```
% 19ucc023
% Mohit Akhouri
% Experiment 9 - Observation 2

% This code will take the input of an audio signal with hissing
 component
% frequency greater than pi/2. Next it will design a low pass filter
 with
% frequency spectrum as : value '1' for all samples except for 66th
 and
% 192th samples , i.e. we design a IDEAL LOW PASS FILTER

% Finally we will divide the audio file into chunks of 1x256 and find
 the
% DFT for each of the chunk. Let it be X(w) . Finally we multiply the
% filter impulse response H(w) with X(w) to get the smoothened version
 of
% audio signal with hissing sound removed.

[x,fs] = audioread('inputwithhissgreaterthanpiby2.wav'); % reading an
 audio file
x = x'; % Taking the transpose to convert from Nx1 to 1xN for easy
 multiplication later on
x_length = length(x); % Finding length of audio signal x[n]

% plot of the original audio signal x[n] + unwanted components (hiss
 sound)
figure;
plot(x);
xlabel('samples(n) ->');
ylabel('x[n] ->');
title('19ucc023 - Mohit Akhouri','Plot of original audio signal x[n] +
 hissing frequency components');
grid on;

samples = 256; % variable to store the sample number for partition of
 audio file
hw = ones(1,256); % ideal low pass filter impulse response

% making the 66th and 192th sample number = 0
hw(66) = 0;
hw(192) = 0;

% plot of the impulse response of the IDEAL LOW PASS FILTER
figure;
plot(hw,'Linewidth',1.5);
xlabel('frequency (radians/sec) ->');
ylabel('H(\omega) ->');
title('19ucc023 - Mohit Akhouri','Impulse response H(\omega) of an
 IDEAL LOW PASS FILTER for removing hissing frequency components');
grid on;
```

```
% main loop algorithm for dividing the audio file into chunks and
% individually applying fft on each chunk and multiplying with filter
% impulse response to get the smoothened version of audio signal
for i=1:samples:x_length
    x_sampled = x(i:i+255); % taking 256 samples chunk
    x_sampled_fft = fft(x_sampled,256); % taking DFT of the 256
 samples
    yw = x_sampled_fft .* hw; % filtering process takes place
    x(i:i+255) = ifft(yw); % rewriting the original audio file with
 smoothened version of audio
end

x = x'; % taking transpose back again to convert to original form
 ( Nx1 )

% Plot of the smoothened version of audio signal x[n]
figure;
plot(x);
xlabel('samples(n) ->');
ylabel('x[n] ->');
title('19ucc023 - Mohit Akhouri','Plot of smoothened audio signal
 after removing the hissing frequency components');
grid on;

sound(x,fs); % listening to the smoothened version of the audio signal
 after passing through IDEAL LOW PASS FILTER
audiowrite('Exp9_obs_2_low_pass_filter_output.wav',x,fs); % Writing
 the final audio signal to an audio file
```

Published with MATLAB® R2020b