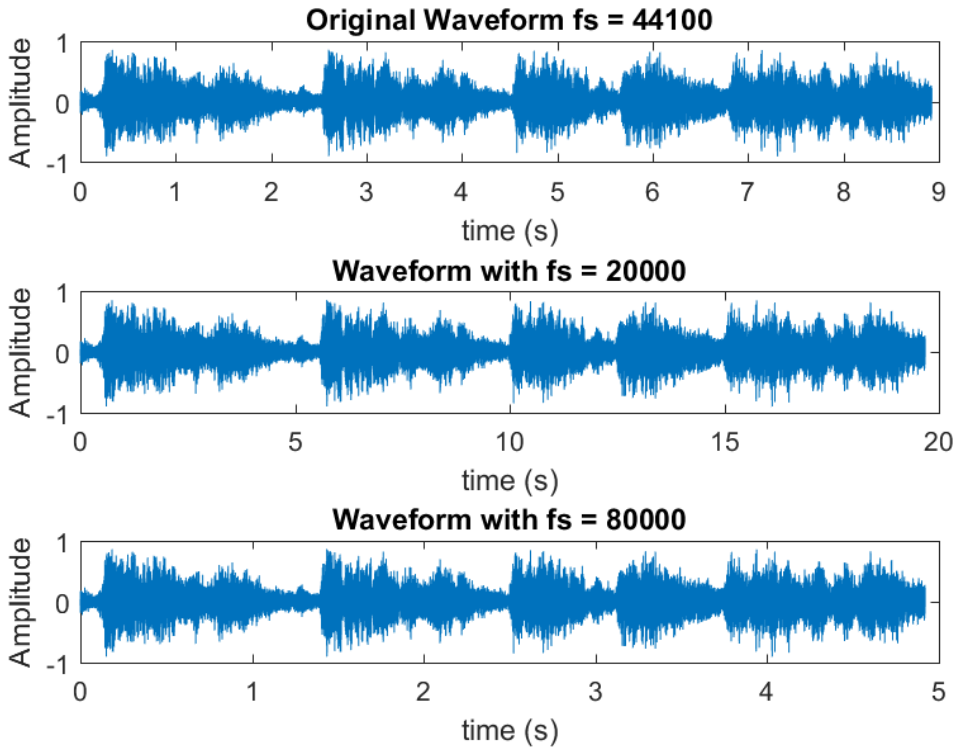
電機三 B05502145 林禹丞

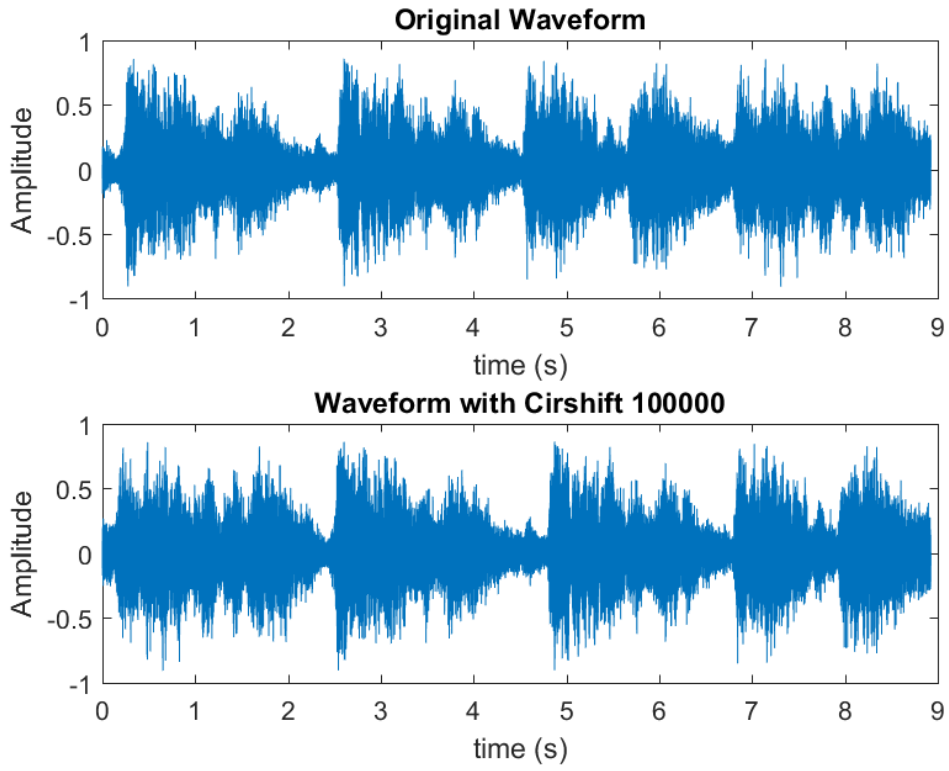
Report

1. Manipulating audio files:
   (a) When playing the audio files, fix ***x*** and choose different f_s. Describe what the audio sounds like?
   原取樣頻率 f_s = **44100(Hz)**
   當 f_s 小於 **44100** 時，會發現聲音的頻率聽起來變低了，而且撥放音樂的速度也變慢了；反之，當 f_s 大於 **44100** 時，聲音的頻率變高了，而且撥放音樂的速度也提升了。
   (b) Choose different data types of the reading file and describe what the audio sounds like.
   Audioread()函數支援兩種 datatype，single 與 double。分別以兩種不同的資料類型讀入，可以發現聽起來沒有什麼差別。
   (c) Show these waveforms in one figure with proper axis label.

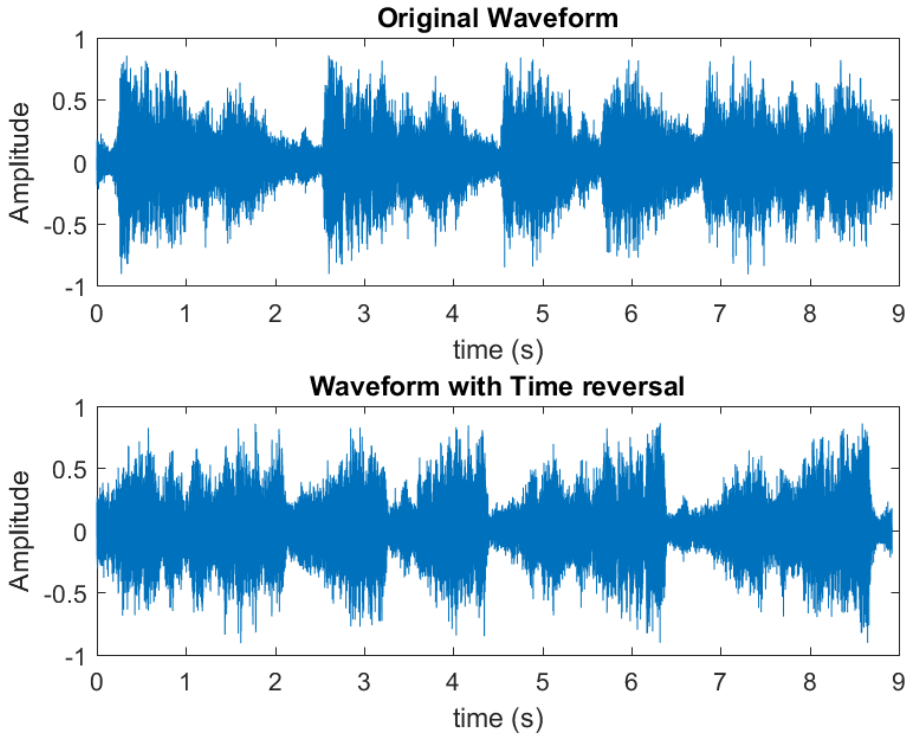


2. Redistributing the time index:
   (a) Perform ***circular shift*** by **100000** on your audio samples. You may use the function **cirshift**. Describe what these signals sound like, and show these waveforms in one figure.
   經過 **Cirshift** 的音訊，聽起來是從後半部開始撥放到尾部之後，再從頭開

