Assignment 5: Sound Effects

Due Oct 26 by 11:59pm **Points** 100 **Submitting** an external tool
Available Oct 5 at 12am - Oct 27 at 11:59pm

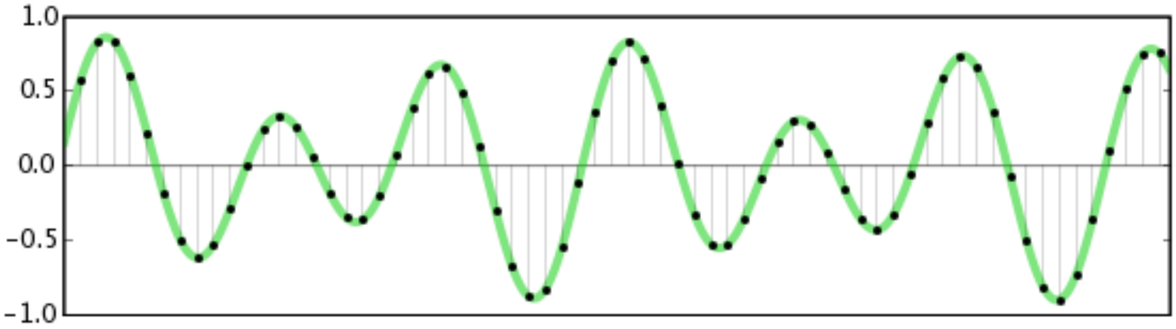
This assignment was locked Oct 27 at 11:59pm.

Assignment Goals

For this assignment, you will practice manipulating values in lists to make sound effects.

Sound Background

As described in class, sound is really a continuous wave of pressure changes. This continuous, or analog, wave cannot be exactly captured by the computer. Instead, the current value of the wave captured at a fast rate, and all these values stored in a list. The computer can play back these samples at a fast rate to approximate the original wave signal. The image below shows a continuous signal and the samples that approximate it. Time is represented going to the right and the power of the signal is its vertical distance away from 0.



In our system, we only have access to the samples, as the sound has already been digitized. The samples are stored as integers, typically ranging from around -32,000 to 32,000 in value. So the above sound might be represented by a list like [15000, 20000, 20000, 16000, 7000, -4000] and so on. Each value is the height of a dot on the sound wave (albeit on a plot that would have different ranges on the vertical axis).

Changes to the samples list values changes the shape of the audio signal when it is played back. When you make some effect on the audio, you are manipulating the height and spacing of the samples stored in the list.

The signal has two easy to understand characteristics. The faster the signal goes up and down, the higher pitched the sound will be. The higher (or lower) the signal, the louder it will be. When you hear changes to the audio, try to think about what is changing to the samples, and from those, the signal.

Assignment Activities

Download the **A5.zip** ⤓ **(https://utah.instructure.com/courses/796373/files/134138658/download?download_frd=1)** file and uncompress it. Open this project in PyCharm.

You can run the SoundApp.py file to open up a small program. This program loads a small audio clip from a well-known movie. The sample values are displayed in the center window. This view can be scrolled left and right to see the entire waveform.

The lower black waveform is the original sound. The upper red waveform shows any modified wave your program will compute. These are drawn using turtle - so it can be a little slow!

The Quit button closes the window, as does the normal window-closing x in the window corner. Make sure you keep closing the window as you test your program, else you will end up will dozens in your taskbar.

