Lab 1: Sort Time Comparisons

Valentino Lei'a
CS_124
Fall 2020

Purpose:

I found the purpose of this assignment to be a cumulative review of c++ concepts. Some of those being:

- Using stream objects (fstream, ifstream, ofstream)
    - File handling (reading, writing, and cleaning the parsed data)
- Printing to the console with formatting (iomanip, iostream, printf if you prefer c style)
- Error handling or input validation beit try throw catch blocks or just while loops with conditionals to get the right data from an end user
- Problem solving using different types of loops (range based, for, while, do while)
    - Also using recursion if applicable
- Introduction into thinking about data structures
    - What is the appropriate way to store the data
    - What is the appropriate way to interact with the data
        - Dynamic memory allocation
        - vectors
        - Presentation
        - changing/modifying
        - Deleting
    - Sorting and searching data in memory
- Program structure
    - Thinking about separate compilation
        - Most efficient way to create reusable/portable functions/classes
    - Thinking about object oriented programming
        - If encapsulation was necessary/beneficial
        - If inheritance was necessary/beneficial
            - I tried using this on one class I think but in this application I didn't find it to be as useful as if I were building a different application that described real life objects
- Reviewing and leveraging STL and other libraries that would have helped programming this lab

Additionally, I found the purpose of the assignment to be pretty cool and end to end. I've noticed in many other classes there are small assignments that are very focused and drilled down on a specific concept, but this assignment was really geared toward an end to end application. Starting with user input and addressing that user, then moving onto manipulating data and doing some computation and printing it out to the end user. Then interacting with them to let them drive the outcome of the program. I felt it was pretty cool to have an all encompassing assignment (in terms of intro programming that is) that gave me more opportunities to practice things I wouldn't have had a chance to on an assignment more focused on a particular concept.

Plan & Organization:

- **What input does the program use? Why?**
    - The input should use getline() for the name at least because the user will be entering their name. There can very well be a space or special characters in a person's name. Whereas cin wouldn't meet those cases
- **What variables will you need? Why?**
    - A string definitely to store simple data like a person's name. An array of strings seemed like the obvious answer to store the data from the file. But as you came to realize you needed to interact with that data you realized that the vector class doesn't have as many limitations as the array of strings has. So that was an easy choice there. But mostly variables were needed to:
        - to store user input (menu options, search key, name)
        - store the parsed file in memory
        - Constants for file names
        - Constants for delimeters
- **What constants, if any, will you need? Why?**
    - Constants were needed because there was data being used inside the program that never was going to change and didn't need to be entered by the end user
        - Delimeters that were in the program that would be accessed by various functions but never changed
        - File names that were in the program that would be accessed by various functions but never changed
- **What functions do you use? How did you decided what operations go into each function?**
    - I used a function to do the following:
        - greet the user
        - Set the user's name
        - Get the users name
        - Create a menu for the text file
        - Create a menu for the algorithm to choose
        - Create a menu for the user to choose printout
        - Validate user input (overloaded to accept string and integers)
            - Overloaded because I didn't write any templates.
            - That said I was going to be accepting 2 types of data
                - Strings
                - Integers
        - Processing the algorithm input that came from the user
            - Based on their selection do something
        - Processing the text file input that came from the user
            - Based on their selection do something
        - A menu to allow the user to repeat the whole program
        - Getter methods for all of the attributes in each class
        - Setters methods for all of the attributes in each class
            - Except for constants, those clearly don't need setters

            - Except for attributes that would take user input
        - Write data from memory to a file
        - Write data to memory from a file
        - Tokenize data from the file
        - Print the first 50 tokenized items in the data set
        - Print the last 50 tokenized items in the data set
        - Search the data set for a key
        - Search functions (quick, merge, selection, insertion, bubble)
            - Function to sort half of the data set
            - Function to sort the whole data set
- **How do you decide that your program is complete?**
    - I decide that the program is complete by first meeting every requirement listed in the requirements document.
    - If every criteria has been satisfied, then I can move onto nice to have functionality if time allows. Those things being the following:
        - Is there any undesired or odd functionality that exists
            - Does it need fixing or to be addressed?
        - Was there anything I "cheap'd" out on in terms of bad coding practice but just went ahead for time management
            - Can I go back and fix it to make it more efficient (cheaper or faster)

Development Process:

