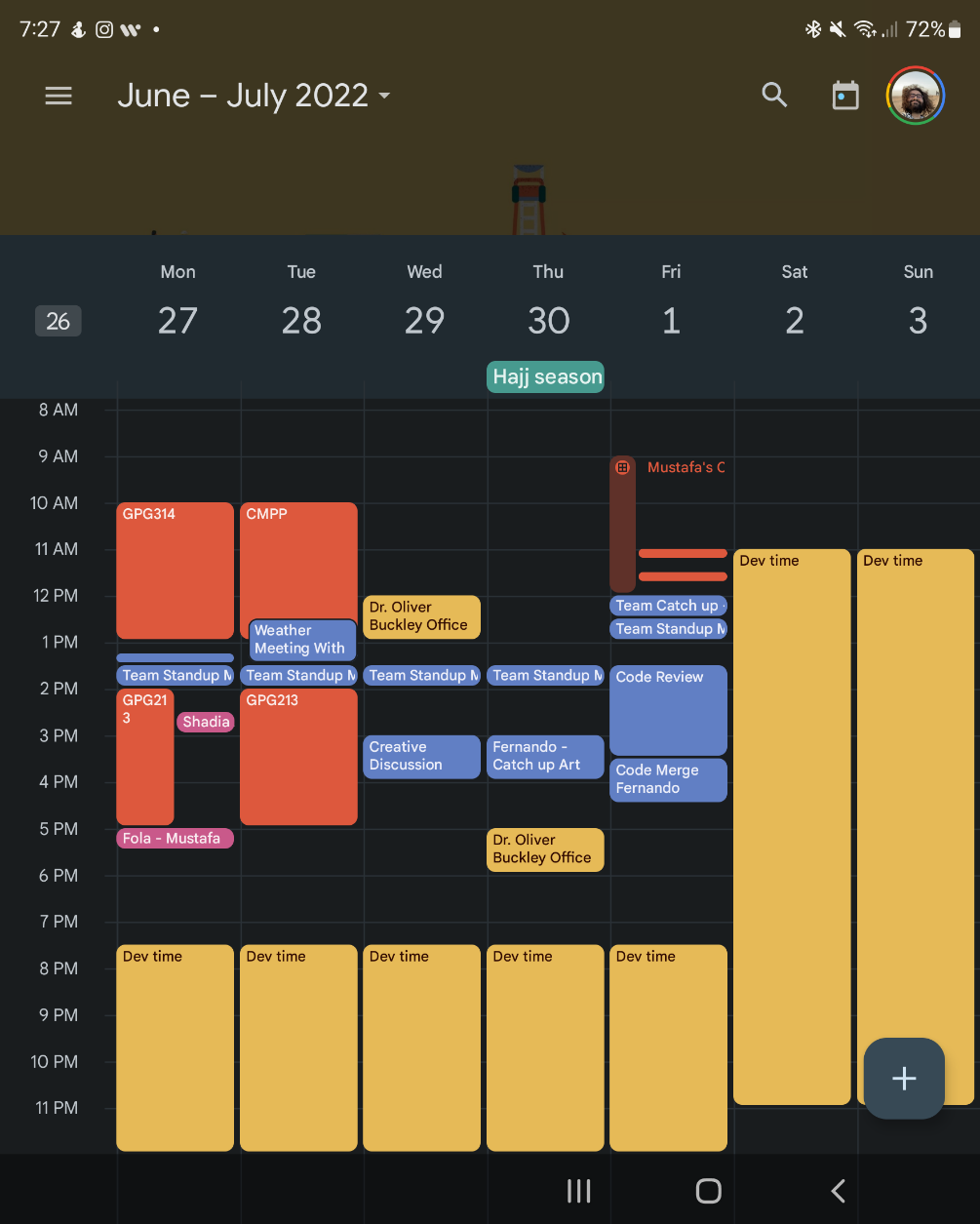
Hopefully, by year-end, I will be fully confident using ARM64 Assembly.

**How I used what I learned throughout this trimester (SO WHAT)**

Now that I have a good grasp of Python. I started teaching it at my local university, where I work as a part-time lecturer. It is easy to learn for students who have never programmed before.

As I mentioned, I started to incorporate more unit testing in my full-time job, which is fantastic but not as fun as adding new features.

One issue that kept popping up this trimester for me was the lack of time. I have my schedule set perfectly to the minute. I usually work seven days a week. Not because I have to but because I enjoy it. My schedule usually looks like this.



However, my schedule and time management have fallen apart because my visa was rejected two months ago. I had to run around chasing after my visa, and I had to cancel or postpone many of my meetings and work. The stress was getting to me. I think this might be, as they say, a blessing in disguise because I am starting to realize that I am taking on way too many things simultaneously. Full time job + part-time job + masters degree + freelance clients.

But to tell the truth, I enjoy having a busy schedule. I love what I do. Nevertheless, I have to dial it down just a little.

Besides that, I am making an Atari 2600 game using the 6502 Assembly language. I am at the early stages, but I am making good progress. The repository can be found at the end of this reflection.

**How will I use what I learned moving forward (NOW WHAT)**

Moving forward, I would like to make a few more changes.

1. Say no to people, have better time management, and account for unexpected events.
2. Use unit testing more in my personal project. After adding a new feature, I have found that having unit tests to run and test old code is invaluable. The amount of time I save by having automated testing is massive. Rather than test things myself, I will let the computer do it for me. Then, if something breaks, I know exactly what it is and where.
3. Design systems in a visual way first. After seeing the benefits of UML diagrams, I started to draw systems that I wanted to design on my tablet first before jumping straight into code. This way I can iron out any flaws in my design before writing the code.

In conclusion, this trimester has been fantastic. I learned a lot, and I am one step closer to writing my operating system.

Atari 2600 repo:

https://github.com/mustafa-sibai/atari-2600-6502-cpu-assembly-game

References:

Anon, 2020. Name mangling in Python. *GeeksforGeeks*. Available at: https://www.geeksforgeeks.org/name-mangling-in-python/ [Accessed September 5, 2022].