TCSS 333 HW6 Sound Files
Due: by midnight Wed. March 2 (or with 10% late penalty, Thurs. March 3)

Go to http://www.mega-nerd.com/libsndfile/
follow the download link and download libsndfile-1.0.26.tar.gz

To work with the library, you should use either the shell in cssgate or the similar command window in cygwin. For this assignment, you should probably not use Eclipse.

Once you have downloaded the .gz file mentioned above, uncompress it with two linux commands:

```
$ gunzip libsndfile-1.0.26.tar.gz
$ tar xf libsndfile-1.0.26.tar
```

After these two commands, you should now have a folder called libsndfile-1.0.26
Using cd, move into that folder and type

```
$ ./configure
```

There should be a long list of messages such as "Checking ...".
On both cssgate and (at least for me) on cygwin, none of the messages is an important error.
Next you compile the sound library by typing

```
$ make
```

There will be many messages and again on cssgate, none of them will be serious errors.
You can now try out the sample programs. Move into the directory called examples and try running the executable make_sine:

```
$ ./make_sine
```

The result should be a new file called sine.wav. If you find that file in the Mobaxterm file display and double click it, Mobaxterm will transfer it to your local machine and open an audio player so you can hear it. (Find the volume control and turn it down before you try this!) My own experience has been that this also works on cygwin if for some reason you don't want to use Mobaxterm until final testing.

You should also go to the directory "programs" and try out the one called sndfile-concat.c.
It makes a bigger .wav file out of 2 or more smaller ones.

After studying make_sine.c and sndfile-concat.c, you should be ready to try writing a program yourself.
Write a program called reverse.c which reads one .wav file and makes up a new one with the recording reversed. In other words, the samples of the new file will be the original ones played backwards. Your program should take two command line arguments, first the name of the original file and second the name of the new file. For example:

```
$ ./reverse myfile.wav backward.wav
```

Your program must use the sndfile library to open, read, and write soundfiles. Your program must be a single .c file. For grading purposes, I will place your program in the folder "examples" of my copy of the libsndfile software and I will compile it with this command:

```
$gcc reverse.c ../src/*.o ../src/ALAC/*.o ../src/G72x/*.o
../src/GSM610/*.o -I../src -lm -o reverse
```

That's all one line! A sample .wav file is posted on Canvas for you to work with. If you reverse a file and then reverse the result, you should get back to the original recording. I suggest you try to read and

write sound file contents in a single call instead of many small ones. You'll run into fewer problems.

The program you are trying to write is only 100 lines or so, but using the soundfile library may be challenging. You should be prepared to print out partial results to see what is happening. When you read or write data to the files, the functions return important results. Be ready to print them out as needed.

Remember that a sound file is made of frames which are numbers representing the sound wave at some moment in time. Our recordings have all been recorded at the rate of 44,100 frames per second. Also the recordings we've encountered have all had two channels, left and right. The sounds files begin with a small header which the sound library will handle for you. Then the data is simply a long list of numerical values:

frame 1: <left channel number> <right channel number>
frame 2: <left channel number> <right channel number>
frame 3: <left channel number> <right channel number>
frame 4: <left channel number> <right channel number>

You will notice the data type, SF_INFO. It is defined in sndfile.h in the src directory. It may help you to be able to print out all its fields to see what information is available inside it. Above all, the most useful thing you can do is study examples/make_sine.c and programs/sndfile-concat.c carefully and imitate what you see there. As a first step to reversing a .wav file, try making a simple copy! You may find some of the documentation that comes with the library to be useful.

Here's your strategy:

- read in all frames from the original file and put them into an array of the right size
- change the data around
- write all data in the array into the new file

libsndfile is a perfectly good library, but you may be surprised by how it works.