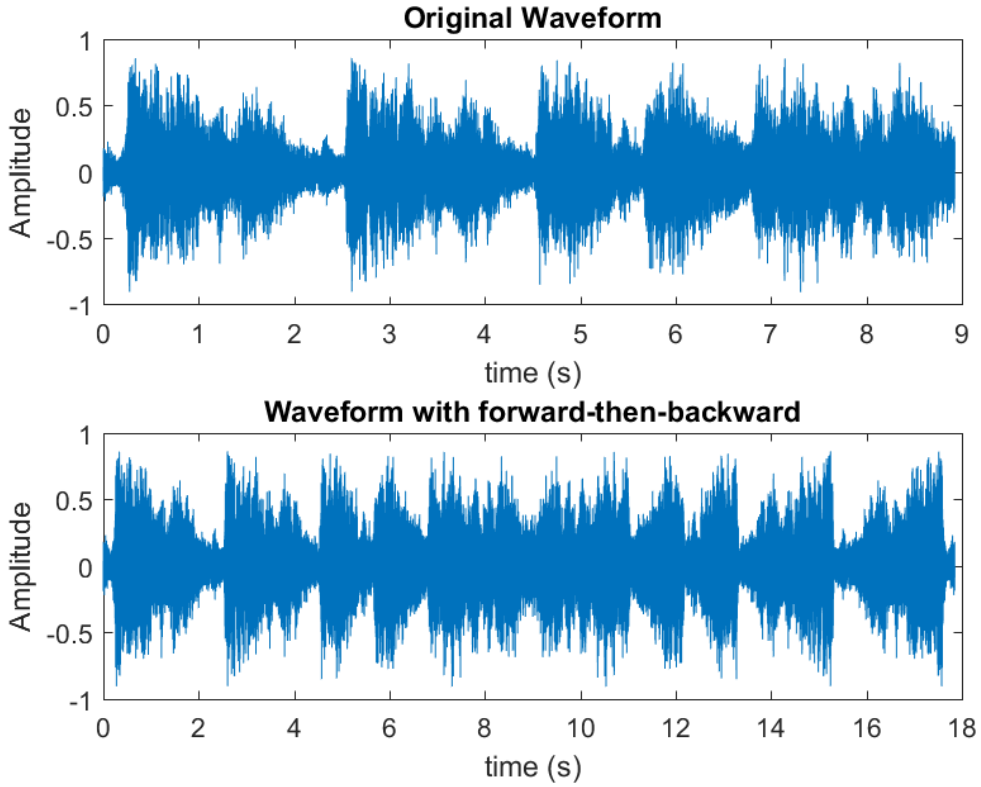
始撥放至初始撥放位置。



(b) Perform ***time-reversal*** on your audio samples. You may use the functions **fliplr** or **flipud**. Describe what these signals sound like, and show these waveforms in one figure.
聽起來像是原音訊倒轉。



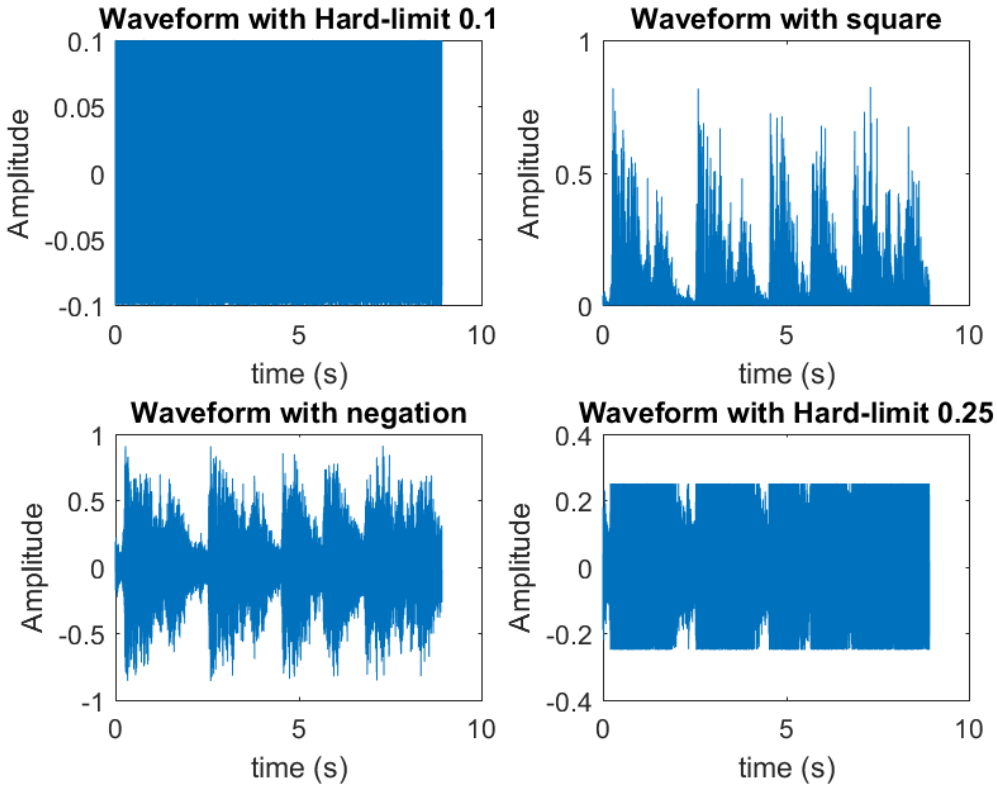
(c) (Bonus) Generate a sound clip which first plays the original clip in the forward direction and then in the backward direction. Explain how you achieve this goal.
直接將 Time reversal 的音訊傳接在原信號之後。



(d) (Bonus) Implement **upsampling** and **downsampling** for your audio file and explain how to achieve this goal.
利用函式 **upsample** 和 **downsample** 對原音訊進行處理。程式碼見 Appendix

