Category Two Artifact Narrative – Data Structures and Algorithms

Miguel Little

CS-499 Computer Science Capstone

Professor Krupa

08/15/2024

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

I selected my Vector Sorting project from my CS-260 Data Structures and Algorithms class. It imported bids from a local CSV file and allowed the user to sort them using quicksort algorithms. This project was created about two years ago now.

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 software development? How was the artifact improved?

I included this artifact in my ePortfolio because the project allowed me to demonstrate an understanding of algorithms and data structures. I was able to build new sorting functions into the already converted Python application and implement time complexity into the functions.

I showcased my skills and abilities by creating structures and functions in Python and adding a "add a bid" function. I also implemented time complexity for sort functions and a new "sort by title" function that utilized a bubble sort algorithm.

Error handling is now in full swing. Article ID must be UNIQUE and a digit, article title must not be NULL, amount entry must be a REAL number, and the date must be MM/DD/YYYY formatted.

Time complexity is now displayed on the GUI instead of only showing on the console. Unit tests have been added, I am working on optimizing the second unit tests for future implementation.

Each sorting function uses a different sorting algorithm. SQL Lite has been used throughout the application.

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 first outcome met: *"Design, develop, and deliver professional-quality oral, written, and visual communications that are coherent, technically sound, and appropriately adapted to specific audiences and contexts."*

I met this outcome by properly implementing the new sort functions and the new bubble sort algorithm for the title sort. I updated the header for the main.py file and commented out the new lines of code. Mergesort is used for the date and error handling is in place to prevent users from adding bad entries.

The second outcome met: *"Demonstrate an ability to use well-founded and innovative techniques, skills, and tools in computing practices for the purpose of implementing computer solutions that deliver value and accomplish industry-specific goals (software engineering/design/database)."*

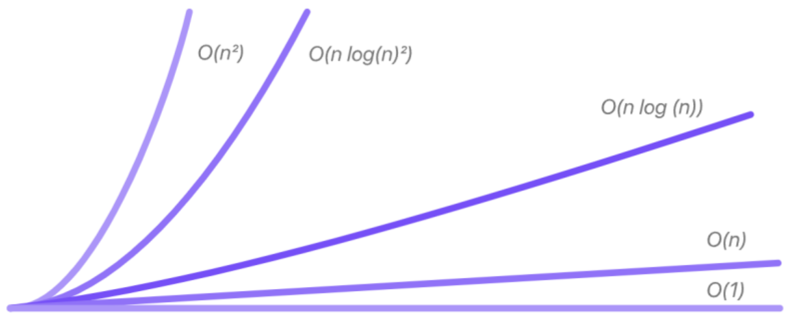
I met this outcome by adding the new sort-by-parameter functions and using the bubble sort algorithm for my “sort by title” function. Implementing time complexity accomplished industry-specific goals as it is commonly used. This application is reusable in the industry as well.

The third outcome met: *“Employ strategies for building collaborative environments that enable diverse audiences to support organizational decision-making in the field of computer science.”*

I met this outcome by uploading my project to GitHub, updating the headers, and commenting out the new lines of code.

The fourth outcome met: *“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 met this outcome by implementing the bubble sort algorithm for the new “sort by title” function and implementing time complexity to all the sorting algorithms. Each algorithm shows best- and worst-case scenarios and can be tested in the application itself.



This chart displays the best, middle, and worst-case scenarios for bubble sort, where n represents the number of items in the array list. This is a very efficient algorithm that was perfect to use. Merge sort and Heap sort are also very efficient.

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?

I felt lost at the beginning of this enhancement, the main thing I learned was with persistence any updates are possible. It just took countless hours of trial and error.

The sort-by-parameter functions returned “nan” and I did not know how to fix it. I found that I was mixing up float and int for the values so that was causing problems. The sort-by-date function was not working because I had to convert them into date time objects. Each sorting function was displaying 0 seconds and I thought the logic wasn’t correct or that I implemented time complexity wrong but it ended up working just fine.

The initial creation of my unit tests added bad data to my database which broke my sorting algorithms, so I had to delete the database, recreate the unit tests, and that worked.

References

Sorting Algorithm - Big-O. (2020). Big-O. https://big-o.io/