UCP Assignment Report

Contents

These are the files for the program and they each have descriptions of their respective functions

Function Descriptions

tvGuide.c:

Main function

The tvGuide.c is the file that contains the main function. The main function is the entry point of the program. It takes two arguments from the command line, an input file name and output file name and passes them into the readFile method and writeFile method respectively. If the number of command line arguments does not equal three, then error message is printed to the screen and the program is terminated. User input for the day and sorting method is obtained from the interface.c method and is used for the qsort and filtering of data respectively. An array of pointers, to TvGuide structs, is exported from the readFile method and is transferred to tvGuideArray which is also an array of pointers to tvGuide structs. The time and name of the show is extracted from each element using the -> operator and then printed to screen aswell as written out to output file.

fileIO.c

readFile Function

The readFile method imports a string as a file name, which is argv[1] from main. When the file is opened, it checked to see if it is empty and if it is, an error message will be outputted, otherwise the file be read from. Each entry in the file, to read, has two lines, the name of the TV show and the date and time the show is supposed to air. Memory is allocated to the struct itself and the strings within the struct with malloc. Each element that is read in is assigned to a variable in the struct. The name of the struct is TvGuide and it holds each entry of tv shows. Each entry is then inserted into a doubly ended double linked list using the insertFirst method by passing in a list to add to and the actual value to add; the tvGuide entry. This whole sequence of events will continue until the end of the file has been reached. The elements in the linked list are then extracted and placed into a dynamically allocated array of pointers to TvGuide structs. All allocated memory has been freed after they finshed with except for tvGuideArray. If tvGuideArray was freed then there won't be anything to export back the main function, therefore tvGuideArray is freed there instead. After method is done, the file is closed.

writeFile Function

The writeFile function imports an array of pointers to TvGuide and a string that is the output file which is argv[2]. Each entry in the array is written to the output file in the format: 'hour:minute – nameOfTvShow'. Each attribute of an entry is extracted from a struct within the array and placed into the file with this format. After all elements of the array have been read in, the outputfile is closed.

Interface.c

userInputDay Function

The userInputDay function imports nothing and exports a string which is a day of the week. The function asks the user to input a string of any day of the week. Input validation ensures that the user does not input anything other than a day of the week. If the user enters invalid input, an error message will print to the screen and he/she will be asked to re-enter input. This process will continue to loop until the user enters a valid day of the week. The strcmp function is used to compare two strings for equality, if two strings are equals, strcmp returns 0. The day the user entered is the exported.

userInputSort Function

the userInput Function imports nothing and exports a string which is either time or name; the value you want the entries to be sorted by. The function asks the user to enter one of the two strings. Input validation ensures that the user does not input anything other than a valid time or name by printing an error message and asking the user to re-enter their option. This process will continue to loop until a valid option has been entered. Just like with userInputDay, the strcmp function is used to compare two strings for equality.

Linkedlist.h & Linkedlist.c

LinkedListNode Struct

The LinkedListNode struct represents a single node in the doubly ended double linked list. Contains a void pointer as the value making it a generic node. The struct also contains pointers to the next node and the previous node which makes it a double linked list, therefore allowing us to traverse the list both ways.

LinkedList Struct

This struct represents a linked list. It contains LinkedListNodes called head and tail which points to the front of a linked list and back of a linked list respectively making it a double ended linked list

createLinkedList Function

This function creates an empty linked list. Memory is allocated to this struct that is the sizeof the LinkedList struct. Head and tail point to NULL since the list is empty. This linkedlist is then exported.

insertFirst Function

This function creates a new LinkedListNode and inserts a value at the start of the list. The data attribute in the linkedlist is assigned the value of the data that has been imported in. The linkedListNode next points to the same place that head points to because it is at the start of the list. Prev points to NULL because, since it is the beginning of the list, there is nothing for prev to point to. Since values are being inserted from the front, the values will be displayed in reverse order.

insertLastFunction

This function creates a new LinkedListNode and inserts a value into the end of the list. The data attribute in the linkedlist is assigned the value of the data that has been imported in. The linkedListNode prev points to the same place that tail points to because it is at the end of the list. Since values are being inserted from the back, the values will appear in their normal order unlike with insertFirst

removeFirst Function

This function deletes the node at the front of the linked list. The function imports a linked list and gives it a temporary node. Temp points to the same address that head points. Head then points to the node after temp. The data within temp has been freed and after that the entire node temp is freed. The node is now deleted.

retrieveElement Function

This function imports a linkedlist and an integer representing the index of the value we want to retrieve. The node current initially points to where head points to. Current then traverses the list using a while loop and while current is not NULL, it will keep traversing until it reaches the index that was imported. Once it reaches the node at index the value stored in that node is extracted and then exported.

freeLinkedList Function

This function frees the memory allocated to a linked list after it is no longer being used. The function traverses through the list and and frees the data first and then frees the entire node. It will do this for every node being traversed which is all of them.

data.h

TvGuide struct

This struct defines a single entry for a TV show. It contains a string representing the name of the TV show, a string representing the day the show is to air, an integer for the hour segment of the time and another integer for the minute segment of the time. The time will be in the format: hh:mm.

Comparison.c

As ran out of time before I could implement qsort, I just wrote a rough design of what I thought the comparison functions should be like in order to demonstrate my thought process.

timeComparator Function

This function imports two TvGuide* entries and compares them in terms of time. The hour and minute are added together for each entry and then compared to each other.

If time1 – time 2 >0 then time1 is bigger than time2. If time1-time2 < 0 then time 2 is bigger than time1. If time1=time2 then they are equals.

nameComparator Function

This function imports two TvGuide* entries and compares them in terms of their TV show names. The names are extracted from the struct and used by strcmp.

If the value is greater than 0 then name1 is greater than name2. If value is less then 0 then name1 is less the name2. If value = 0 then the names are equal

Filtering Entries From Array

The array contains entries from over the whole week. The optionDay variable gets a day of the week from the userInputDay() function in interface.c The array tvGuideArray is then traversed within a for loop with size(tvGuideArray) being the final index of the array. The day is then extracted from the struct within element at index ii. The strcmp() function is used to test equality between the day extracted from tvGuideArray and the day that has been inputted by the user. If strcmp returns 0 then they are equal in which case, the contents at index ii must be printed to the screen and written to file.

The printf statement will print to screen by extracting the hour, minute and the name of the tvShow at index ii where the day from tvGuideArray is equal to the day inputted by user. The same is to be done for writing out to file. If the day from tvGuideArray is not the same as the day from the user input then the contents at index ii will not be printed and the next index will be checked, this process will repeat until the end of the array is reached.

Demonstration of Program

Since I ran out of time I was not able to fix all the errors, therefore I am not able to show any samples of my input and output apart from the command line and user input.

Command line arguments: ./program input.txt output.txt

User Input as seen on terminal:

Enter a day → Friday