Lab 4 – Self-Organizing List – Approach Document

Assignment Objective

This assignment’s objective is to implement Count, Move to Front, and Transpose Heuristics using a linked list. The implemented will be tested first using characters and second with strings read in from a file.

Assignment Requirements

This program should:

* Run without errors
* Correctly implement the Count, Move to Front, and Transpose Heuristics
* Test implementation using both characters and strings read in from a file
* Size of list, number of compares, and each record along with its frequency should be properly printed out to the command line
* Include a healthy amount of comments (but not too many)
* Include an approach document

Approach

* Review all instruction files (word document and powerpoint slides) to get a better understanding of the requirements of this lab. Read through the algorithm explanation of each organizing heuristic to get a better understanding of how they work
* Start implementing each Self-Organizing List as their own separate implementations
* Test the add function, then move to find, then create and implement a reorder function for each self-organizing heuristic
* Write the approach doc
* Submit code to Canvas

Build Log

Day 1-3 (4-3-22 – 4-5-22) – Today I set up my Visual Studio project by copying the provided files into it. I then began implementing the SelfOrderedListADT class. At first I tried to implement each self-organizing heuristic in this class using an array of LList objects, but I realized soon that this is not a very effective method, and makes things very confusing. I thought it would make sense to use an array of 3 LList objects and use each separate one for the 3 organizing heuristics, and I got pretty close, but I eventually decided that this was not the best approach. It forced me to use unnecessary for loops, square brackets, and I just couldn’t get it to output everything correctly.

Day 4-5 (4-6-22 – 4-7-22) – After discussing with a couple classmates, I decided to scrap what I had and go for separate implementation files for each organizing heuristic. Luckily, my add, find, and print functions needed very little work to get working properly. I pretty much just had to remove all of the unnecessary for loops and square brackets. Now that these functions seemed to work, it was time to begin implementing the reorder functions. These were pretty simple after reading through the instructions that explained the algorithm for each. Once I got these functions done, I fixed a few bugs that were messing up the output that originated from the find function. Once this was complete and I verified that the outputs were correct, I started adding functionality for reading in from a file. This was very simple as all I had to do was create three more objects of the various sort heuristic implementations, loop through each word of the file and input them into the list for each heuristic using the find function. Once I finished the main bulk of the assignment, I began adding comments, working on the approach document, and then submitted the project.