3. Amplitude distortion:

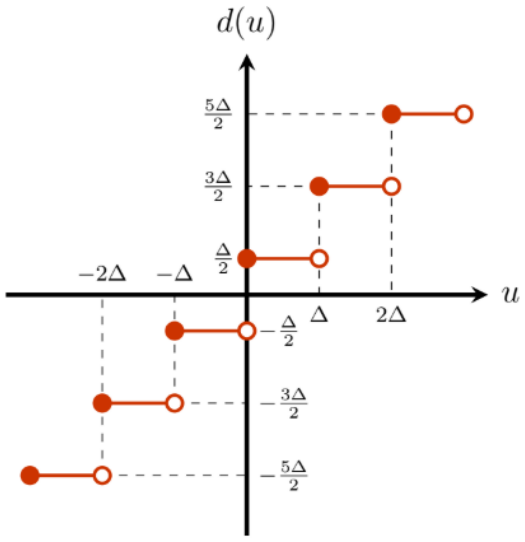
(a) Apply **hard limit** with **T = 0.1**, **squaring**, and **negation** to audio files. Describe what these signals sound like. Plot waveforms and save your results to files.
經過這三種處理後，分別會有：很重的雜音；聲音變得很不清楚也有些微雜音；聽起來和原音訊一樣。



(b) Adjust threshold T in (2). Describe what these signals sound like. Plot waveforms and save your results to files.
隨著 **T** 值變大，原音訊的成分會越來越高，雜音也會慢慢消失。

4. Quantization:

(a) Find an expression of the mapping function d( ) of the L-level quantizer.
使用 Midrise uniform quantizer。



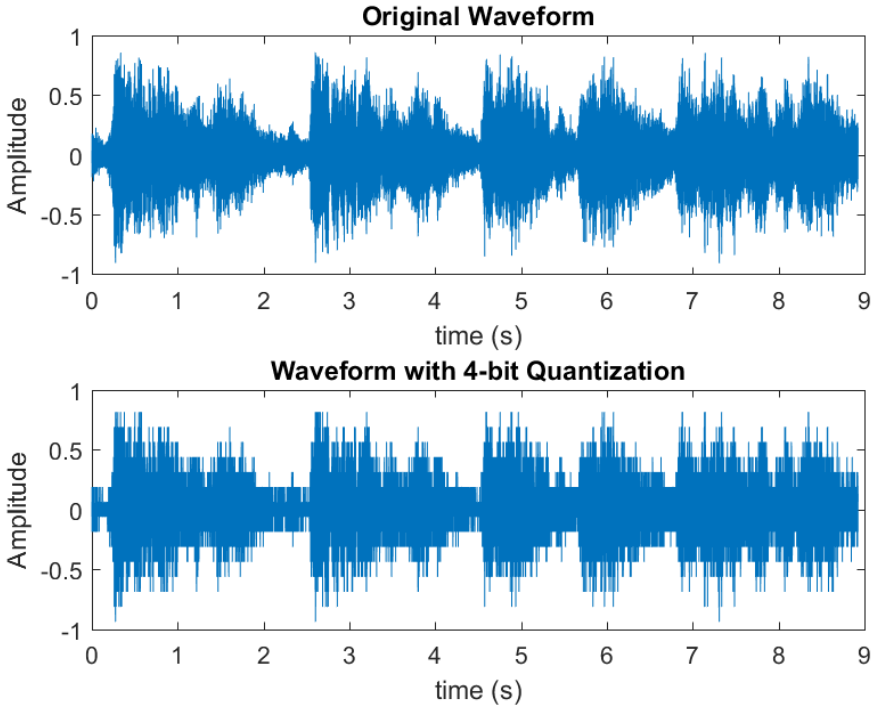
(b) Write Matlab function **quantizer_L_level** with the following arguments: **x**, the unquantized signal, **x_max**, the maximum of **x**, **L**, the number of levels, **y**, the

quantized signal.

程式碼見 Appendix

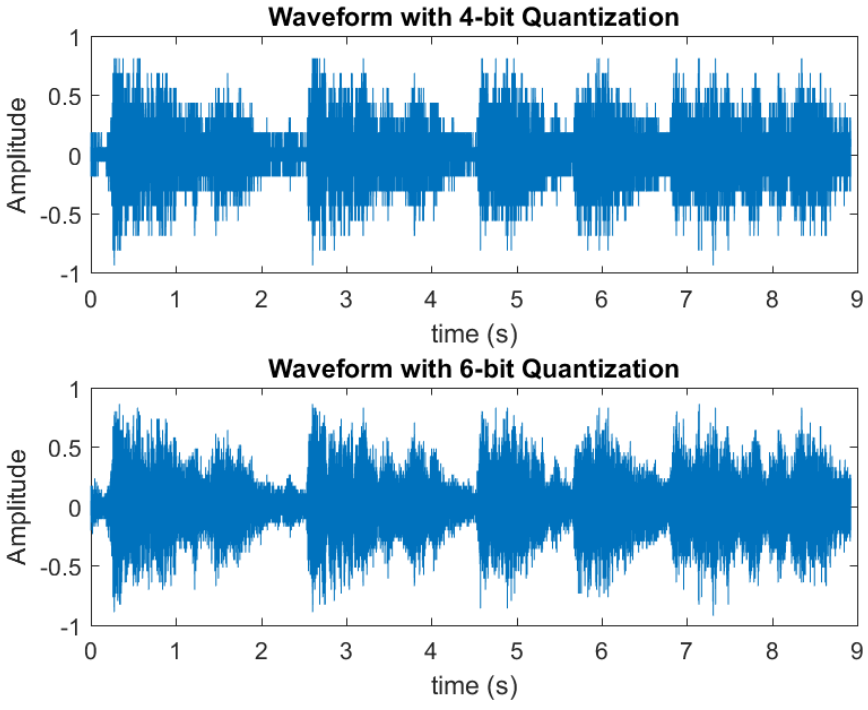
(c) Apply your quantizer to audio files with 4-bit quantization. Describe what these signals sound like. Plot waveforms and save your results to files.

還是可以聽到原本的音訊，但聲音不太清楚。

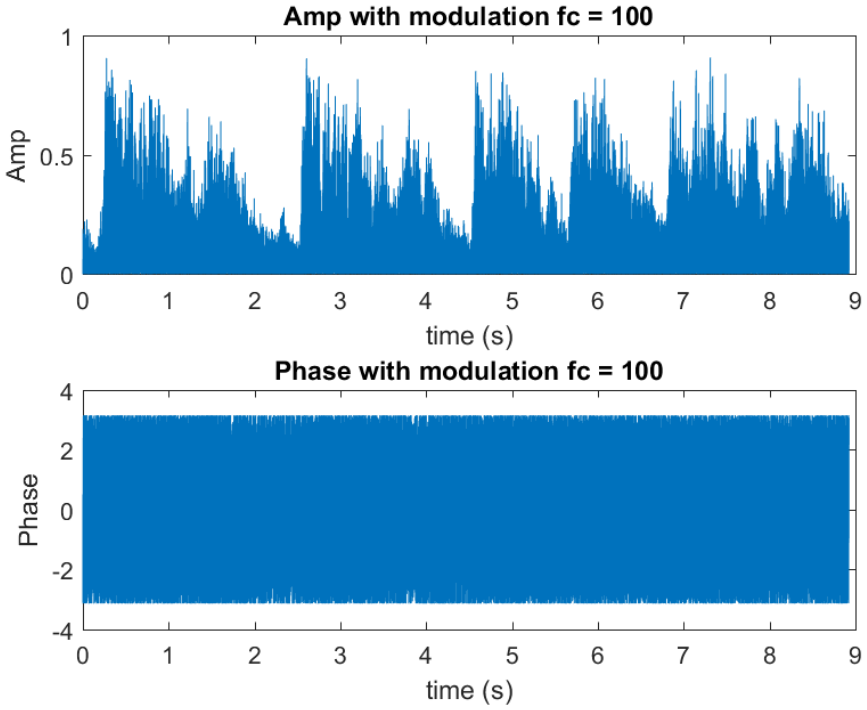


(d) Change the number of levels and repeat the previous experiment.

