Lawrence Linnell

January 31, 2021

Southern New Hampshire University

CS 499 T3254 21EW3

Milestone Three: Enhancement Two

Algorithms and Data Structures

The artifact chosen for enhancement is from the final project of IT-145 which is an application for a luxury cruise line company. This application is a good candidate because it was created when I had very little development experience. Since that time I have studied various languages, algorithms and data structures which will allow me to enhance the artifact to improve the user experience as well as application process efficiency.

To showcase my ability and understanding of algorithms and data structures I developed a quicksort algorithm which sorts the objects in a Java ArrayList lexicographically based on the objects shipName attribute. When a user chose to display ship names to the screen a list would be populated but would not be sorted. This is fine for a small database of names but makes it very difficult for the user to find a ship in the system if the database grows larger. The quicksort algorithm was chosen based on its proven reliability and performance with an average time complexity of O (n log n).

The quicksort algorithm that I designed and implemented illustrates my ability to choose an algorithm that is efficient while also considering system resources. This algorithm also showcases my ability to access object attributes and use that information within conditional logic statements to perform an in-place sort. The implementation also helps to showcase that I am capable of taking commonly understood algorithms, customizing them and extending their functionality for proprietary software.

I did meet the course objectives that I planned to meet. Further functionality that I plan to implement is to extend the quicksort algorithm to sort other custom data types. In this program it solely sorts the shipList ArrayList of Ship objects. I am considering either implementing an opaque pointer in the quicksort algorithm to minimize on method overloading or potentially create a new template class that the other objects inherit abstract methods from with similar method call names and then update the quicksort method to be more generic to accept these additional data types.

While enhancing this artifact I have refreshed my memory and skills when using the Eclipse debugger. I ran into a buffer overflow problem which set me back quite a bit because I was very rusty with Eclipse’s debugger. After reviewing some resources on the internet, I was able to isolate the error. It was somewhat time consuming trying to find a good instructional source to learn how to use Eclipses debugger due to a lack of tutorials with clearly spoken English.

This was a good exercise for me to revisit recursion and divide and conquer sorting algorithms. I plan to review the algorithms and data structures that I have studied in the past. A lot of my previous work with algorithms and data structures is written in C, so I also had to learn some new syntax to implement a Java version of quicksort. Overall, I thought the Java version is much easier to code and interpret.