Problem Set 9: Reverb

Please send back via NYU Classes

- A zip archive named as
  `PS09_<your name as FirstLast>.zip`

containing the C code files that implements all aspects of all problems.

Total points: 100

The instructor has supplied the following files:

```
wav_reverb.c
wav_reverb.h
process_audio.c
process_audio.h
convolve.c
convolve.h
```

You have to add code to the following files to complete the assignment:

```
wav_reverb.c
process_audio.c
```

Part 1: Parse command line and open all files

`wav_reverb.c`

The program should have the following command line usage:

```
wav_reverb ifile.wav reverb_file.wav [ofile.wav]
```

where:

```
ifile.wav is input audio file
reverb_file.wav is reverberation impulse response file
ofile.wav is output audio file
```

All files must have the same sampling rate and same number of channels and be no more than 2 channels.

Parse command line

Parse the command line. If parsing fails, print an error diagnostic and exit. If parsing succeeds, then print command line values after parsing.

Open WAV files

Use the libsndfile library to open WAV audio files and read the WAV header of each input file. Use error checking and error reporting in all operations.

Check that all files have the same sampling rate and same number of channels. If not, then print an error and exit.

Part 2: Allocate buffers and read input audio signals

Allocate sufficient storage to ibuf, rbuf and obuf (input buffer, reverb buffer and output buffer) using malloc() to read the entirety of each of input WAV files. In addition, allocate the same storage for output WAV file. The number of sample frames in obuf is the number in ibuf plus the number in rbuf.

Read the ifile.wav and rfile.wav audio data. In every case use error checking and error reporting. The data in the file will be interleaved by channel. Add code to de-interleave the sample frame data into separate channel buffers.

Part 3: Do reverberation and write result

Modify `process_audio.c`
Process the input and reverb signals to create the output signal. The function should operate on an arbitrary number of channels (up to MAX_CHN). Create three "versions," selected via a #define MODE N preprocessor directive:

#define MODE 1
Just copy the input file values to the output file, and copy an additional rframes-1 zeros. This can be used to test the fread() and fwrite() code in your main program.

#define MODE 2
Convolve the input signal with the reverberation impulse response. The convolution result has iframes+rframes-1 samples.

Compute the RMS value of the input and output signals for all samples and all channels.

Normalize the output signal such that it has the same RMS value as the input signal.

#define MODE 3
Use the supplied convolve() function to perform the convolution via FFTs.