ECE 3331, Dr. Hebert, Fall 2023, Programming assignment 08 due Saturday 10/21 at 11:59 pm

Download and refer to the handout regarding WAV files.

The file ts9_mono.wav is a short WAV file that has a single channel (mono), instead of two channels (stereo).
The audio data are short int values (16-bit signed values using 2's comp).
The file amerika3.wav is a short WAV file that has two channels (stereo).
Here, the audio data are 24-bit signed int values (24-bit signed values using 2's comp).

Your program will open this WAVE audio file (ts9_mono.wav), read the values in the WAV header and print them to the screen, and finally overwrite 3 values in the header, such that the file will then play at half speed.

Step 1) Create a 200 element 1-D char array in your main( ) function to hold a filename entered by the user. Call a subroutine that takes as its argument a pointer to the char array and returns an integer. The subroutine will prompt the user for a path/filename for an existing mono WAV file (you can use ts9_mono.wav to debug your program). The subroutine will read this path/filename from the keyboard as a string, and store it in the 1-D 200 filename char array. When you are done with Step 1, test your code by printing the path/filename to the display.

Step 2) In the subroutine that you wrote in Step 1 above, add some additional statements to check that the file extension (the 3-4 letters past the **last** period in the path/filename), testing whether the file extension entered by the user is WAV or wav. A simple way to do this is to use strstr( ) to search the filename string for either of the strings ".wav" and ".WAV" . If the file extension is WAV or wav, have the subroutine return the integer value 1, indicating the path/filename is valid. If the file extension is not wav or WAV, notify the user of his/her error, and have the subroutine return the integer value 0, indicating that the path/filename is not valid.

Step 3) In the main( ) function, if the subroutine returns the integer 0, have the main( ) function re-prompt the user for a path/filename and call the subroutine in Step 1.) again. Here, you can see that you should be doing Steps 1,2 (and Step 3) inside a loop in the main( ) function that exits when the user enters a valid path/filename for a WAV file.

Step 4) In your main( ) function, when the user has entered a WAV path/filename, open the file as "r+b" (read-write-binary). This ("r+b") doesn't create a new file if the file doesn't already exist. Also, "r+b" doesn't destroy the contents of the file upon opening if the file does exist. Finally, "r+b" will allow you to write to the fille, as well as read from the file. Check the file pointer to see if it is NULL. If so, notify the user, and re-prompt (i.e. call the subroutine) starting the process over. Here, you can see that you should be doing Steps 1,2, 3, 4 inside a loop in the main( ) function that exits when the conditions for inputting a valid filename from the keyboard are met.

Step 5) In the main( ) function, if the WAV file is successfully opened, have your program read all of the header values and print them to the display. To do this, declare appropriate variables for each of the values in the header. Use a 4-character array for ChunkID, for example
char chunkid[4];
You might use a loop to read the 4-characters from the file, store them in chunkid[ ], and print them to the display. Use fread( ).
Also, for example,
unsigned int chunksize; // use fread( ) to read-in the value from the WAV file and store it in variable chunksize.
….
unsigned short int audioformat; // (2-byte unsigned short int)
unsigned short in numchannels; // (2-byte unsigned short int)
unsigned int samplerate; // (4-byte unsigned int)
etc..

Step 6.) Divide your variables sampleRate, byteRate, and blockAlign by 2. Use fseek( ) and fwrite( ) to overwrite these existing values in the file with your variables that were divided by 2.

Step 7.) Close the file.