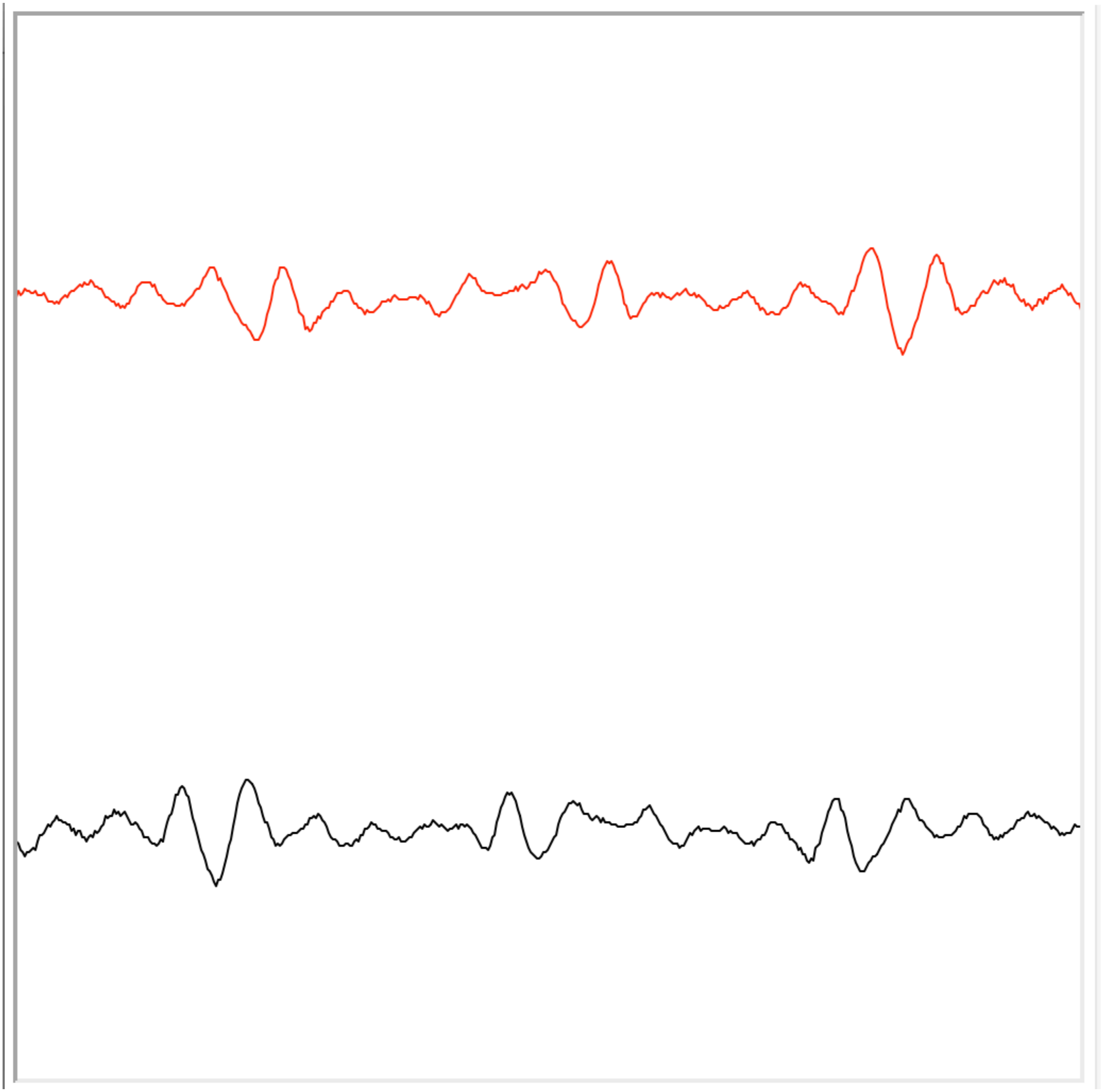
The Play button plays the original audio. The Chipmunk button is just for fun - it plays the audio back at a faster rate, which also raises the pitch. The rest of the buttons do things you need to add code for as described in the SoundTools.py.

Your task is to fill in unfinished code in the SoundTools. In general, most of the functions take in a list with the original samples and each function returns a new list with modified values.