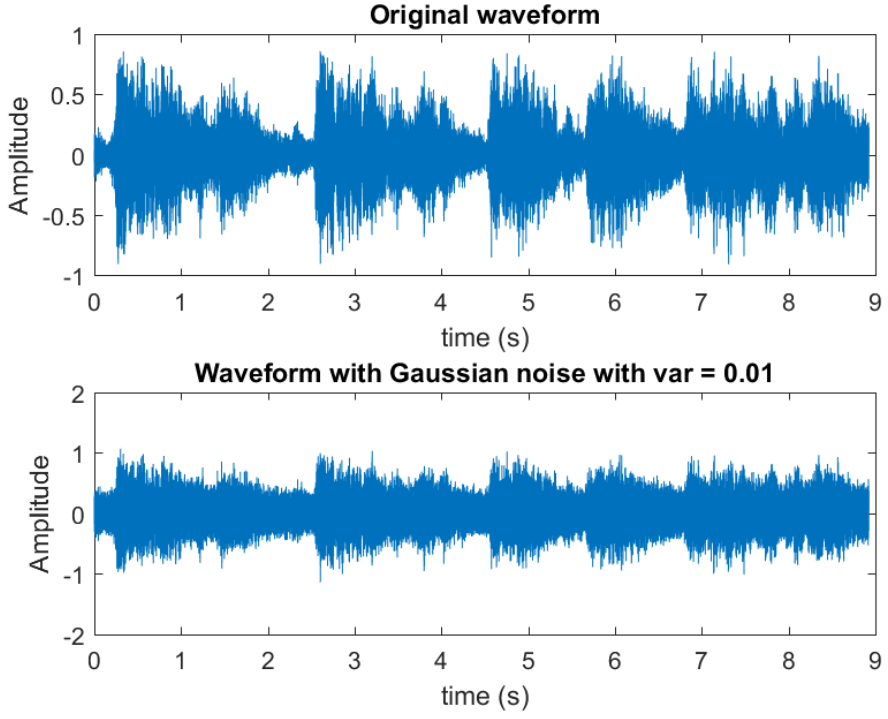
換成 6-bit quantization，可以發現聲音越來越接近原本的音訊。



5. Modulation: Describe what it sounds like, and show its waveform in one figure.
   聽起來聲音的頻率變低了



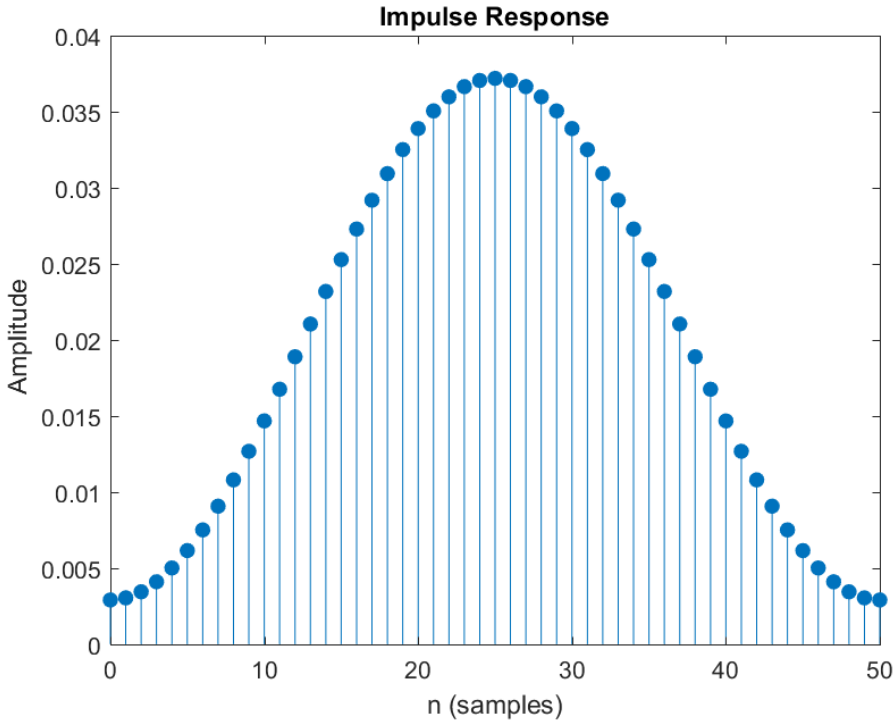
6. Noise addition: Describe what it sounds like, and show its waveform in one figure.
   可以聽到輕微雜音。



7. Filtering:

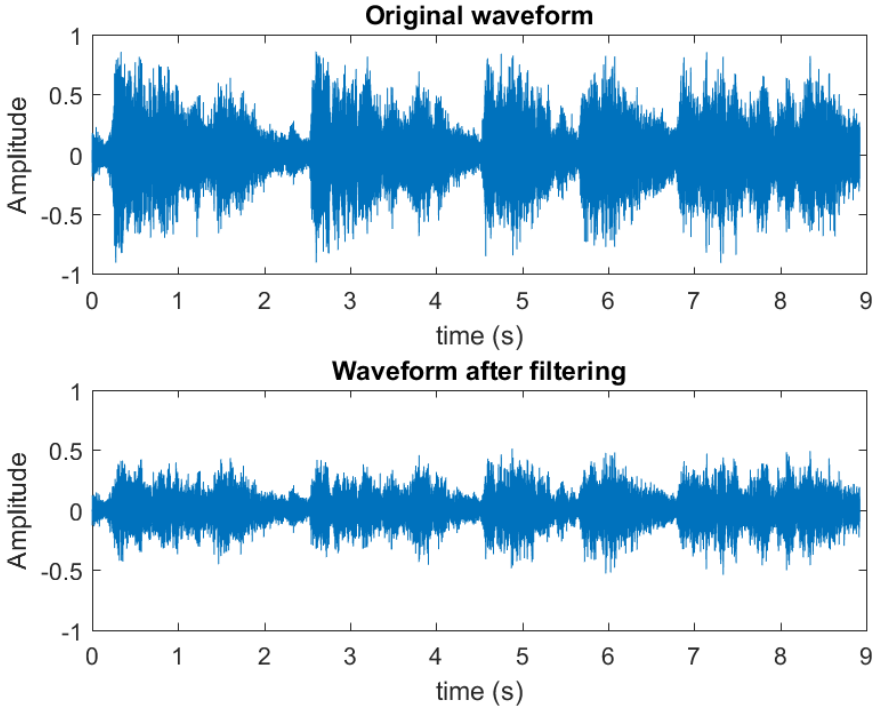
(a) Design an FIR filter to select a voice of men (94Hz to 142Hz)

利用 MATLAB 的 fir1 函式，order 設為 50。所以我們採用 50-order 的 Hamming window bandpass filter。

