Throughout my time in my computer science program, I have learned a lot about the discipline and developed many skills. I have learned about different programming languages including C++, Python, and Java. I experienced using collaborative work environments and techniques like Git and code reviews. There have also been times where I needed to deploy these skills and develop new ones like linking Python to Mongodb for database management. I have had many ups, downs, tight deadlines, struggles, and victories throughout my journey. These experiences have allowed me to grow and become skilled and knowledgeable in my field.

One area that I have grown in is my ability to work in a collaborative environment. I have used platforms like Git to foster collaborative work with classmates and fellow computer scientists. I have also, from time to time, performed code reviews for myself and for others. These code reviews are important in a team environment to help catch bugs and design flaws that can then be corrected through code improvement. A best practice I have picked up and refined over my time is commenting. I used to be unsure of exactly what made a good comment, but through study and practice my inline communication has vastly improved. This can be seen when comparing my original PokemonBattler.py to the upgraded version of it or my Farkle game. I am capable of clearly articulating what my code accomplishes, and the logic behind it.

There have also been times where I have had to focus on software design throughout my time in my program. When I had originally envisioned my Farkle game for example, I figured it would be simple to just allow users to pick dice that they wished to set aside. This posed several problems however in that there was a lot of room for error, and was vulnerable to bad input crashing or maliciously affecting the program. I got around this by implementing a complex algorithm to allow the user to select their scoring options to remove much of these error chances. All while developing this system I continued to use all of the features offered by C++, while directly confronting limitations through the use of features like vectors and pointers. When I had my second crack at the program by porting it to Python, I created an even more elegant version that kept all the features of the original, but used what Python offered to slim the program down even more.

I also had chances to demonstrate my skills with personal work I made. My previously mentioned Pokemon battler was a massive undertaking in order to recreate the real game experience of battling through my own code. I spent a lot of time planning out the steps of rolling out features to the game. It started with creating classes to represent the game actors of Pokemon, trainers, moves, type match ups, and so on. From there I developed selection systems for users to select their characters to participate in the game and allow them to battle automatically to ensure functionality. After this I implemented several options for users to select battle actions, allowing for play for the first time in the program’s life cycle. Finally, with my enhancements for my capstone I have now implemented the status conditions and efficiency improvements to make for a more engaging experience. Each cycle through the design allowed me to add more features and refine the ones that came before it. With my enhancements, I was able to create a more professional piece of software, even though I created it just for fun and practice.

One skill set that I developed later in my program was considering design flaws that could lead to vulnerabilities in my software. There are some things that we take for granted now like usernames and passwords that are a vital component of software security. One piece of information that stuck with me though was when I learned that the most vulnerable point of software is when receiving user input. This part of most software can allow users to inadvertently or intentionally crash a program or cause unexpected behavior. I now know that I should use safer data types for user input, and to properly validate and account for mistakenly entered input. Some other areas that I learned about besides input streams are file streams and connections to servers. My Farkle game ensures to close the rules text file before moving on in both versions. My animal shelter script has to account for being connected to a database stored on a server. On top of making sure the connection is closed, I had to use Javascript’s asynchronous functions and await keyword to handle server promises and see if they would be rejected or fulfilled. Each process of enhancement allowed me to apply many of the lessons I have learned to these programs.

Throughout my time in my program and in this final class specifically, I have demonstrated skills that, while not related to computer science specifically, will serve me in my professional career. From this class alone I suffered technical issues that set me back three weeks, giving me two weeks to complete almost the entire ePortfolio. I buckled down and finished my enhancements for Farkle and the Pokemon Battler in the span of three days. I had a drive to meet my goals and complete my program, allowing me to push forward and complete these works. My Animal Shelter artifact took longer, but this was because I had to learn Javascript which I had never used before. On top of that, I had to relearn how to interact with Mongodb and learn how to handle connections to active servers. It was difficult and there were many struggles, but I managed to pull through and complete my work. This demonstrates that I have a resilient mindset, desire to learn new skills, and that I can buckle down and focus on clean and efficient coding and design. I am glad to say I put in a lot of great work for the span of just two weeks, and I look forward to continue growing as a computer scientist.