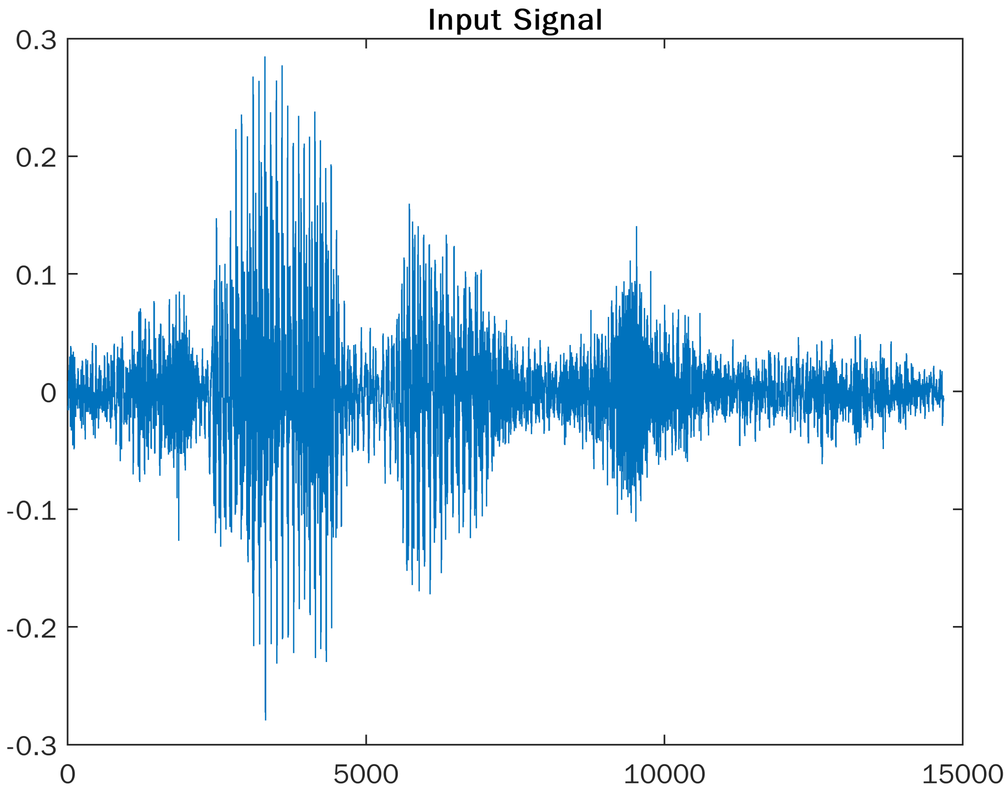
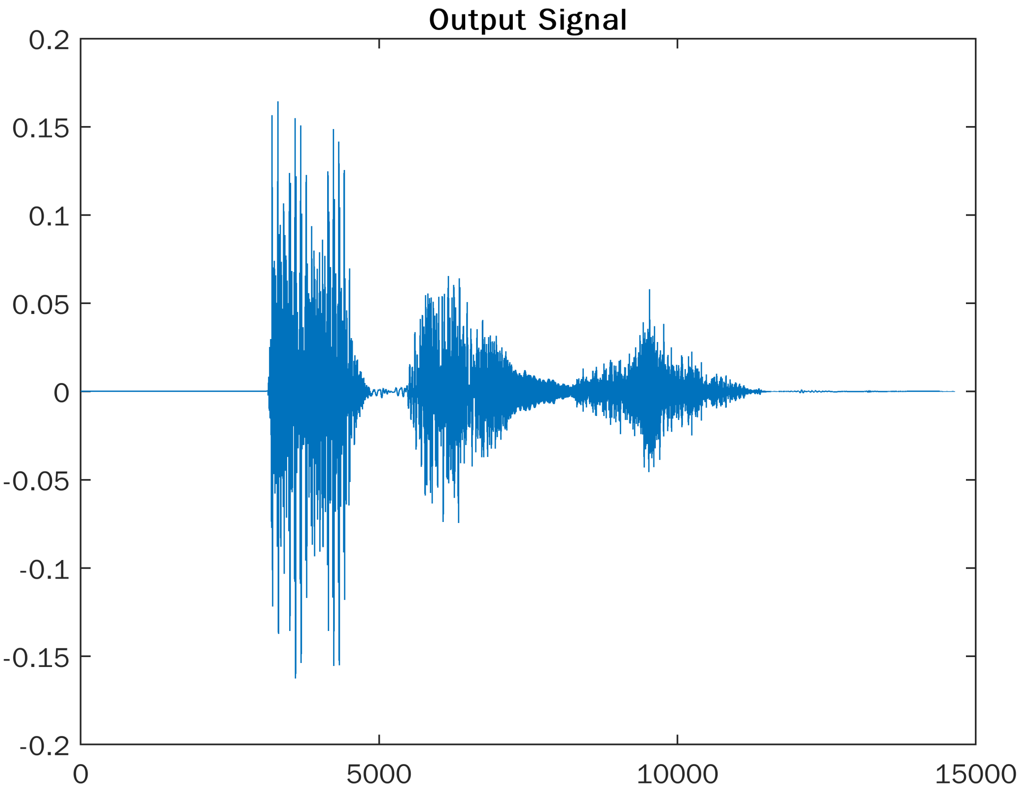
```
%% Start of Instructions

% Function to read speech file from Local Drive
[file,path] = uigetfile('*.wav','Pick an Audio Wave File');
handles.filename=file;
% Open standard dialog box for retrieving files
% To browse the file from the coding folder
if isequal(file,0) || isequal(path,0)
    % If you not selecting file or click the cancel button
    warndlg('You should Have to Select a File First');
    % Open warning dialog box with message "You should Have to Select an file First"
    % Means it display the dialog box
else
    [y,fs]= audioread(file);
    % Read the wave file using waveread; function and store in 'y'
  %k=[24000,1]
  [actual, Fs] = audioread(file);
end

x=actual;
fs=Fs;

%% Initialization

    IS = 0.2;
    % window length
    window_length = 0.01;
    wnd_length = fix(window_length*fs);
    % hamming window calculations
    wnd = hamming(wnd_length);
    inc = fix(wnd_length / 2);
    [f,t] = cut_frame(x,wnd,inc);
    silentFrameNo=fix((IS*fs-wnd_length)/(0.5*wnd_length) +1);
    f=f';
    Y = fft(f);
    [m,n] = size(Y);
    YPhase = angle(Y((1:(fix(m/2)+1)),:));
    YY = abs(Y(1:(fix(m/2)+1),:));
    noise_est = mean(YY(:,1:silentFrameNo)'.^2)';
    X=zeros(size(YY));
    % transferring command to wiener filter
    X = wienerfilter(YY,silentFrameNo,noise_est);

    XX = X.*exp(1i*YPhase);
if mod(size(f,1),2)
    XX=[XX;flipud(conj(XX(2:end,:)))];
else
    XX=[XX;flipud(conj(XX(2:end-1,:)))];
end
    s_out = real(ifft(XX));
    Output = add_overlap(s_out',wnd,inc);
    y = Output;

figure;
plot(x)