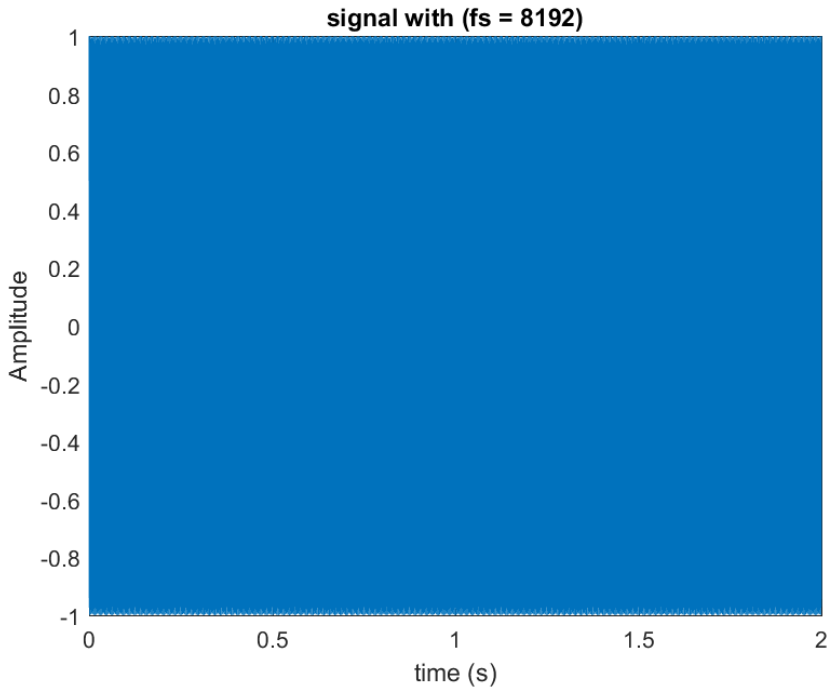


(b) Apply the filter in Problem 7a to the signal in Problem 1. Describe what it sounds like, and show its waveform in one figure.

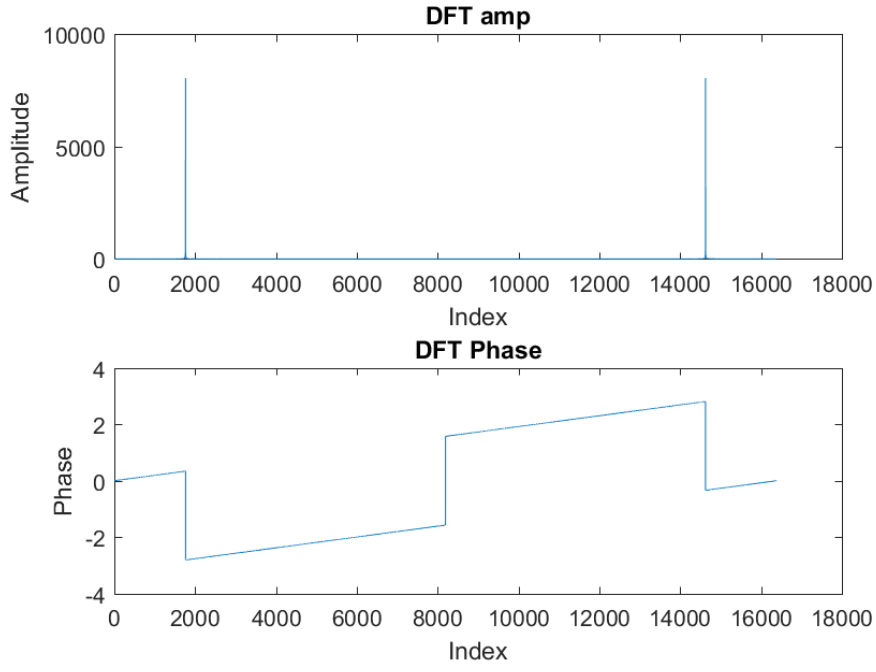
高頻的聲音(如：樂器聲)被濾掉了，只聽得到人的聲音。



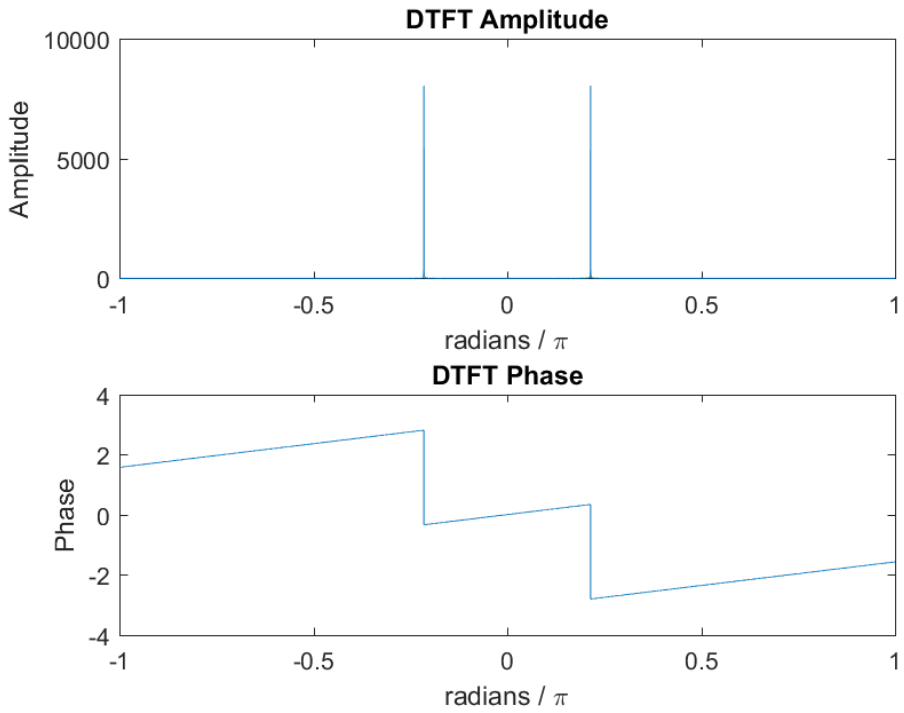
8. The Fourier Transform of sinusoids:
   (a) Assume that $x[n] = x(nT)$ are the discrete-time samples with $T = 1/fs$, and fs = 8192Hz. Plot $x(t)$ with respect to the time t in seconds, the horizontal axis is. Describe what this signal sound like
   聽起來是單頻的聲音。



