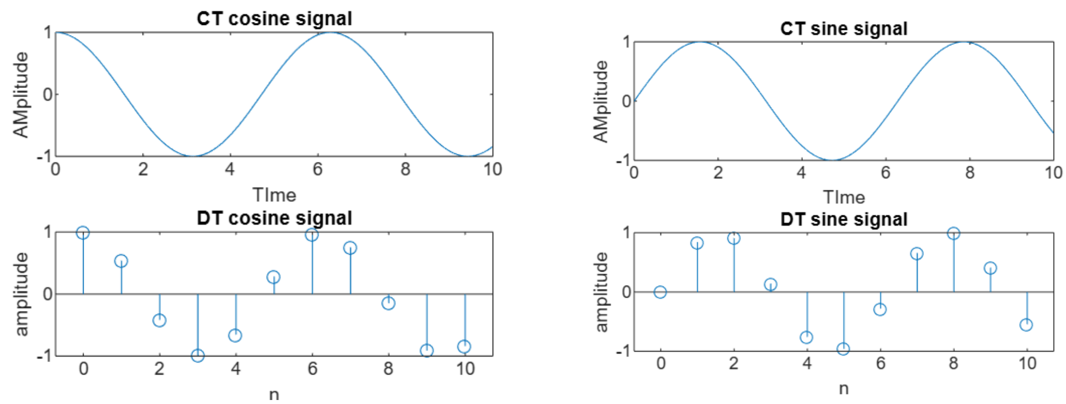
Exp 1: Time domain representation of continuous time (CT) and Discrete time (DT)signals.

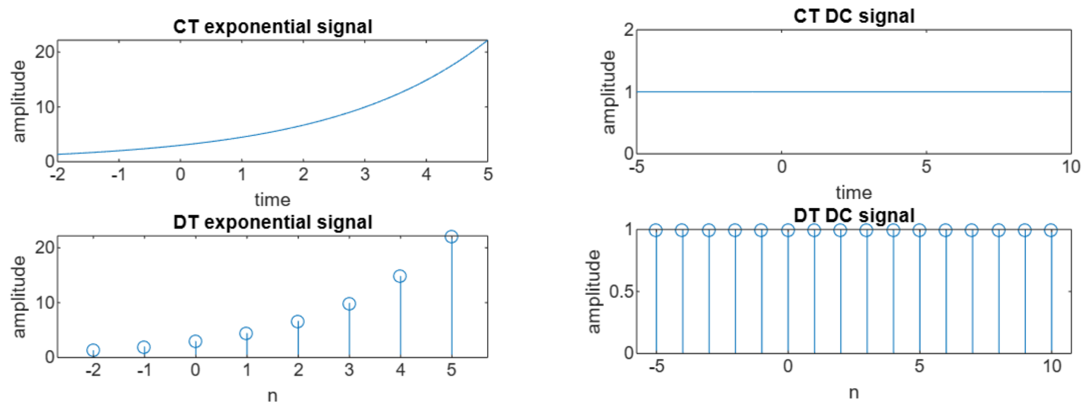
```
%cosine signal
t=0:0.1:10;
y_ct=cos(t);
n=0:1:10;
y_dt=cos(n);
figure;
subplot(2,1,1);
plot(t,y_ct);
xlabel("TIme");
ylabel("AMplitude");
title("CT cosine signal");
subplot(2,1,2);
stem(n,y_dt);
xlabel("n");
ylabel("amplitude");
title("DT cosine signal");
% sine signal
t=0:0.1:10;
y_ct=sin(t);
n=0:1:10;
y_dt=sin(n);
figure;
subplot(2,1,1);
plot(t,y_ct);
xlabel("TIme");
ylabel("AMplitude");
title("CT sine signal");
subplot(2,1,2);
stem(n,y_dt);
xlabel("n");
ylabel("amplitude");
title("DT sine signal");
% exponential wave
t=-2:0.1:5;
x_ct=3*exp(0.4*t);
n=-2:1:5;
x_dt=3*exp(0.4*n);
figure;
subplot(2,1,1);
plot(t,x_ct);
xlabel("time");
ylabel("amplitude");
title("CT exponential signal");
subplot(2,1,2);
stem(n,x_dt);
xlabel("n");
ylabel("amplitude");
title("DT exponential signal");
```

