[CS-499-19912-M01 Computer Science Capstone 2025](https://learn.snhu.edu/d2l/home/2019781)

3-2 Milestone Two: Enhancement One: Software Design and Engineering

Eddy Kwon

09/21/2025

**Artifact Description**

The artifact I selected for my first enhancement in software design and engineering is the **Advising Assistance Program** from CS 300. This program was originally created to read a CSV file of computer science courses, load the data into memory, and provide a simple menu for the user to list courses or view details of a specific course, including prerequisites. The project was implemented in a single C++ file and used data structures such as unordered\_map and vectors.

I created this artifact during CS 300, and I chose it because it provides a strong example of software design principles applied to data organization, string handling, and user interaction. At the same time, it had areas for improvement that I could address as part of my enhancement.

**Justification for Inclusion**

I decided to include this artifact in my ePortfolio because it demonstrates skills that are directly tied to software engineering, including:

* **Data modeling**: structuring course information with course numbers, titles, and prerequisites.
* **Algorithmic efficiency**: using a hash map (unordered\_map) for O(1) average-time lookups.
* **User interaction**: creating a menu-driven system with input validation.
* **String processing**: implementing helper functions to normalize input and prevent errors.

This artifact also highlights my ability to not only build a working solution but also to identify weaknesses and improve the design. Employers value the ability to write clean, maintainable, and reliable code, and this enhancement shows my growth in that area.

**Improvements Made**

In my enhancement, I made several changes to improve the quality and reliability of the artifact:

1. **Input Validation**
   * The original program did not properly validate menu input or course codes.
   * I added stricter checks so invalid or empty entries are handled gracefully.
2. **String Normalization**
   * I enhanced helper functions like canonCode, ltrim, rtrim, and stripBOM to ensure course codes are stored and compared in a consistent format.
   * This prevents problems with spacing, capitalization, or hidden characters.
3. **Error Handling**
   * In the loadCourses function, I added warnings for invalid lines in the CSV file.
   * The program now outputs helpful error messages if a file cannot be opened or if a course code is missing.
4. **Code Organization and Readability**
   * Improved comments and structure throughout the file.
   * Used consistent naming conventions and modular functions to make the code easier to follow.

These enhancements improve the robustness, usability, and maintainability of the program, aligning with best practices in software design and engineering.

**Outcomes Achieved**

In Module One, I planned to demonstrate the following course outcomes:

* **Software Engineering/Design**: showing the ability to implement enhancements that improve clarity, maintainability, and error handling.
* **Algorithms and Data Structures**: ensuring the artifact uses efficient lookups with unordered\_map and sorting with std::sort.

Through this enhancement, I believe I achieved these outcomes. My program is now more resilient to bad input, more user-friendly, and written with cleaner software design practices.

**Reflection on the Enhancement Process**

Enhancing this artifact gave me the opportunity to practice real-world software design techniques. I learned the importance of **defensive programming**, such as validating every input and handling unexpected file errors. I also reinforced the value of **string normalization** to avoid subtle bugs when dealing with external data like CSV files.

One of the challenges I faced was ensuring that duplicate or invalid course entries in the CSV file were handled properly. I had to add checks and warnings so that users would understand when data was not formatted correctly. Another challenge was balancing simplicity with robustness: the project must remain a single file for the assignment, but I needed to modularize the logic to keep it clean and readable.

Overall, this process showed me how even small improvements in code quality can make a significant difference in usability and maintainability. I now feel more confident in applying these principles to future projects.

**Conclusion**

This enhancement demonstrates my growth in software design and engineering by improving a functional artifact into a more reliable, maintainable, and professional-quality program. It shows that I can identify weaknesses, implement solutions, and reflect on the development process to learn from it.

Including this artifact in my ePortfolio will allow me to showcase my ability to design, implement, and refine software solutions, skills that are essential for success in the computer science field.