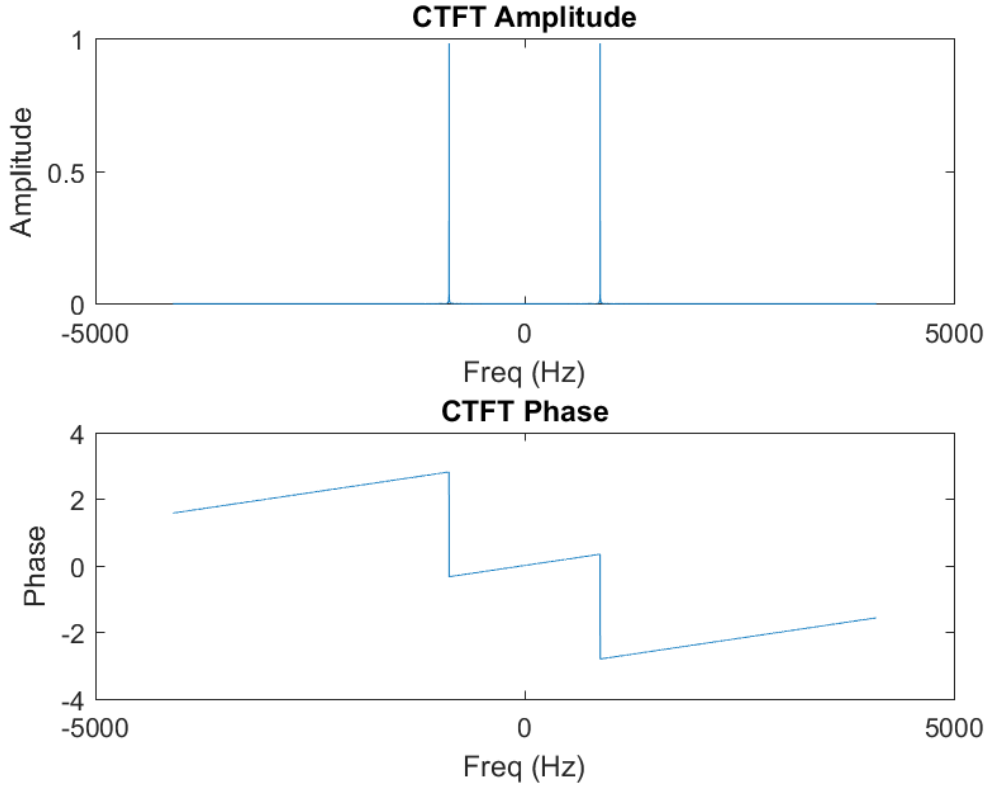
   (b) Let $x[n]$ be the discrete-time sequence of the above signal. Compute the **DFT** $X[k]$ and plot its magnitude and phase in one figure. The horizontal axis is the index k



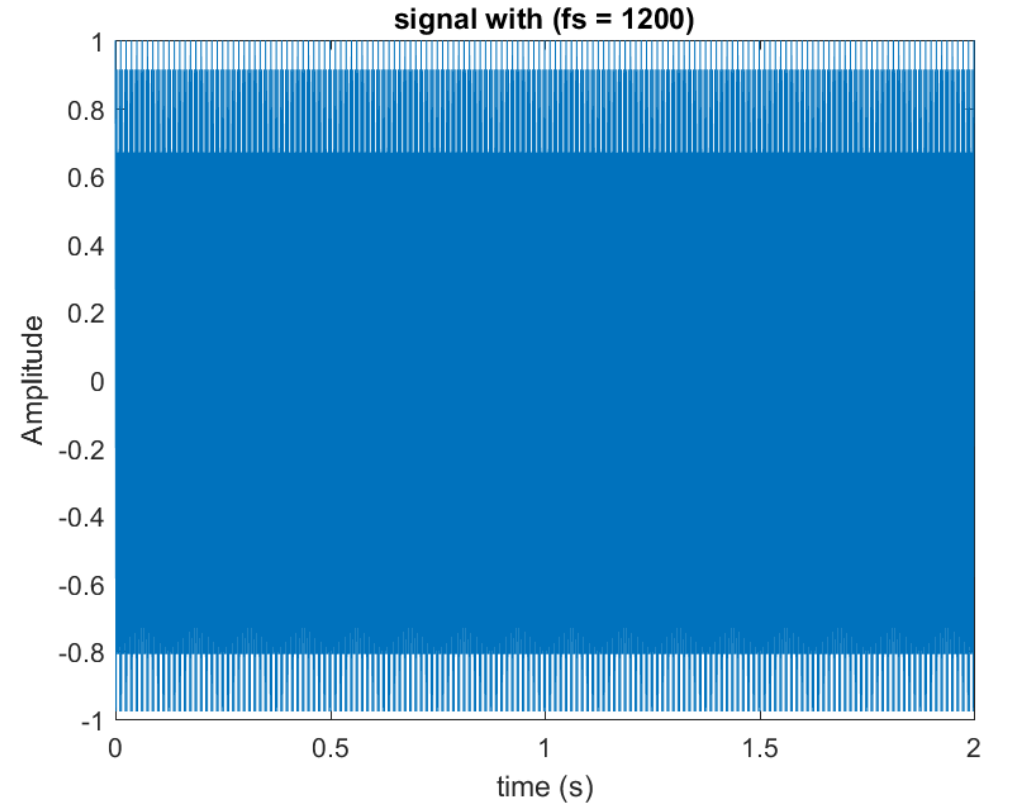
(c) Depict the magnitude and the phase of **DTFT** of $x[n]$ in one figure. The horizontal axis is ω in the range of $[-\pi, \pi]$.

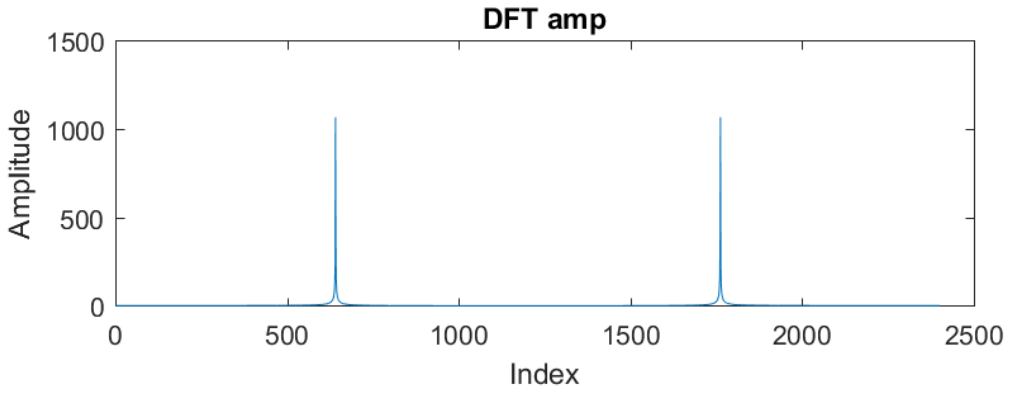


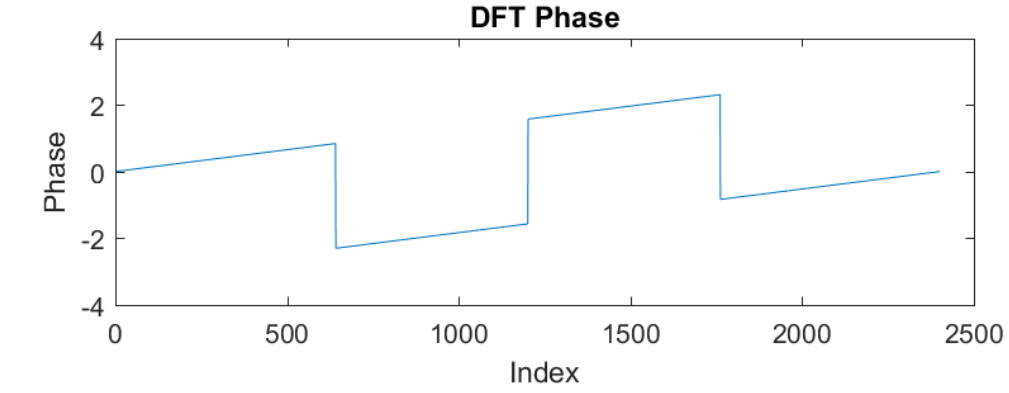
(d) Depict the magnitude and the phase of **CTFT** of $x(t)$ in one figure. The horizontal axis is the frequency f in the range of $[-f_s/2, f_s/2]$.