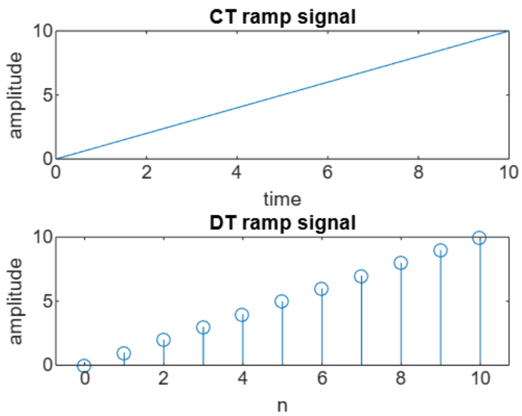
```
%DC signal
t=-5:1:10;
u_ct=ones(size(t));
n=-5:1:10;
u_dt=ones(size(n));
figure;
subplot(2,1,1);
plot(t,u_ct);
xlabel("time");
ylabel("amplitude");
title("CT DC signal");
subplot(2,1,2);
stem(n,u_dt);
xlabel("n");
ylabel("amplitude");
title("DT DC signal");
%ramp
t=0:0.1:10;
r_ct=t;
n=0:1:10;
r_dt=n;
figure;
subplot(2,1,1);
plot(t,r_ct);
xlabel("time");
ylabel("amplitude");
title("CT ramp signal");
subplot(2,1,2);
stem(n,r_dt);
xlabel("n");
ylabel("amplitude");
title("DT ramp signal");
%triangluar
t=0:0.01:1;
tri_ct=sawtooth(2*pi*5*t,0.5);
n=0:10;
tri_dt=sawtooth(2*pi*5*n/100,0.5);
figure;
subplot(2,1,1);
plot(t,tri_ct);
xlabel("time");
ylabel("amplitude");
title("CT triangle signal");
subplot(2,1,2);
stem(n,tri_dt);
xlabel("n");
ylabel("amplitude");
title("DT triangle signal");
%square
```

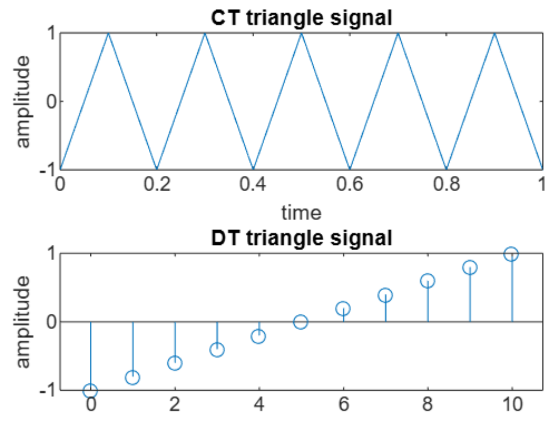
```
t=0:0.001:0.5;
s_ct=square(2*pi*8*t,50);
n=0:1:100;
s_dt=square(2*pi*8*n/100,50);
figure;
subplot(2,1,1);
plot(t,s_ct);
xlabel("time");
ylabel("amplitude");
title("CT square signal");
subplot(2,1,2);
stem(n,s_dt);
xlabel("n");
ylabel("amplitude");
title("DT square signal");
```

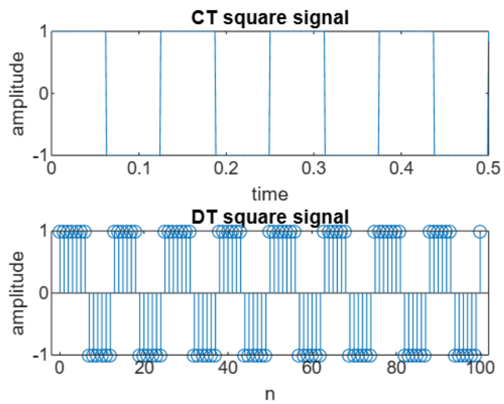
Output:









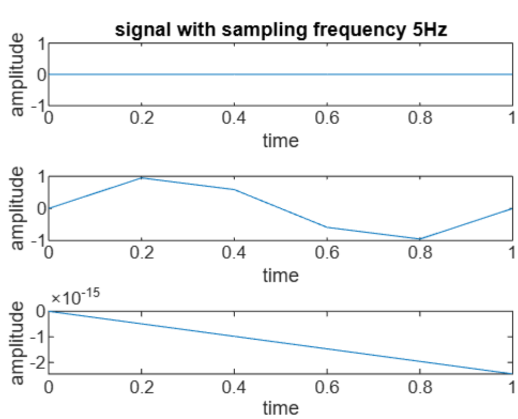


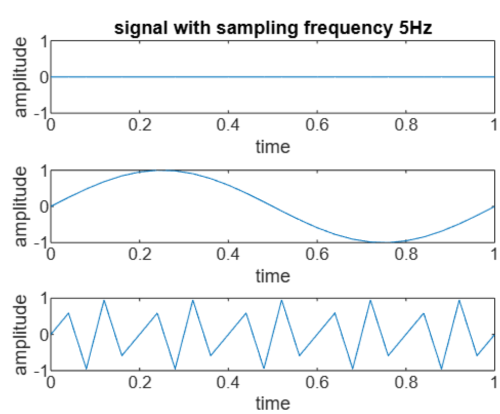
EXP 2 : sampling theorem and aliasing effects with various sampling frequencies.(5hz,25hz,50hz)

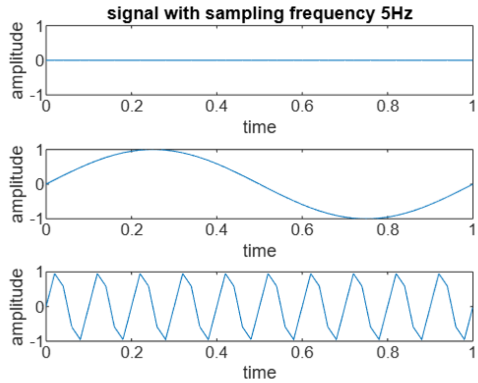
```
fs1=5;
t1=0:1/fs1:1;
a1=sin(2*pi*0*t1);
a2=sin(2*pi*1*t1);
a3=sin(2*pi*10*t1);
figure;
subplot(3,1,1);
plot(t1,a1);
title("signal with sampling frequency 5Hz");
xlabel("time");
ylabel("amplitude");
subplot(3,1,2);
plot(t1,a2);
xlabel("time");
ylabel("amplitude");
subplot(3,1,3);
plot(t1,a3);
xlabel("time");
ylabel("amplitude");
fs2=25;
t2=0:1/fs2:1;
b1=sin(2*pi*0*t2);
```

```
b2=sin(2*pi*1*t2);
b3=sin(2*pi*10*t2);
figure;
subplot(3,1,1);
plot(t2,b1);
title("signal with sampling frequency 5Hz");
xlabel("time");
ylabel("amplitude");
subplot(3,1,2);
plot(t2,b2);
xlabel("time");
ylabel("amplitude");
subplot(3,1,3);
plot(t2,b3);
xlabel("time");
ylabel("amplitude");
fs3=50;
t3=0:1/fs3:1;
c1=sin(2*pi*0*t3);
c2=sin(2*pi*1*t3);
c3=sin(2*pi*10*t3);
figure;
subplot(3,1,1);
plot(t3,c1);
title("signal with sampling frequency 5Hz");
xlabel("time");
ylabel("amplitude");
subplot(3,1,2);
plot(t3,c2);
xlabel("time");
ylabel("amplitude");
subplot(3,1,3);
plot(t3,c3);
xlabel("time");
ylabel("amplitude");
```

Output:



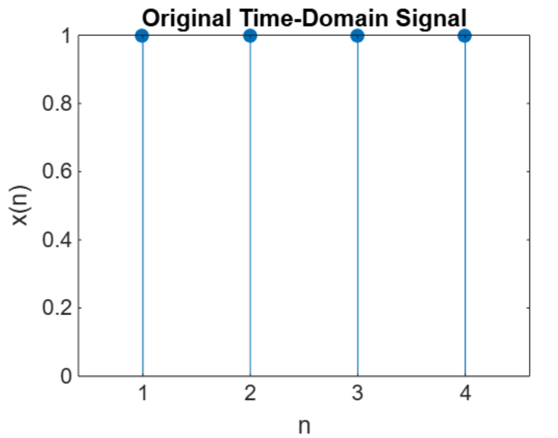


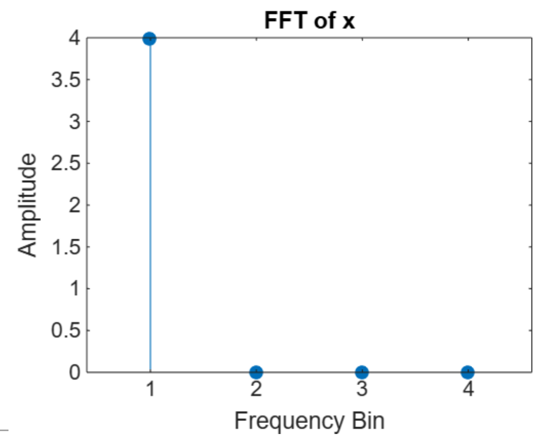


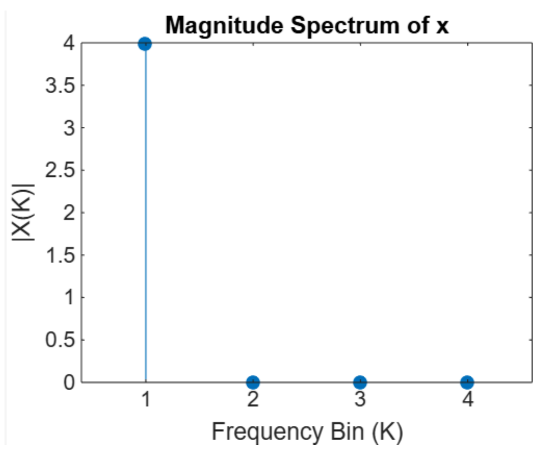
EXP 3: frequency domain analysis of the signal using FFT.

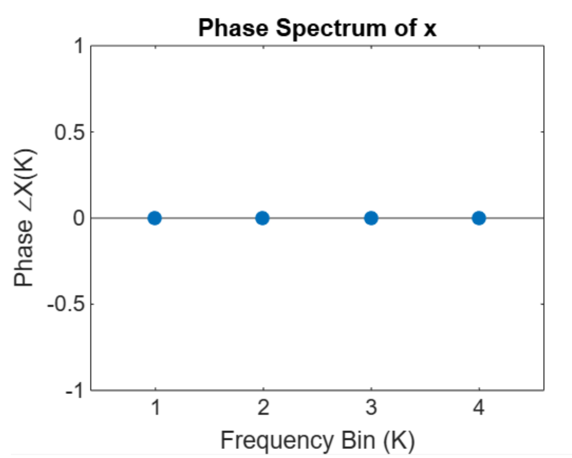
```
x = [1 1 1 1];
X = fft(x);             % FFT of the signal
X_mag = abs(X);         % Magnitude of FFT
X_ang = angle(X);       % Phase (angle) of FFT
% Original time-domain signal
figure;
stem(x, 'filled');
xlabel('n');
ylabel('x(n)');
title('Original Time-Domain Signal');
% FFT of x
figure;
stem(X, 'filled');
xlabel('Frequency Bin');
ylabel('Amplitude');
title('FFT of x');
% Magnitude of FFT
figure;
stem(X_mag, 'filled');
xlabel('Frequency Bin (K)');
ylabel('|X(K)|');
title('Magnitude Spectrum of x');
% Phase of FFT
figure;
stem(X_ang, 'filled');
xlabel('Frequency Bin (K)');
ylabel('Phase ∠X(K)');
title('Phase Spectrum of x');
% Inverse FFT
x_rec = ifft(X)
% Some basic statistics on the magnitude
mean_val = mean(X_mag)
min_val = min(X_mag)
var_val = var(X_mag)
std_val = std(X_mag)
```

