CSCE 240H Project 6 Report

Drew Dabe

This project is intended to be the cumulative final of the previous 5 projects. Over the course of the semester I have learned and implemented methods and techniques which have helped me to form the basis of a chatbot. The final product showcases the combined knowledge of code optimization, input and output, and reusability in code gained in the course.

One of the challenges faced was the progression and combination of our programs as we went from project to project. For me, my code went through several progressions and recreations throughout the semester. This even extended to the base of programming language, as I actually went through a few different languages depending on the project. I started off in C++ where I worked with some methods of file reading and understanding the input and corresponding outputs of what the program should be expected to do. It served as a singular function, though, and I soon moved on to working in Java. In Java, I took some of the same methods from the previous project but implemented classes to better store information and prepare a better output. When moving on to requiring better and more formatted output based on a user input, I was able to use extensive string manipulation to properly read various user inputs. This, however, grew to be difficult and tedious to modify, and its existing state was not in a position where it could meet the needs that I was wanting. So, I decided to move to Python and try to implement a more advanced method of parsing user input. More specifically, I used levenshtein distance to compare user input to expected inputs. At first, it started off a bit rough and imprecise. There were various challenges, which I'll go more into later, and it wasn't exactly what I wanted it to be. So, I moved to make the coding as easy as possible, by determining all the valid distances and

determining statistics like standard deviation to ensure that a matching query is valid enough to be an output. I also further improved the general reading and running of the functions to make it work as intended. At this point the program actually became a chatbot. I worked it so that the programmer can easily specify an expected input query and the respective output query of the chatbot so that eventually any amount of queries could be covered and the chatbot could answer as much as the programmer wants it to. There were some instances of the code where I used just excessive amounts of queries and many of which weren't actually needed or conducive to a functional chatbot, but I later scrapped it down to the bare bones of the parts and items of this assignment so that anybody wanting could easily implement or expand based on their own wants. In that sense, moving into the final project, I didn't try to make any major changes. Instead, I checked for success as a chatbot, I worked out bugs and kinks, added some additional functions and functionality that could be useful, cleaned it up, and tried to leave some extra comments. As a final product it came out quite refined in that it really isn't even that many lines, especially compared to a few of the inner projects I did. It helps with some of the readability and allows for more advanced work to be added without it being specifically work intensive. I figured this would be the best direction going forward, especially in the spirit of python, aiding the programming experience and making it easier to code.

There were various challenges faced when creating the different projects. There is, of course, the challenge of converting between languages. In terms of reading from the Form 10-K files, I change almost nothing between instances. I look for an all-caps PART or ITEM, and then I load the specific content into a class based on when it sees them (it will also go past the table of contents properly). If I had stayed in and refined my code I could have further refined this method to be most efficient. Then, there was the issue of being introduced to a problem and not

quite knowing how to solve it in the way you want since you just came from a different mindset. When I first moved to Python, it had been a while since I had last worked in the language, so the foundation became quite simple, with mostly rudimentary methods being used and even more time being used to figure out simple things like lists or basic errors that would otherwise be avoided. This led to my first python program, while technically meeting the basic requirements, being not very accurate, and generally not meeting my personal requirements for how I would want to submit an assignment. Furthermore, poorly written code can tend to be difficult to understand and even more difficult to reuse or expand upon. As such, in many of the project expansions, I would have to start over or reexamine the program from a ground level. However, this led to an overall more refined program, and as I got used to the experience, I got better at improving my code overall. So, despite the challenges, I was able to either grow from them or become further motivated to improve in the future.

As for work in the future, there would be various changes that I would be especially interested in changing. Starting from the classes, as the program works now, all of the files that are read are stored in memory entirely. This is, for two files, perhaps an acceptable solution, but if I wanted, for example, to hold infinitely many classes, it would be much better to hold the simple start and end line numbers from a file and then put off that work to the processor, especially considering that it would be minimally extensive to do so in a single query. Secondly, for the output, I'm not a huge fan of an entire part being spit out in place of certain key information that might pertain to the part and query. It could be possible for example, to look for certain phrases that match the user query in lines as we read through them and specifically read that out. To be fair, I'm no businessman, nor do I understand the logistics of reading through something like this. However, to me, it seems like if you wanted this kind of information it

wouldn’t be too much different from simply reading through the file yourself. So it is perhaps merely a limitation of the scenario that would leave me with some disrest. Lastly, I would work to make the code more like an artificial intelligence in the sense that it could grow and expand upon itself, or be able to code itself, as it goes. I could do this by moving the list of queries which are currently hardcoded into an external file which the program could read and write to. This would be more complicated, of course, but this would be most effective in terms of how this tactic is done, and how the program would work and evolve in the future.

As for reusing others' code, though I did take some considerations, I ended up not taking from any classmates. For general purposes what I was trying to do here was somewhat different than the general goal of the class, and I felt that it would be best in keeping it simple, keeping it connected, and keeping it refined, to condense it all into something I'm familiar with. Still, in the future, especially considering the amount of people that were using python, which was my final language, I would want to try and talk with some of the students to see if we could correspond issues and work better to use the classmates themselves as a resource in starting from scratch which was, at times, what I was doing. Then we could improve each other's code and would further be more encouraged to use that code as well.