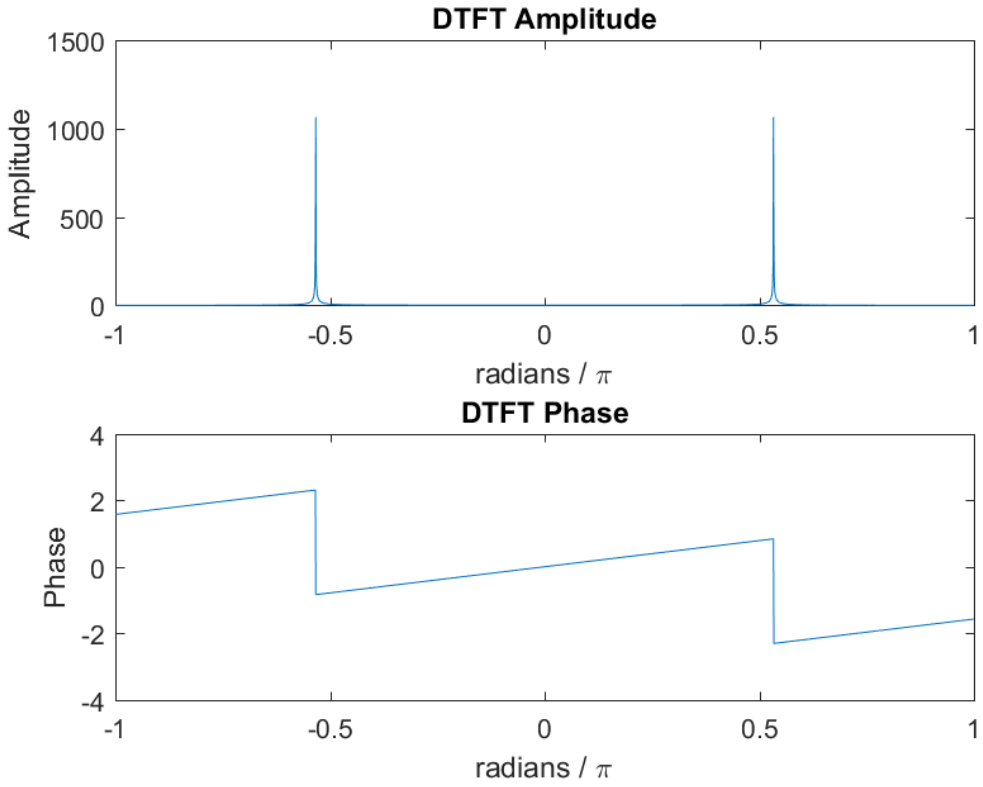


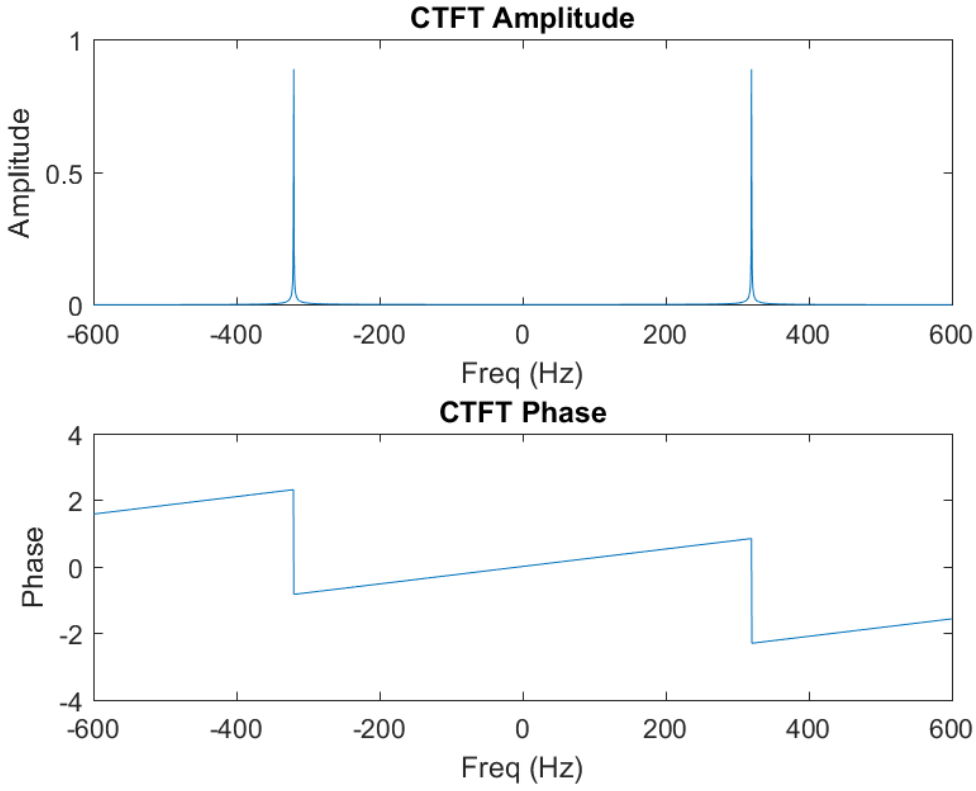
(e) Now let us assume f_s = 1200Hz in Problem 8a. Repeat Problems 8a to 8d








DTFT Amplitude
Amplitude
1500
1000
500
0
-1
-0.5
0
0.5
1
radians / π
DTFT Phase
Phase
4
2
0
-2
-4
-1
-0.5
0
0.5
1
radians / π


CTFT Amplitude
Amplitude
1
0.5
0
-600
-400
-200
0
200
400
600
Freq (Hz)
CTFT Phase
Phase
4
2
0
-2
-4
-600
-400
-200
0
200
400
600
Freq (Hz)

Appendix

1. Code for Problem1

```
%% Problem 1

% Read audio files
[x, fs] = audioread('handel.ogg', 'native');
whos x
sound(x, fs);
% changing sampling freq and plot
figure;
manip(x, fs, 1, 'Original Waveform fs = 44100');
fst = 20000;
manip(x, fst, 2, 'Waveform with fs = 20000');
fst = 80000;
manip(x, fst, 3, 'Waveform with fs = 80000');
saveas(gcf, 'Q1.png');

function manip(x, fs, num, tit)
    [row, col] = size(x);
    time = [0:1/fs:(row-1)/fs]';
    subplot(3,1,num);
    plot(time, x);
    title(tit);
    xlabel('time (s)');
    ylabel('Amplitude');
end
```