When I was first reading through the requirements, I was already planning out how I wanted to break down the assignment. Immediately, I thought about how one of my coworkers constantly preaches getting a minimum viable product to ship out. Then from there, slowly adding in more features of nice to haves to mvp as you go through the product cycle. So I'd say that definitely impacted my mind set here with this lab. My overall plan for the assignment started off with getting an initial working program that met at least 1 requirement that I could ship to a "customer" (or submit as an assignment). What that looked like for me was:

- Create a text file with minimal data (two sentences that I just copy/pasted 5x)
- Read that file into memory
- Print the file's contents that are stored in memory out to the console
- Create a menu for the user to give a name
    - Store and print the name
- Let the user choose the hardcoded file
- Clean the data that was saved from memory

Then basically from there I began looking at each requirement from the requirements document that was given and added onto what I had from there. As comes with every development cycle I had to decide what were nice to haves and what was meeting the requirements. With this in mind I let a couple bugs slide here and there so I could move onto more challenging parts of the assignment. For example: in my code I validated the user's input from the menus with a conditional statement while the entry was less than 1 or greater than 6. But not all of the menus

I provided in the program had 6 options. Some had 5, 2, 3, or 1. So if they entered 2-5 they wouldn't get a validation but the program would end. That was something that I defined as not failing to meet the requirements but something I didn't need to address for delivering the product as there were more "core" or important things I needed to do, such as completing the quick sort. Stepping back a bit, I also thought to myself: should this code live in its own function? Do I want my main to be this long? Do I need to create this many objects? Private, protected, or public? Get or set, how? These all drove my decision making on whether a function member needed to be public, constant or if the function should be derived from a super class. Even as I was developing in the later stages I was considering breaking up a class to make it more portable, but was something I threw out the window because my project hadn't met all of the requirements and I was running low on time.

Product:

- Test cases: (this part wasn't hard for me seeing as I work as a QA Engineer…)
    - During user input
        - What data can you enter
            - Numbers
            - Characters allowed
            - Character count
            - Queries
        - Useful messaging when validation user input
        - When printing sorting times
            - Formatting/alignment
            - Reasonable header names
            - Context to where the user was during the program
            - Accurate data output/validation/calculation
        - When printing out menus
            - Again context of where the user was
            - User friendly messages
            - Good line spacing as to not clutter the console

Pitfalls:

The easy answer for pitfalls is always going to be a new programmer not having knowledge of best practices and *not reinventing the wheel.* I often find myself not having exposed myself to many libraries that would optimize my code or make a solution easy. An example of that in this assignment being vectors. Prior to this assignment they were something I shied away from because I didn't want to put the effort into expanding my c++ knowledge (lazy student alert). So as I tried to meet the requirements using an array of strings I began to realize I was making my problem solving harder than it needed to be. As I researched vectors I began to see the function members of the class and realized how much simpler it was. Another pitfall I had was putting off complexity and focusing on an end to end shippable product. I put of quick and merge sort for as long as I could because I *really* only needed to get 1 sort function in my sort class working and address the rest later. That said I actually didn't leave myself enough

time to get back to merge sort and fell short on meeting that requirement. Therefore in a real life scenario I would have failed to meet the customers requirements by the deadline. So you could say time management was also a pitfall. Which would then fall back on planning & organization.

Possible Improvements:

- What could you have done differently to improve your specific completion of the assignment?
    - Better time management 100% would have improved my completion of the assignment.
        - I'm personally a fan of visual aids. At work we use physical scrum boards to track our progress (agile development). I guess I could adapt a practice similar to that for me. But my sprint or cycle would be to meet the time given for the assignment. Then break that up based on the assignment. But I actually wrote out parts of the program I wanted to complete with days I wanted them done on a whiteboard on my desk. So I guess I sorta did that.
- What could have been done differently to avoid some of the possible pitfalls in general, especially those pitfalls that seem to be common through your completion of different assignments?
    - I think my answer for the previous bullet meets this questions too.
    - Additionally though, more reviewing of concepts outside of a given assignment. I actually did review pointers/file i/o the weekend before you assigned this lab. That helped me out tremendously with pass by reference/reading the file.
- What could be improved about this assignment overall, for you to better demonstrate a successful and mindful master of the programming concepts covered so far?
    - More emphasis on the concepts that were covered during this class. I feel like sorting and searching were like 30% of the assignment. Searching and sorting were actually two of the things I found to be the easiest. I think I spent most of my time on cleaning the data and breaking apart my functions/classes appropriately.