**4-2 Milestone Three Narrative**

Corey Sampson

Southern New Hampshire University

CS499: Computer Science Capstone

Professor Jeff Phillips

July 25, 2025

**4-2 Milestone Three Narrative**

The original artifact I selected was a course scheduling program written in C++ as part of the CS 300: Data Structures and Algorithms course. It was developed to load a list of college courses from a text file and allow the user to view available courses or search for a specific course. The initial implementation utilized vectors and basic string parsing to load data; however, it lacked modularity and did not support viewing recursive prerequisite chains.

I selected this artifact for enhancement because it represents foundational knowledge of algorithmic and data structures that is critical to my development as a software engineer. The original implementation demonstrated basic file input, data storage in a vector, and conditional logic. However, it was limited in structure and functionality. To better align with real-world expectations and course outcomes, I enhanced the project by rewriting it in Python. I designed a **Course** class to encapsulate course data, including IDs, names, and a list of prerequisite IDs. I used a dictionary data structure for efficient key-based access to courses and implemented a recursive algorithm **(resolve\_prerequisites**) that traces prerequisite chains for any given course.

This enhancement shows my ability to apply object-oriented design for modular, maintainable code, choose appropriate data structures (dictionary over vector) for efficiency, implement recursive algorithms for depth-first traversal of dependencies, and perform input validation and handle file-related exceptions. The enhanced project is not only more readable and scalable but also includes a user-friendly menu and additional logic to prevent infinite looping caused by circular dependencies.

**4-2 Milestone Three Narrative**

I have met the course outcomes planned in Module One. Specifically, this enhancement aligns with the course outcomes for designing and evaluating computing solutions using algorithmic principles, which was achieved by evaluating the limitations of the original solution and developing a more efficient recursive function to resolve course prerequisites. I utilized innovative tools and practices by transitioning to Python and using dictionaries and recursive functions, and adopting modern, industry-relevant tools and patterns. Lastly, I developed a security mindset, even though this project is not security-centric, I added error handling for file access and input validation, reflecting secure and defensive coding habits. At this time, there are no planned updates to the outcome-coverage plan as the current enhancement supports the outcomes I intended to meet.

Enhancing this artifact provided me with a valuable opportunity to revisit algorithmic design in a different language context. While the logic itself remained straightforward, I learned how to better separate concerns using object-oriented design and how to leverage Python’s built-in features to streamline development. I also gained confidence in working in VS Code with Python for the first time, which expanded my comfort zone beyond Visual Studio and C++. One of the biggest challenges I faced was ensuring that the recursive function to trace prerequisites handled edge cases like nonexistent course IDs or circular prerequisites, without crashing. Another challenge was rethinking the program structure from a procedural C++ implementation to a modular Python program with clean, readable code and helpful inline comments and docstrings. Through this process, I became more aware of how thoughtful data structure choices (like using a dictionary instead of a vector) can significantly improve the performance and

**4-2 Milestone Three Narrative**

usability of a program. The experience reinforced the importance of clear documentation, good code structure, and error handling, which are key skills in professional software development.