Jonathon Harris

CS202

11/27/20

Media Manager - Efficiency Writeup

The main purpose of project 4 was to create media managing software that can store different types of media in a playlist. The playlist is an array of linear linked lists in which every index represents a playlist with the same topic. My playlist class was responsible for adding, removing, displaying and finding different types of media. My class, Media, was my abstract base class that holds the title, topic, description, comments and view count which gets incremented every time the user displays that media. My video, video with quiz, and documents were inherited from media. For my function overloading I made constructors that takes different arguments and create objects based on what arguments are passed in. Due to my layout and class responsibilities, I didn’t have a lot of functions that were called with dynamic binding. My display was the only function shared between children. I could prove that the function overloading was working because when I passed in a Media object into the function a new object was getting created versus when I passed in each parameter explicitly. My main program was a clean interactive menu giving the general options that when selected would expand to more options.

The first menu was add, remove, display and import. But when you clicked on display it would give you options for display one, display all topics, and display all media. I’ve changed my ways of having a gigantic menu with all the options in the beginning, I find it can be overwhelming for the client.

Overall, I really enjoyed java, there is way less responsibility placed on the programmer. Deleting for instance was a little strange for me because you’re essentially just reconnecting the list circumventing the node to be removed. I felt it was really nice to not have to worry about deallocating memory. I believe the efficiency overall was really good. My add function added to the start of the list which resulted in O(1) time complexity. My import media from a text file also used the add function so it also had a O(1) time complexity. The remove would have to go through the whole list so this was O(n) time complexity. The display all also resulted in a O(n) time complexity.

Working with an integrated development environment was life changing. The main difference for me was when you made a mistake you could immediately see that it was a problem. It would make suggestions about variables being vague. Also, it would insert try and catch blocks based on the checks of my program. For example, when importing data from a text file it set up a catch block for if the media file was not found. It also setup a catch block for my menu in the case that the user doesn’t input a valid option. I ran into issues regarding the removal of nodes. My remove function would work only if there were multiple media items in my list but when it came to removing one item at the head of my list it wouldn’t remove it. After hours of troubleshooting, I figured out that I needed an additional check for if the data was at the head of the list if it was then set head’s next pointer to null. I found this to be strange since my initial check was seeing if the nodes title matched then set the next pointer. Which is essentially the same thing as this check. I’m grateful we can use an IDE now it makes writing in a new language much easier