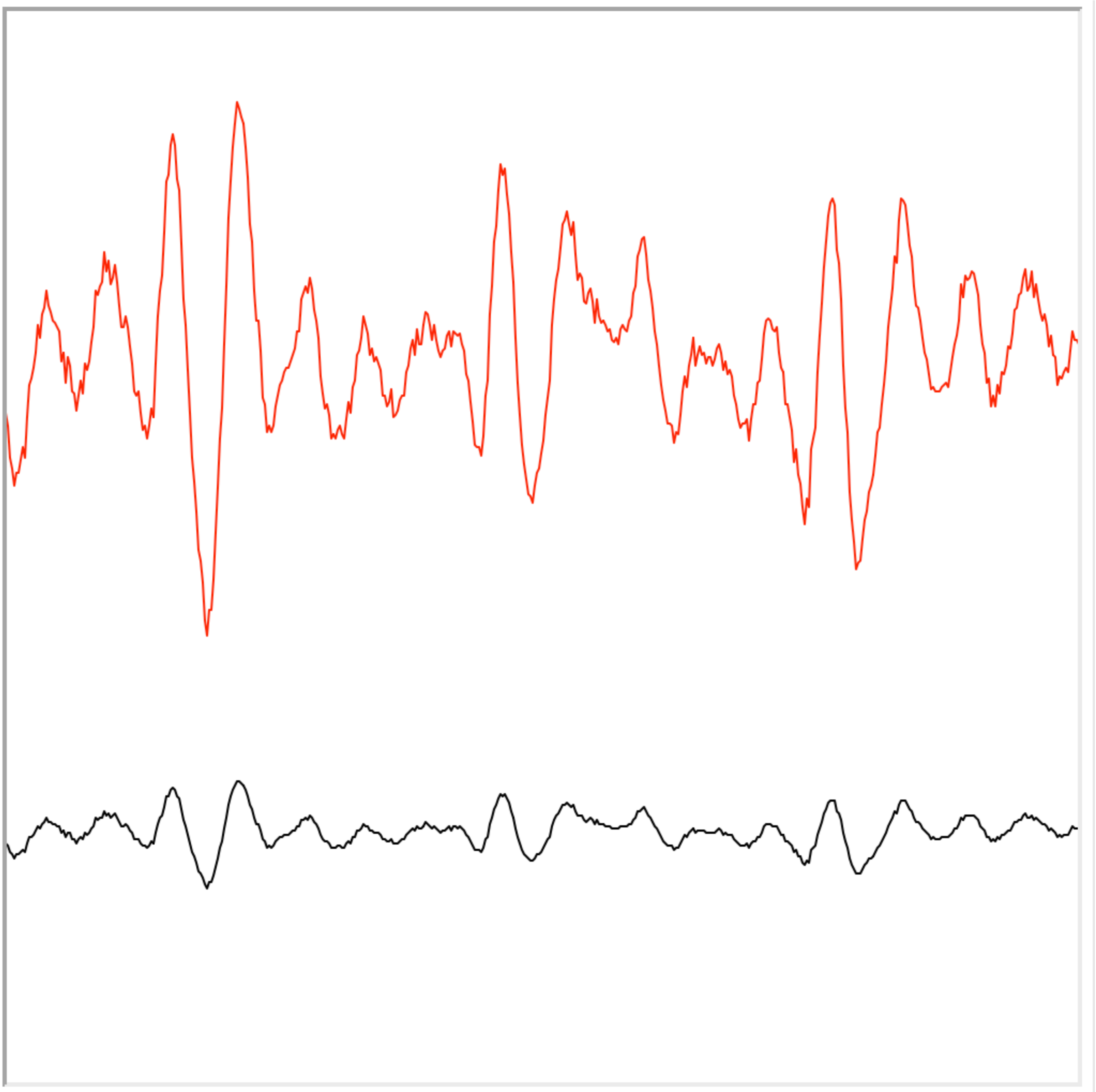
Even though these functions work on sound, you can think of them as just changing lists of numbers. There are small examples given in each comment that show the expected behavior. You can run code that checks your functions with small tests by adding code at the bottom of the SoundTools.py file and running that file instead of SoundApp. This is not required, but will likely be a useful exercise as you write and test your code.

You can also test your code by running the sound app. When you press a button for an effect, it should change the audio playback and also display a different waveform. I show example results for the 6 different problems below.

