Dylan Killman

3/30/2025

**Enhancement Two Narrative**

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

The artifact that I selected for this category is my final project, the Course Planner program, from CS-300: Data Structures and Algorithms class, which was originally created in the spring of 2024. The program is relatively simple; it’s a command-line-driven course planner application written in C++. Users can load a list of courses from a .csv file, display the full list of courses in alphanumeric order, or search for specific details about any course, including any prerequisites required for that course. I’ve also added a feature that displays the time it took for any process to complete in microseconds. The original version of this program, which I submitted when I took CS-300, used a vector to store the course information and utilized the sort() function from the standard library to alphabetize the data every time it was requested.

1. **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 structures? How was the artifact improved?**

I chose this artifact because it’s a relatively clean, simple, yet functional example of working with data structures. I figured the best artifact for the Data Structures and Algorithms category was the final project from the Data Structures and Algorithms class. I immediately thought of this artifact because I remember that back when I originally created the program, I purposely chose to use a vector because it was simpler to code, and I didn’t really have a fantastic grasp on data structures back then. This enhancement required completely overhauling the program, including designing and implementing a new CourseNode structure, coding recursive Binary Search Tree (BST) operations, and replacing all of the sorting and searching code with a BST-based traversal. Also, I added the timing functionality using the chrono library in order to showcase the performance difference between the vector and BST versions of the program. I believe these enhancements demonstrate my ability to employ algorithmic thinking, analyze performance differences, and assess and apply trade-offs between different concepts, specifically between various data structures. I also created a larger dataset with 1000 entries, instead of the original dataset's eight entries, so that the performance improvements could be measured and observed. With this, the BST provided a significant improvement in performance, showing that as the dataset gets larger, the algorithm becomes more efficient.

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

I originally planned to meet Course Outcome #3 (designing and evaluating computing solutions using algorithmic principles while managing trade-offs) and Course Outcome #4 (demonstrating techniques, skills, and tools in computing practices), and I believe that I fully met both of these. For Course Outcome #3, I analyzed the existing vector-based solution and saw that as the datasets get larger, the mechanism becomes slower and less efficient because of the very nature of vectors, and found a better solution to the problem. For Course Outcome #4, I applied data structure design principles and algorithms to create a functional BST-based program instead of an inefficient vector-based program, which demonstrates my understanding of tools and techniques to solve real-world computing problems. I currently have no changes planned to my outcome coverage plans.

1. **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?**

During this enhancement, I improved my understanding of both algorithmic design and data structures. Even though the BST may be a little more complex and harder to implement and write code for, I think the advantages and trade-offs are worth it, especially when you start working with large datasets. One of the most educational parts of this enhancement was working through the process of implementing and measuring time. I had to choose microseconds because if the time were in seconds, the difference in performance wouldn’t be measurable. I learned that this strategy can and should be employed in almost everything I do that’s code-related. It’s very valuable to write something one way, test it, and then write it a different way and test it and measure the difference in order to select the most efficient thing. One challenge that I faced during this enhancement was the BST implementation itself. I was never really great with BSTs when I took the original class, so at first, this was a tall task for me and my current skills. Getting the search time down to O(log n) from O(n) seemed like a foreign language to me at first. And using insertion instead of push\_back wasn’t something that I had a lot of practice with.