Stephen Scott, Michael Tang

Our fileSort.c implements a linked list by using a node struct and a linklist struct. These structs serve as the foundation for the linked list data structure. We have various functions which help with constructing and maintaining the linked list. When the program is run, it checks if 3 arguments are present, and if the given file is accessible. If not, then an error is displayed. Then, it opens the file, reading 1 byte at a time. It also checks if a byte is numeric or a char, which will be used when deciding what comparator to use. While reading, it inserts the byte into the linked list. After all bytes are read, then it closes the file.

Next, the program checks the command line flag to decide which sort to use. It sorts the linked list and prints the final result, line by line.

void linklist_insert_head(linklist *p_linklist, int index, char *str) : Takes a pointer to a linked list, an index, and a char*. It inserts the char* into the linked list at the given index.

Void linklist_remove_head(linklist *p_linklist): Takes a pointer to a linked list and removes the head.

Int linklist_get_node(linklist *p_linklist, int index, char* pstr): Takes a pointer to a linked list, an index, and a char*. It copies that node's string to the char*.

Int linklist_modify_node(linklist *p_linklist, int index, char * pstr): Is the same as linklist_get_node except it copies the char* into the node.

Static int print_callback(node *p_node, void *p_data): Prints the next node.

Void linklist_print(linklist *p_linklist): Prints the given linked list by implementing print_callback.

Void linklist_deinit(linklist *p_linklist): Removes all heads from the linked list.

Int numericSort(void *str1, void*str2): Is a comparator that takes the two numeric tokens and uses atoi() to compare them.

Int charSort(void* strptr1, void* strptr2): Is a comparator that takes two char tokens and compares them.

Int insertionSort(void* toSort, int(*comparator)(void*,void*)): Takes the list that needs to be sorted and a comparator. Implements the insertion sort algorithm on the linked list by using a given comparator (numeric or char).

Int quickSort(void* toSort, int(*comparator)(void*,void*)): Implements quick sort algorithm on a linked list and a comparator.

Int main(int argc, char* argv[]): Takes command line arguments, reads a given file, stores the contents to a linked list, sorts the linked list based on the given flag, and outputs the sorted list.