2. Code for Problem2

```
%% Problem 2

% Read audio files
[x, fs] = audioread('handel.ogg');

% 2. (a)
xt = circshift(x, 100000);
figure;
manip(x, fs, 1, 'Original Waveform');
manip(xt, fs, 2, 'Waveform with Cirshift 100000');
saveas(gcf, 'Q2a.png');
```

```
% 2. (b)
xt = flipud(x);
figure;
manip(x, fs, 1, 'Original Waveform');
manip(xt, fs, 2, 'Waveform with Time reversal');
saveas(gcf, 'Q2b.png');

% 2. (c)
xt = [x;xt];
figure;
manip(x, fs, 1, 'Original Waveform');
manip(xt, fs, 2, 'Waveform with forward-then-backward');
saveas(gcf, 'Q2c.png');

% 2. (d)
xt = upsample(x, 3);
figure;
manip(x, fs, 1, 'Original Waveform');
manip(xt, fs, 2, 'Waveform with upsample 3');
saveas(gcf, 'Q2d1.png');

xt = downsample(x, 6);
figure;
manip(x, fs, 1, 'Original Waveform');
manip(xt, fs, 2, 'Waveform with downsample 3');
saveas(gcf, 'Q2d2.png');

function manip(x, fs, num, tit)
    [row, col] = size(x);
    time = [0:1/fs:(row-1)/fs]';
    subplot(2,1,num);
    plot(time, x);
    title(tit);
    xlabel('time (s)');
    ylabel('Amplitude');
end
```

3. Code for Problem3

```
%% Problem 3
```

```
% Read audio files
[x, fs] = audioread('handel.ogg');

% 3. (a)
y = x > 0.1 | x < -0.1;
xt = x.* (1-y) + 0.1*y;
figure;
manip(xt, fs, 1, 'Waveform with Hard-limit 0.1');
xt = x.^2;
manip(xt, fs, 2, 'Waveform with square');
xt = -x;
manip(xt, fs, 3, 'Waveform with negation');

% 3. (b)
thres = 0.25;
y = x > thres | x < -thres;
xt = x.* (1-y) + thres*y;
manip(xt, fs, 4, 'Waveform with Hard-limit 0.25');
saveas(gcf, 'Q3.png');


function manip(x, fs, num, tit)
    [row, col] = size(x);
    time = [0:1/fs:(row-1)/fs]';
    subplot(2,2,num);
    plot(time, x);
    title(tit);
    xlabel('time (s)');
    ylabel('Amplitude');
end
```

4. Code for Problem4 & 5 & 6

```
%% Problem 4 5 6

% Read audio files
[x, fs] = audioread('handel.ogg');

% 4. (a) (b) (c) (d) change bit
% assume amp is in [-1, 1]
```

```
xmax = 1;
bit = 4;
bit2 = 6;
level = 2^bit;
level2 = 2^bit2;
xt = quantizer_L_level(x, xmax, level)';
xt2 = quantizer_L_level(x, xmax, level2)';
% sound(xt2, fs);
figure;
manip(xt, fs, 1, 'Waveform with 4-bit Quantization');
manip(xt2, fs, 2, 'Waveform with 6-bit Quantization');
saveas(gcf, 'Q4.png');

% 5.
fc = 100;
I = [1:size(x)];
clear i;
xt = x.*exp(i*2*pi*fc*I'/fs);
% sound(real(xt), fs);
figure;
manip2(abs(xt), fs, 1, 'Amp with modulation fc = 100', 'Amp');
manip2(angle(xt), fs, 2, 'Phase with modulation fc = 100', 'Phase');
saveas(gcf, 'Q5.png');

% 6. (Noise)
var = 0.01;
xt = x + sqrt(var)*randn(size(x));
sound(xt, fs);
figure;
manip(x, fs, 1, 'Original waveform');
manip(xt, fs, 2, 'Waveform with Gaussian noise with var = 0.01');
saveas(gcf, 'Q6.png');

function y = quantizer_L_level(x, xmax, level)
    delta = 2 * xmax / level;
    partition = [-xmax:delta:xmax];
    codebook = [0,-(level-1)*delta/2:delta:(level-1)*delta/2,0];
    [I, y] = quantiz(x,partition,codebook);
```

```
end


function manip(x, fs, num, tit)
    [row, col] = size(x);
    time = [0:1/fs:(row-1)/fs]';
    subplot(2,1,num);
    plot(time, x);
    title(tit);
    xlabel('time (s)');
    ylabel('Amplitude');
end

function manip2(x, fs, num, tit, yl)
    [row, col] = size(x);
    time = [0:1/fs:(row-1)/fs]';
    subplot(2,1,num);
    plot(time, x);
    title(tit);
    xlabel('time (s)');
    ylabel(yl);
end
```

5. Code for Problem7

```
%% Problem 7

% Read audio files
[x, fs] = audioread('handel.ogg');

% 7. (Filtering)
W = 50;
lower_freq = 94 / fs;
higher_freq = 142 / fs;
h = fir1(W, [lower_freq, higher_freq]);
impz(h);
saveas(gcf, 'Q7a.png');
% freqz(h,1,512)
xt = filter(h, 1, x);
sound(xt, fs);
figure;
```

```
manip(x, fs, 1, 'Original waveform');
manip(xt, fs, 2, 'Waveform after filtering');
saveas(gcf, 'Q7b.png');


function manip(x, fs, num, tit)
    [row, col] = size(x);
    time = [0:1/fs:(row-1)/fs]';
    subplot(2,1,num);
    plot(time, x);
    title(tit);
    xlabel('time (s)');
    ylabel('Amplitude');
end
```

6. Code for Problem8

```
%% Problem 8

% 8. (a)
fc = 880;
fs = 1200;
I = [0:1/fs:2];
x = cos(2*pi*fc*I);
sound(x, fs);
figure;
plot(I, x);
title('signal with (fs = 1200)');
xlabel('time (s)');
ylabel('Amplitude');
saveas(gcf, 'Q8a.png');

% 8. (b)
[col, row] = size(x);
y = fft(x, row);
figure;
subplot(2,1,1);
plot(abs(y));
title('DFT amp')
xlabel('Index');
ylabel('Amplitude');
```

```
subplot(2,1,2);
plot(angle(y));
title('DFT Phase')
xlabel('Index');
ylabel('Phase');
saveas(gcf, 'Q8b.png');

N = row;
w = 2*pi * (0:(N-1)) / N;
w2 = fftshift(w);
w3 = unwrap(w2 - 2*pi);
figure;
subplot(2,1,1);
plot(w3/pi, abs(fftshift(y)));
title('DTFT Amplitude')
xlabel('radians / \pi');
ylabel('Amplitude');
subplot(2,1,2);
plot(w3/pi, angle(fftshift(y)));
title('DTFT Phase')
xlabel('radians / \pi');
ylabel('Phase');
saveas(gcf, 'Q8c.png');

y = y / fs;
figure;
subplot(2,1,1);
plot(w3/pi*fs/2, abs(fftshift(y)));
title('CTFT Amplitude')
xlabel('Freq (Hz)');
ylabel('Amplitude');
subplot(2,1,2);
plot(w3/pi*fs/2, angle(fftshift(y)));
title('CTFT Phase')
xlabel('Freq (Hz)');
ylabel('Phase');
saveas(gcf, 'Q8d.png');
```