**Lessons Learned:**

Possessing no real programming experience outside of learning HTML back in 1997, I embarked on this journey to learn Python as I am looking to transition from a career as a Sales Representative into an eventual career as a Data Scientist. I made an extra effort to learn about the syntax and logic of Python as I have heard great things about it and knowing that I must know how to program in Python in order to be a successful Data Scientist. Along the way, I have learned many things in this course.

One of the first things that I learned is what an algorithm is, a series of step-by-step instructions to solve a computational problem, and why algorithms are needed. I also learned about flowcharts being a visual representation of the algorithm or series of step-by-step instructions. Each step is drawn as different shapes of boxes with lines depicting the logical flow by interconnecting arrows. Flowcharts are important to improve workflow processes, documentation, etc. Similarly, I learned pseudocode, a text-based basic outline of an algorithm that can be written into programming statements. Some of the benefits of pseudocode are as follows: simpler than flowcharts, language-independent, and are not represented visually like a flowchart.

Another important Python topic that I learned is the containerization of data via lists and dictionaries. One important difference between the two is that lists are indexed via integers while dictionaries are accessed via key-value pairs and are not indexed by integers. Dictionaries are also unordered while lists are ordered and can be indexed by integers. I had a little difficulty writing programs that iterated over the contents of a dictionary and finally realized that I could only iterate over the key and not the value of the dictionary’s contents. When I accessed the dictionary’s key, my program printed the key and the value, a tuple, associated with the specific key.

Looping in Python also proved to be very challenging for me as I delved into ***for*** and ***while*** loops. One thing that really helped me was knowing when to use one or the other: use ***for*** loops when I know how many items to iterate and use ***while*** when I don’t know how many times to iterate. I learned that I can use while loops with Boolean values: True and False. For example, **while True**, execute the following lines of code. I encountered numerous programming errors while writing nested while loops as I failed to increase the index or variable value for programs that required it.

Learning how to use functions helped me write code that reduced duplicate programming as I could create multiple objects by calling or invoking the function multiple times versus writing the same lines of code repeatedly. Furthermore, I learned how to use Python’s built-in functions (input(), int(), etc.) as well as custom methods to call later in my program. As for the naming of functions within my program, I learned to use function names like AddHome or add\_home versus one-word names that could be confused with variable names.

Writing classes and the accompanying definitions proceeding them helped me to understand the order of operations in terms of where to start writing a complex program that utilizes classes, definitions, and objects. One thing that I failed to do in writing my early programs was not initializing the class later in the program. Class objects must be initiated before they can be used in the program. Furthermore, I learned why class encapsulation is so important: restricting access to methods and variables prevents errors and data corruption.

Finally, I learned how to tie everything that I learned together in writing my final portfolio program. I had a few minor issues naming my class attributes, but I fixed them and ran the program as it should run. I am looking forward to continuing my Python education and learning how to query data using SQL and SAS software applications. I have had a very good experience learning Python this semester at CSU-Global Campus.