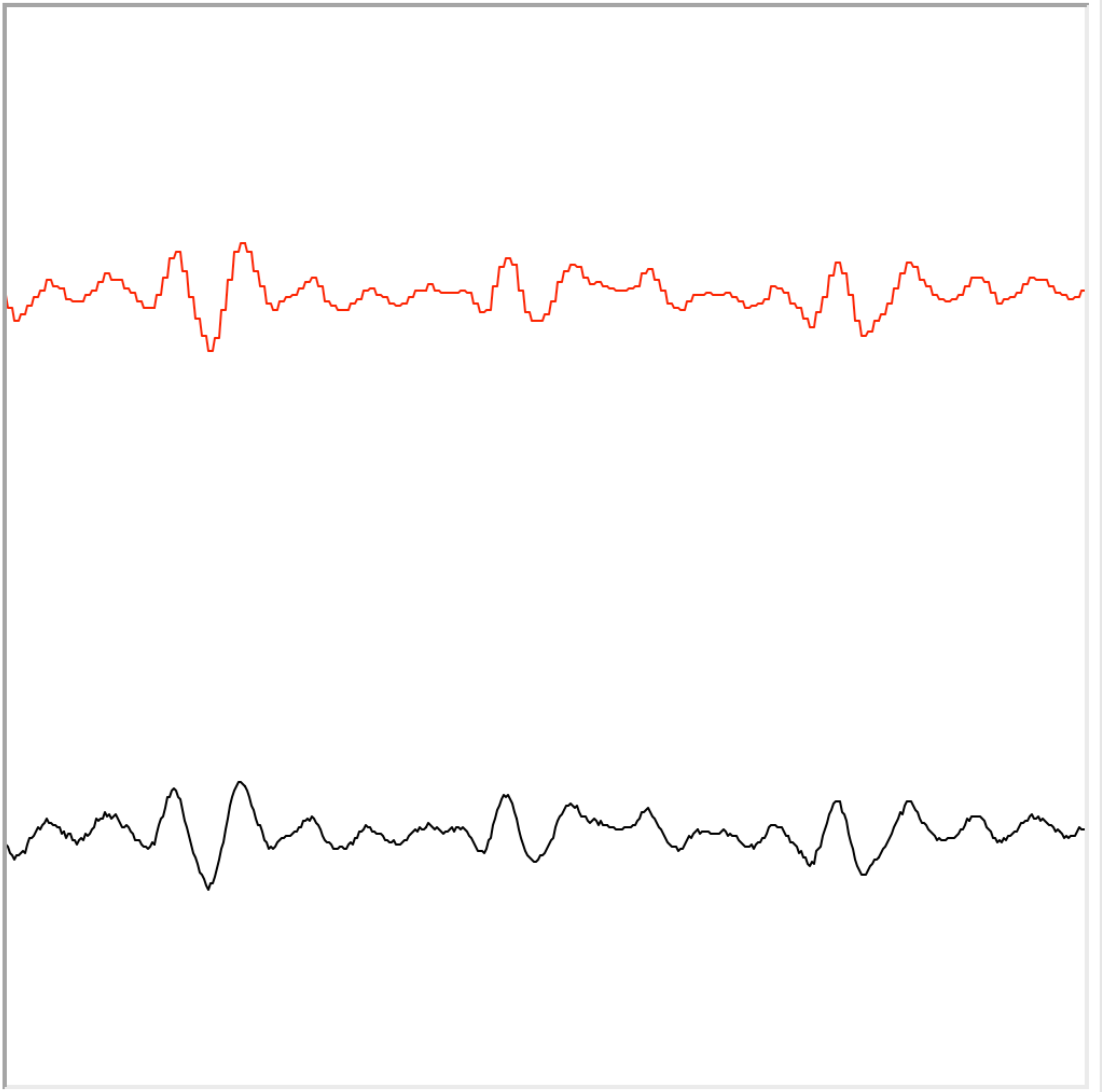
1) Reversed



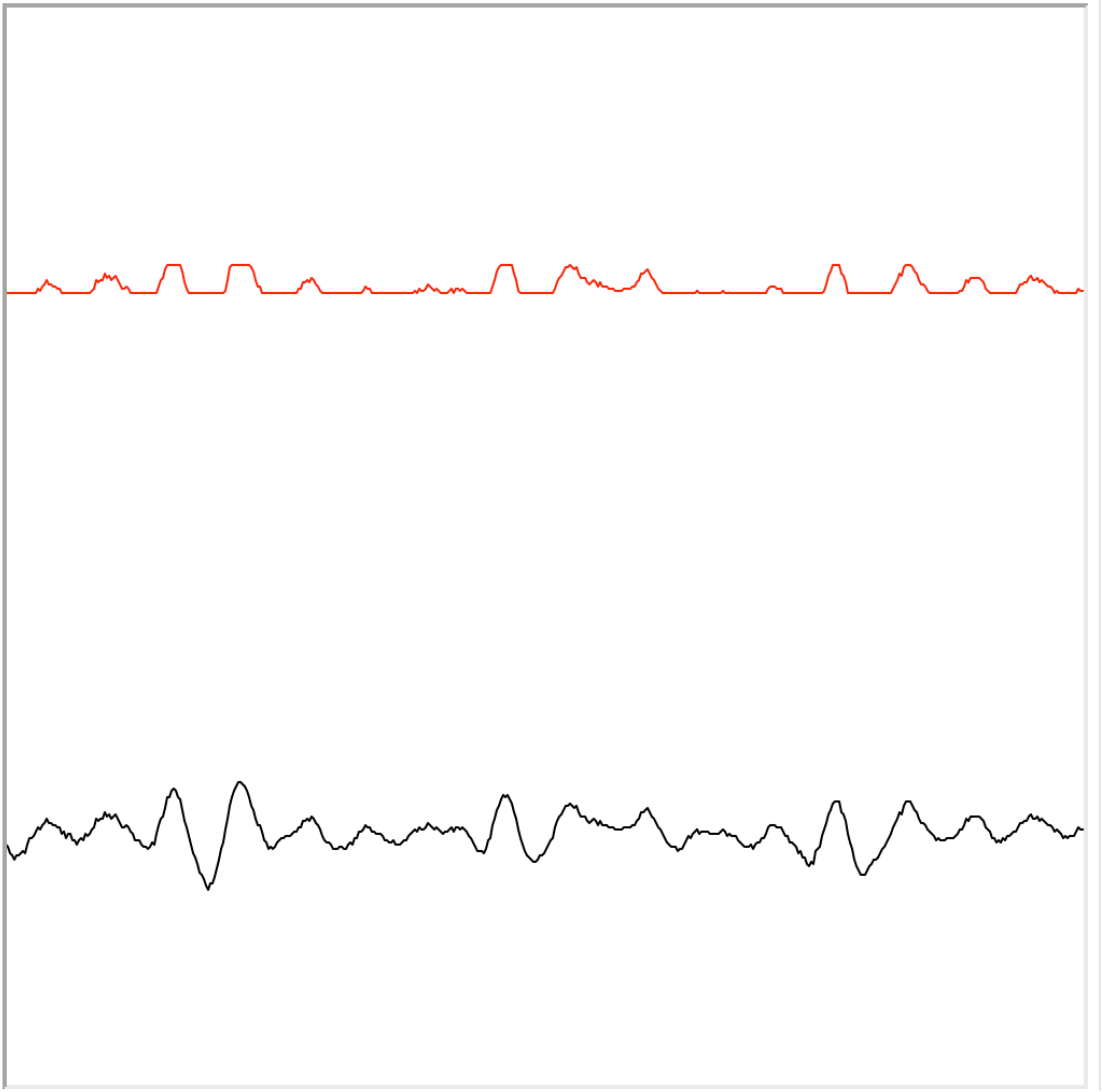
2) Louder



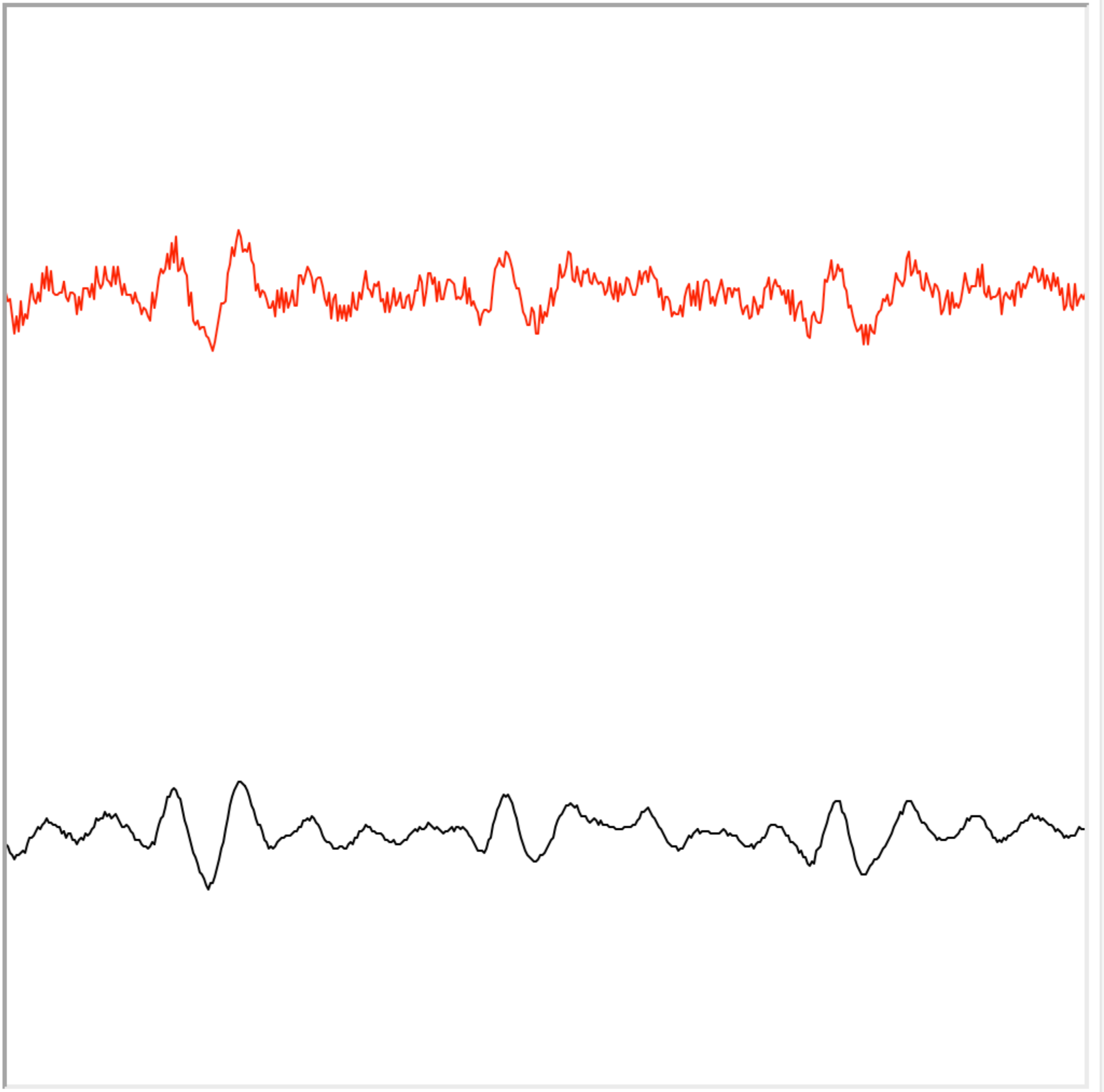
3) Reduce



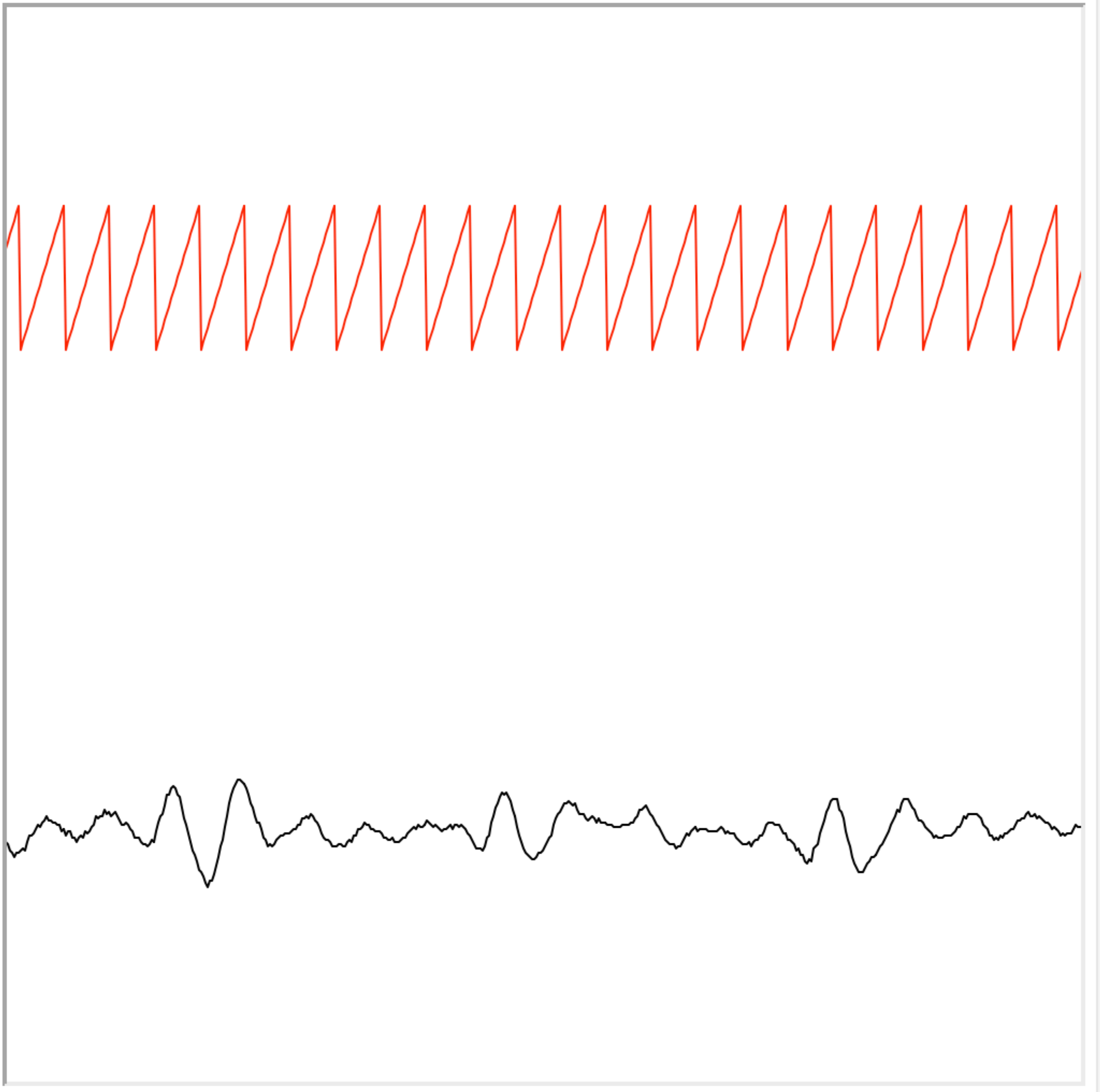
4) Clip



5) Add Noise



6) Make Tone



a51.PNG (https://utah.instructure.com/courses/796373/files/134138648/download?download_frd=1)

a52.PNG (https://utah.instructure.com/courses/796373/files/134138656/download?download_frd=1)

Submitting

You should only change the SoundTools.py file. Submit only that file.

A6 Sound Rubric

Criteria	Ratings	Pts
General: The functions work on arbitrary-length input lists rather than a fixed length		20 pts
General: The functions return new, modified lists. The functions should not change the original list or just print results.		20 pts
General: The functions use the input parameters to control volume, skipping, etc. rather than just doing a constant change.		20 pts
Reversed: The reversed function reverses the list using a loop.		5 pts
Louder: The louder function multiplies each sample by the scaling parameter.		5 pts
Reduced: The reduced function skips samples in the original list as specified.		5 pts
Clipped: The clipped function clips both negative and positive numbers to the clip value.		5 pts
Noisy: The noisy function adds appropriate noise to the original values.		5 pts
Tone: The tone function makes a sawtooth wave of correct shape and repetitions.		5 pts
The code shows good effort, style, and formatting.		10 pts
		Total Points: 100