MODULE 8 Portfolio

Gregory Norris

CSU Global Campus

ITS320-4 – Turano

Module 8 Portfolio

**Lessons Learned**

There were a few good lessons learned in ITS320. This would be the third ‘programming 101’ course I have attended, but it is the first Python class I have taken. I had some experience with Python programming previously, mostly in terms of processing CSV files and some web browser automation for data entry. That said, a lot of what I have programmed in the past for Python has been a process of cut-paste solutions and a lot of stack overflow queries on python conventions. The Zylabs book used in this course definitely helped to clarify some of the basic components of a program, yet there were many topics it glossed over towards the ends where additional explanation would have been useful. For example, it’s four-page chapter segment on handling CSV files was definitely missing information regarding file flags (-a, -w, -r, +, -) and why one would potentially use dictwriter instead of writer. The segment on Numpy and plotting did exceed expectations, although it was likely beyond the scope of this class. What I would have liked to see in the Zybooks is more focus on designing either text-based user interfaces or simple graphical interfaces. This book did a good job of explaining the basics such as loops, iterators and string operations.

There is little I can do after taking this class than I was capable of prior; however, I would consider myself to be much more proficient in writing python programs than prior to taking this class. Actually, one thing the book helped to explain to me which I had some trouble understanding on my own is string slicing in Python. Another shift in thinking is in storing data as lists of classes being easier in Python, compared to storing data in a tabular multidimensional array like one would in C++ or Java. I also found being able to recast a variable as a different data type to be easier in Python than in any other programming language I have worked with so far, and this definitely opens up some opportunities for more readable computer code. The book was also good at clarifying programming conventions and style. I learned that it’s better to use four spaces than a tab for indentation just in case you are copy/paste code between different compilers. As for the code that I am turning in for my final project, there are a lot of things I would have done differently. For one, I would have declared the class attributes once and used an iterator for obtaining input instead of specifically declaring each variable. I think my final program probably could have been accomplished with about a third the amount of code used if I had spent a bit more time thinking about how to efficiently obtain user input. I likely would have used numpy instead or python’s built-in SQL service if I had spent more than two caffeinated nights coding, but the program meets the minimum requirements, and I suppose that’s what really counts.

I also found out that Python does not use prototyping, nor does it have a need for it. This seems a bit strange going from C++ to Python, where the main() method is called at the bottom of the program after initializing all of the functions. I personally like to forward-declare methods so I can keep the logic at the top of the program, but it’s only a minor complaint.

**Conclusion**

I certainly see myself continuing to use Python for its ability to rapidly construct working code with minimal effort. There are still a few Python fundamentals I can continue to work on, specifically error catching and class construction. At this point, I feel that I have enough of a base in Python programming to experiment with additional modules and understand how they work.