title('Input Signal')
```



```
figure;
plot(y)
title('Output Signal')
```



## End of Instructions

```
function y=add_overlap(f,win,inc)
    [m,n] = size(f);
    w = win';
    n_buf = ceil(n/inc);
    buf_len = n + (m-1)*inc;
    y_tmp = zeros(buf_len,n_buf);
    y_tmp(repmat(1:n,m,1)+repmat((0:m-1)'*inc+rem((0:m-1)',n_buf)*buf_len,1,n)) = f.*re
    y = sum(y_tmp,2);
end
%% we cut the signal into frames of samples

function [f,t]=cut_frame(signal,window,increment)
            L = length(signal);
            W = length(window);
            Len = W;
            Nf = fix((L-W)/increment)+1;
            f = zeros(Nf,Len);
            Ind=(repmat((0:(Nf-1))'*increment,1,Len)+repmat(1:Len,Nf,1))';
            windowMatrix=repmat(window,1,Nf);
            f=signal(Ind).*windowMatrix;
            if nargout > 1
```

```
                t = (1+W)/2+ increment*(0:(Nf-1)).';
            end
            f=f';
end
function X = wienerfilter(YY,silentFrameNo,noise_est)
    a = 0.95;
    Gain = ones(size(noise_est));
    X=zeros(size(YY));
    last_post_SNR=Gain;
    frame_Num = size(YY,2);

     for i=(silentFrameNo+1):frame_Num
        current_post_SNR=(YY(:,i).^2)./noise_est;
        % posterior SNR  = Y^2/Noise^2;
        prior_SNR=a*(Gain.^2).*last_post_SNR+(1-a).*max(current_post_SNR-1,0);
        % prior SNR =X^2 / Noise^2 = (Gain*Y)^2/Noise^2 = Gain^2 * posterior SNR
        last_post_SNR = current_post_SNR;
        % record current posterior SNR;
        Gain=(prior_SNR./(prior_SNR+1));
        % gain = prior SNR/(prior SNR+1) = X^2 / (X^2 + Noise^2)
        X(:,i)=Gain.*YY(:,i);
     end

end
```