Milestone Three: Algorithms and Data Structures

David Garcia

CS 499 Capstone

Prof Brooke

September 25, 2023

Algorithms and Data Structures Narrative

Artifact

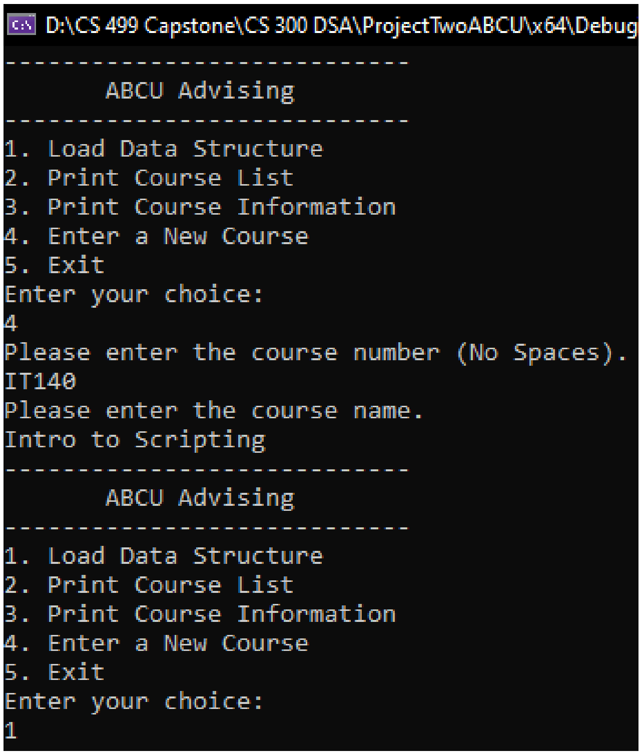
This artifact is from the course CS 300 Data structures and algorithms which I took a little over a year ago. DSA taught me the fundamentals of well, data structures and algorithms. This course was a very important one throughout my career here at SNHU. The artifact itself was the final project for that course. It allowed me to show my proficiency with vectors. I was given a text file that was prewritten with course and my responsibility was to create a vector and populate that vector with the courses from the text file. The user was able to load the courses into the vector, print the course list, and search a course name with the given course number.

Why I Chose This Artifact

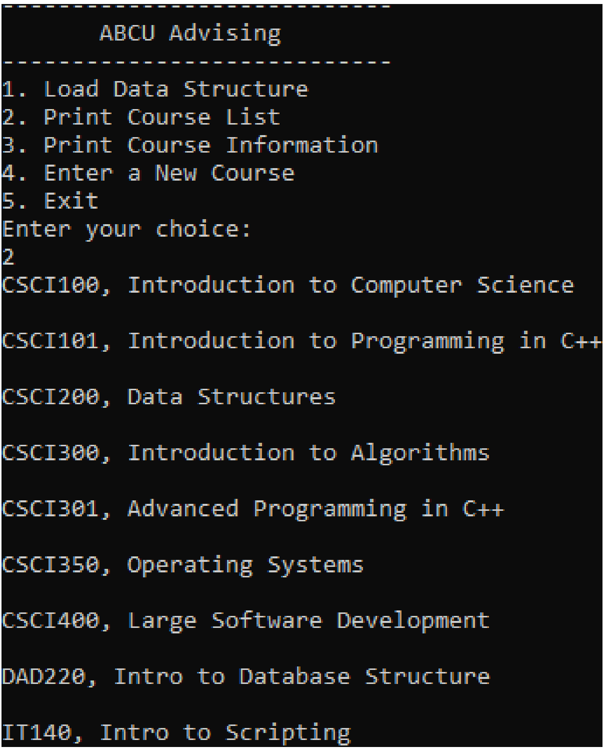
I chose this artifact from CS 300 because there is no better way to showcase my proficiencies with data structures and algorithms than from the course that educated me on them. The specific components of the artifact that showcase my skills include the creation of the vector itself, how I was able to insert information into it, utilize that populated vector, and incorporate it into a small program. This artifact was improved by implementing code that allows the user to add a course via user input of course number and name. Then the newly added course would be part of the course list/vector to be searched up and displayed.

Course Objectives

The course objectives of using algorithmic principles and applying my knowledge, skills, and available tools to solve a computing problem was met with this artifact enhancement. I successfully implemented the ability to add courses to a text file and into a functioning vector. The user is able to input any course they wish so that it will be displayed as part of the course list. After implementing this insertion algorithm, I was able to identify that it did not impact performance or speed of the program. The time complexity, Big O notation, of this algorithm would be O(1) because it does not impact the performance of the program regardless of the accepted input. With that said, I was able to design and code a compact insertion algorithm that did not affect program performance, while greatly improving the ABCU course advising program from its original state. Lastly, security was implemented with input validation when selecting a menu option. The screenshots below show the process of inserting a custom course, loading the vector up with the new information, and listing the updated course list.



```
D:\CS 499 Capstone\CS 300 DSA\ProjectTwoABCU\x64\Debug
----------------------------
       ABCU Advising
----------------------------
1. Load Data Structure
2. Print Course List
3. Print Course Information
4. Enter a New Course
5. Exit
Enter your choice:
4
Please enter the course number (No Spaces).
IT140
Please enter the course name.
Intro to Scripting
----------------------------
       ABCU Advising
----------------------------
1. Load Data Structure
2. Print Course List
3. Print Course Information
4. Enter a New Course
5. Exit
Enter your choice:
1
```



```
----------------------------
       ABCU Advising
----------------------------
1. Load Data Structure
2. Print Course List
3. Print Course Information
4. Enter a New Course
5. Exit
Enter your choice:
2
CSCI100, Introduction to Computer Science

CSCI101, Introduction to Programming in C++

CSCI200, Data Structures

CSCI300, Introduction to Algorithms

CSCI301, Advanced Programming in C++

CSCI350, Operating Systems

CSCI400, Large Software Development

DAD220, Intro to Database Structure

IT140, Intro to Scripting
```

Reflection

The process of enhancing this artifact was a difficult one but it has taught me a lot about perseverance and determination. If I am stuck, I can research concepts, solutions, troubleshooting and more to aid me in solving any problem. The first big challenge I faced when working on this artifact was that when I attempted to insert a course with user input, it would overwrite the whole text file instead of adding to it. So, I did some research and found the solution of appending user input to an already written text file. The next challenge I faced was that the user input was not being fully read, the first word of the course name was always being cut off. Cin.get() and getline() fixed that problem.