Lessons Learned Reflection

Dr. H. Anwar Ahmad

(ITS-320-3) – Basic Programming

Colorado State University-Global Campus

March 4, 2019

Lessons Learned Reflection

ITS-320 was my first experience with computer programing. In the beginning, I was frequently frustrated with my abilities. Simple homework assignments would take hours to get right. One of the most useful tools was reading my classmate’s discussion board posts. Robert, John, and Jenna continually impressed me with their knowledge, experience, and creative approaches to a problem. As the course progressed, somewhere around week four when we started talking about if/else logic statements, things began to “click” for me, and the ease of which I could express my ideas in Python expressions began to improve. I think the online learning environment made things more difficult for me than a traditional classroom would have. I would find myself having to search StackOverflow to answers to my questions if they were not related to the week’s discussion post. Without a good forum to chat about general issues, I frequently felt like I was on my own to try to figure out this abstract way of expressing ideas. Thankfully my hard work has paid off and I have been fortunate to receive a good grade so far.

**Modules 1 thru 8**

Starting out in week one, we started to learn that Python is an “object-based” language. We began to discuss the Python platform as well as “interactive interpreters” such as IDLE. We learned about thinking computationally and read about basic input and output as well as the different types of errors that are oh-so-common when learning how to make your own programs. Next, we learned about different types of objects such as strings, integers, floats, and how these objects can be put into “containers” like lists, dictionaries, tuples, and sets. In week 2 we were also introduced to how to use built-in modules in Python and practiced using functions within the math module. An area of the course I think could be improved would be to not wait till week eight to learn how to save and import modules! This would have saved me countless time copy/pasting line after line of code to call on something I had made in the past.   
 In week 4, we began to learn just how powerful, and how much creative freedom is available when making our own programs when we learned about branching if/else statements. I finally get the space versus tab debate I had learned about on Reddit before taking this course. I also learned that Bill Gates uses tabs, and as such, the dispute is now settled. Branching if/else are the backbone of how almost all software.

We often want to be able to "conditionally" do things in our programs - we want to be able to say "if this thing is true, then do X but if this other thing is true, then do Y." It's like when we wake up in the morning - "if it's raining outside, then I take an umbrella, but if it's sunny, I wear sunglasses." We can do things conditionally in our programs using if statements and if/else statements combined with conditional expressions. (KhanAcademy, n.d.)  
From branching, we moved to talking about loops. We learned about when it is appropriate to use a “while” loop, and when it is appropriate to use a “for” loop. We learned about how to make nested loops by using “inner” and “outer” loops, and how to exit the loop using “else” logic. At this point in the class things, the constant practice had gotten me to the point of near-comfort typing the archaic commands into the interpreter. I was beginning to start to feel pretty good about my coding skills when I got a “C” on the critical thinking assignment because my program wouldn’t load on the instructor’s computer. After reading the instructor’s example, I realized that my program was substantially similar enough to it, and took some screenshots of IDLE displaying my working code. The instructor took a degree of mercy on me and credited my grade to represent a functional program. I am grateful. I include screenshots with my work now for easy readability. I never did learn what caused that program not to work, but I assume it came from using IDLE instead of “Python-proper” to proofread my assignments. With confidence in myself restored, I went into week 6 to learn about the usefulness of functions, and how sometimes it makes sense to make “user-defined” functions instead of using the functions built into python. Again it would have been useful to know how to save and import functions into the interpreter. It was about this time we started working on Milestone One, and I was so frustrated with copy/pasting each line of code that I used StackOverflow how to learn about the sys.path folder.  
 Approaching the end of the course, we started a more advanced discussion regarding string manipulation, container modification, and classes. We learned about exception handling and how not to crash your program by unpredicted user input. Most recently, we read about working with classes and discussed how they can be used concerning our portfolio project. Finishing out the textbook was not required by the syllabus, but I found it prudent to continue to learn about things like recursion, inheritance and using files within Python. I have a 400 level programming course in my degree upcoming, and I feel like my time learning about these topics was wisely spent.

**Takeaways**

Python is more or less billed as a functional coding language that is easy for beginners to learn. The wonderful website Python.org (n.d.) describes some aspects of Python that were especially helpful for me:

Often, programmers fall in love with Python because of the increased productivity it provides. Since there is no compilation step, the edit-test-debug cycle is incredibly fast. Debugging Python programs is easy: a bug or bad input will never cause a segmentation fault. Instead, when the interpreter discovers an error, it raises an exception. When the program doesn't catch the exception, the interpreter prints a stack trace. A source level debugger allows inspection of local and global variables, evaluation of arbitrary expressions, setting breakpoints, stepping through the code a line at a time, and so on. (Python.org, n.d.)

Eight full weeks into this class I am more or less confident in my basic programming abilities. I am not intimidated looking at a block of code I don’t understand as much as I was in the beginning and have learned how to use the vast repository of resources available for free on the web. Sites like StackOverflow, Python.org, StackExchange, GitHub, and GeeksforGeeks have proven to be tremendously valuable in my quest to become a better coder. One of my favorite features in Python is “dynamic typing” it is so satisfying to watch your code come to life! I feel confident that this primary education in computer literacy will serve me in my career as an insurance adjuster so that I can improve my perceived value by presenting programming solutions to common problems that pop up during the workday. All things considered, I don’t think that I want to be a professional programmer, but the lessons and functionality that I learned I will take with me for the rest of my life to solve problems creatively, efficiently, and with practice, without too much frustration. Catch a fish for a man, and he will eat for the day. Teach a man to fish, and he will eat for the rest of his life. Thank you for teaching me how to fish.

References

KhanAcademy. (n.d.). Review: Logic and if Statements. Retrieved March 9, 2019, from <https://www.khanacademy.org/computing/computer-programming/programming/logic-if-statements/a/review-logic-and-if-statements>

What is Python? Executive Summary. (n.d.). Retrieved March 9, 2019, from <https://www.python.org/doc/essays/blurb/>