Nick Silvestro

CS-499 Computer Science Capstone
July 28, 2024

Category Two Enhancement Narrative

Briefly describe the artifact. What is it? When was it created?

The artifact selected for the category of algorithms and data structures was the course planner project from CS-300 course (Data Structures and Algorithms). The projects focused on aiding the "ABC University" client on developing a program to aid academic advisors in assisting students. The program loads a file of course information, in this case a list of computer science courses, and allows advisors to print schedules, or search for a specific course. Ultimately, this project was about determining appropriate data structures for a client's needs. This artifact was created roughly halfway through the degree program.

Justify the inclusion of the artifact in your ePortfolio. Why did you select this item? What specific components of the artifact showcase your skills and abilities in algorithms and data structure? How was the artifact improved?

Initially, after reviewing the category title, it felt fitting to review the data structures and algorithms course projects. I selected this artifact as it strongly aligned with a specific course outcome. To align with the outcome, creating algorithms with selected data structures lends to solving problems using algorithmic principles, while more importantly reviewing efficiency and cost will involve reviewing and weighing trade-offs (such as worst cases, best cases, total cost, etc.).

Overall improvements include a focus on efficiency: reviewing and selecting the data structure and the responsible code for the data structure, as well as maintaining the code structurally by moving repetitive code to individual functions. General variable and method naming, and documentation are also addressed. Lastly, the addition of input validation and the

assurance of proper memory use in allocation and deallocation will improve the security of the artifact.

Did you meet the course objectives you planned to meet with this enhancement in Module One? Do you have any updates to your outcome-coverage plans?

The main objective of designing using algorithmic principles and evaluating solutions, managing trade-offs is accomplished through the deliberation of factors and selection of an appropriate data structure, underlying structure, and the creation of an algorithm to support the use of the structure.

Developing a security mindset was invoked in a form where assurance of proper memory usage to avoid potential leaks was ensured, as well as ensuring proper input validation.

Reflect on the process of enhancing and modifying the artifact. What did you learn as you were creating it and improving it? What challenges did you face?

The documentation throughout the code was cleaned up to follow more consistent naming conventions and to include more adequate descriptive information.

Aspects of the code were restructured, such as creating individual specific functions from repetitive code that could be separated (ex. the menu of options). Additional methods were created and documented as well. These include the display course outlook, file path checker, duplicate file checker, and course key format check functions. The display course outlook function was created to bring additional functionality to the program in general. This method allows a user or advisor to view the information more meaningfully for some classes, allowing them to detail which future classes will be benefited from a current class.

The file path checker was created with security in mind. This function aims to validate the file path provided from the user, checking that the proper extension is provided as well as validating the existence of the file before attempting to load in subsequent lines of code. One

challenge I ran into here was one of my own logical errors. I had attempted to pass the csv path to the function to validate these aspects. Initially, I was attempting to alter the csv path in a function's scope without returning the altered path. I restructured the code and locations in the code where user input was accepted to remediate this.

The duplicate file checker was created to notify users when they were attempting to load a file that had already been loaded. The course key format checker aims to validate the course number provided by users during various menu options. This function verifies the input's length and alphanumeric pattern ensuring that they match the course numbers loaded. One challenge I encountered here was the decision to use manual format checking or to use regular expressions to verify the format. In weighing the cost of the speed of the two, information that I could find on the topic tended to lean towards the conclusion that regular expressions can be more costly to time. This is a topic that I wonder if you may be able to provide some feedback on.

Additionally, the time complexity and cost of the data structures were weighed again. The use of the hash table seems most efficient and consistent. I would continue to recommend using the hash table for the required functionality of the course system. Beginning with the best-case runtime for creating and storing course objects (as the worst case was the same for reviewed structures), the hash table is able to insert more efficiently than the tree and more consistently than the vector through the use of the hashing function, as opposed to a pushback. The use of an efficient hash function on a unique identifier (such as a course number) in combination with a chaining approach would lead to a sorted table as well. Lastly, a worst-case search of a hash table is comparable to the vector or tree (linear), however the average search for a hash table exceeds the alternative choices.

The responsible code was made efficient through the use of a hash function with a large enough table size that collisions should be avoided with the task at hand, while minimizing the need for constant resizing (although dynamic resizing can be achieved) and avoiding unnecessary use of memory overhead. The chosen use of the underlying vector structure with the hash table, as opposed to an array, is due to the anticipated ability to continue adding more course data. The vector will allow for dynamic resizing where the array would be static.

Resources:

Aayushi2402. (2023, March 28). *Applications, Advantages and Disadvantages of Hash Data Structure*. GeeksforGeeks. https://www.geeksforgeeks.org/applications-advantages-and-disadvantages-of-hash-data-structure/

Arora, D. (2024, April 22). *Understanding Time Complexity with Simple Examples*. GeeksforGeeks. https://www.geeksforgeeks.org/understanding-time-complexity-simple-examples/

Utkarshpandey6. (2023, August 9). *Time Complexity and Space Complexity*. GeeksforGeeks. https://www.geeksforgeeks.org/time-complexity-and-space-complexity/