sorter.h

I first created a struct handling each record. Then I created another struct put each record in to a node with a reference to the following node. Then a linked list struct handling all the nodes(records) in a linked list fashion. The linked list struct also handles the length of the list, the column to sort, and the sorting type(0 is a string, 1 is int, 2 is a double).

sorter.c

Read in the csv file via stdin. Dynamically allocate memory for the linked list. Dynamically allocating memory for each node. Then read each line with skipping the first line and putting each record in at the end of the linked list. Once that is done, free the buffer and the data node. Next, I define the list types in linked list for it can be easier to use merge sort. Then I use merge sort to sort the linked list based on the given field inputted in the terminal. I put the new records in a new csv file to see if the csv file was sorted correctly before using stdout. Then I used stdout to print out the new sorted records via the terminal

linked_list.c

I created this c file as helper functions for the linked list struct. This allow the linked list to add nodes at the tail of the linked list. Define list types of the variables in the linked list. Also free the linked list out of memory.

mergesort.c

This mergesort.c is your everyday mergesort. Just recursively do mergesort till all the nodes in the linked list are sorted based on the column inputted via the terminal.

Challenges I had was using the malloc function. Coming from using Java a lot, using pointers and dynamically allocating memory was kind hard for me. I would rather deal with a null pointer exception in java than a memory corruption in C. I just feel like its easier to debug a null pointer exception than memory corruption in C. Maybe overtime if I practice a lot with programming in C, it will become second nature to me to fix memory corruption errors in C. Sometimes it can be nuisance when you allocate memory then get a stack trace of errors in the terminal. It took me a while just to fix that problem. Other challenges I had was to deciding to use an array to store the records or use a linked list struct. In my opinion, I felt like using a linked list data structure was easier because I prefer using linked list than an array.