Alexa Szlykowicz

Module 4

Milestone Three

Enhancement Two: Algorithms and Data structures

Narrative

The name of the artifact “VectorSorting\_Updated.” This artifact is a small program that takes in data from a csv file full of auction bids and performs a selection sort and a quicksort. The name of the file that was enhanced was Vector\_Sorting.cpp. The bids are loaded from a csv file created by the client. The interface includes a menu where the user can upload bids, view bids, selection sort the bids, or quick sort the bids. I created this artifact in CS 300 Data Structures and Algorithms.

In this artifact I demonstrate optimization, time complexity (BigO), as well as efficiency of the algorithmic logic in the code, comments, and narrative by replacing the selection sort algorithm with the quicksort algorithm. The quicksort algorithm is more efficient because as the quantity of the input grows, the runtime declines slower than selectin sort. The key component that makes quicksort’s average-case performance better than selection sort is that it splits the array into sub arrays that are solved independently. I selected this item because of the use of data structures to sort the given vector. This project also showcases my use of the selection sort algorithm and the quicksort algorithm. I showcase my skills and abilities in algorithms and data structure by implementing a partition algorithm that splits the vector according to the chosen pivot. The partition algorithm is then applied in the quicksort method.

Course Outcome:

* Design and evaluate computing solutions that solve a given problem using algorithmic principles and computer science practices and standards appropriate to its solution, while managing the trade-offs involved in design choices (data structures and algorithms).
* I have met this course outcome. I showcase the use of algorithm principles and computer science practices by using two known algorithms, selection sort and quicksort, to solve a problem of sorting thousands of bids. I manage the trade-offs involved in design choices by utilizing a quicksort to improve the time complexity from O(n^2) to O(n log(n)), thus enhancing its overall performance.

One thing that I learned while developing this project are the principles of time-complexity. I had to really take the time to relearn the quicksort algorithm. I had to take a step back and understand the recursive process of this algorithm. I struggled with creating the partition method. It took me some time to understand the logic and how to nest while loops to implement this algorithm.