```
% Assignment 1 : Peak Amplitude Detection : Real Time Processing
%       1. Use a 4th order low pass filter at 100 Hz
%       2. Try to predict the peak amplitude (absolute local maxima) that will occur
%       3. Try different techniques. You can verify the effectiveness of different↙
technique


fc = 100; %Cut off Frequency  = 100 Hz
fs = 88200; %Sampling  Frequency = 88200Hz

[x,Fs] = audioread('C:\Users\Suhas\Desktop\NeroSoundTrax_test3_PCM_Mono_CBR_8SS_8000Hz.↙
wav')
%x = transpose(x);
% Reading an audio input file

%x = sin(2*pi*fc*(1:88219)/88219)+randn(1,88219);
%x = sin(2*pi*fc*(1:fs)/fs);
%x =  randn(1,10000);


samples = size(x,2); %Determining the size of the input
index = 1; %Pointer to keep track of peak and low samples
blk = 100; %Block Size for Block Processing



%Preprocessing the input signal to find the local maxima and minima
% Array a() hold the preprocessed output of the input signal
%z[];
blk_no = samples/blk;

%To check if Number of Blocks is an Integer or not
check_int = isinteger(blk_no);
if(check_int==0)
    blk_no = ceil(blk_no); %Convert block size to an integer value
end


mod_op = mod(samples,blk);
if(mod_op)
    append_zero = blk - mod_op;
    x = [x,zeros(1,append_zero)]; %Appending Zeros to the input signal to make the↙
block size dynamic
end

%Block Processing of Input Data for Peak Amplitude Prediction
z=[];
a=[];

for j = 0 :blk_no-1
for i = ((j*blk)+2) :(((j+1)*blk)-1)
```

```
    a((j+1)*blk)=0;
    if(x(i)>0 || x(i)==0)
        if((x(i)>x(i-1)) && (x(i)>x(i+1)))
            peak = x(i);
            a(index:i) = peak;
            index = i;
        elseif((x(i)<x(i-1)) && (x(i)<x(i+1)))
            low = x(i);
            a(index:i) = low;
            index = i;
        end

    else
        if((x(i)<x(i-1)) && (x(i)<x(i+1)))
            low = x(i);
            a(index:i) = low;
            index = i;
        elseif((x(i)>x(i-1)) && (x(i)>x(i+1)))
            peak = x(i);
            a(index:i) = peak;
            index = i;
        end
    end
    index = index;

end
aa = lpc(a,4);
yout = filter([0 -aa(2:end)],1,a);

end


%Plotting the input signal with respect to the preprocessed to signal
plot(a,'r--');
hold on
plot(x,'y');
hold off;
xlabel 'Number of samples'
ylabel 'Amplitude';
title 'Input Signal and Processed Signal';



                        %%Filter Processing%%
% [a,g] = lpc(x,p) finds the coefficients of a pth-order linear predictor
% (FIR filter) that predicts the current value of the real-valued time series
% x based on past samples.

% p is the order of the prediction filter polynomial, a = [1 a(2) ... a(p+1)].
% If p is unspecified, lpc uses as a default p = length(x)-1. If x is a
% matrix containing a separate signal in each column, lpc returns a model
```

```
% estimate for each column in the rows of matrix a and a column vector of
% prediction error variances g.
% The length of p must be less than or equal to the length of x.

%aa = lpc(a,4);

%Predicted output of a 4th Order Linear FIR Filter
%yout = filter([0 -aa(2:end)],1,a);



                            %Post Processing
%Normalizing the output value to match the amplitude of the input signal
for c = 1 : size(yout,2)
val(c) =a(c)-yout(c); %Holds the difference in value to the output and input
yout(c) = yout(c)+val(c); %corrects the output signal by adding the difference value
end



%Plotting the input and amplitude predicted signal

figure
subplot(3,1,1)
plot(yout,'r--');
xlabel 'Number of samples'
ylabel 'Amplitude';
title 'Amplitude Predicted Signal';
legend('Predicted Signal');

subplot(3,1,2)
plot(x,'y');
xlabel 'Number of samples'
ylabel 'Amplitude';
title 'Input Signal';
legend('Input Signal');

subplot(3,1,3)
plot(yout,'r--');
hold on
plot(x,'y');
hold off;
xlabel 'Number of samples'
ylabel 'Amplitude';
title 'Input Signal and Amplitude Predicted Signal';
legend('Predicted Signal','Input Signal');

figure
plot(yout,'r--');
hold on
plot(x,'y');
```

```
hold off;
xlabel 'Number of samples'
ylabel 'Amplitude';
title 'Input Signal and Amplitude Predicted Signal';
legend('Predicted Signal','Input Signal');
```