OUtput:









EXP 4: N point circular convolution.

```
x = [1 2 3];
h = [1 1 1];
X = fft(x);
H = fft(h);
Y = X .* H;
y_circular = ifft(Y);

l1 = length(x);
l2 = length(h);
N = l1 + l2 - 1;          % Output length for linear convolution
x_pad = [x, zeros(1, N - l1)];
h_pad = [h, zeros(1, N - l2)];
X_pad = fft(x_pad);
H_pad = fft(h_pad);
Y_pad = X_pad .* H_pad;
y_linear = ifft(Y_pad);

figure;
stem(0:length(y_circular)-1, real(y_circular));
title('Circular Convolution using FFT');
```

```
xlabel('n');
ylabel('y[n]');
grid on;

figure;
stem(0:length(y_linear)-1, real(y_linear));
title('Linear Convolution using FFT (Zero-Padded)');
xlabel('n');
ylabel('y[n]');
grid on;
```

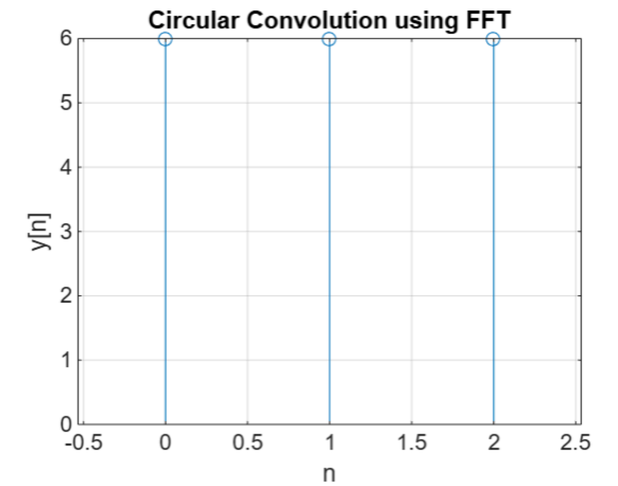
Output:

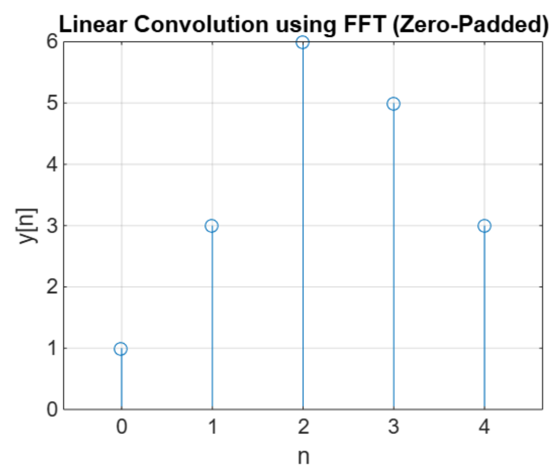
```
y =

     6     6     6


y =

    1.0000    3.0000    6.0000    5.0000    3.0000
```



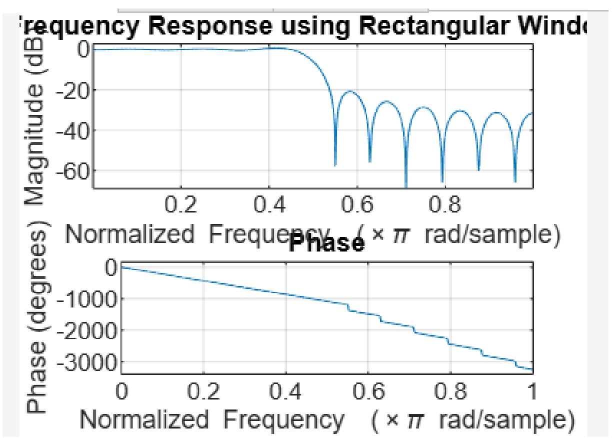


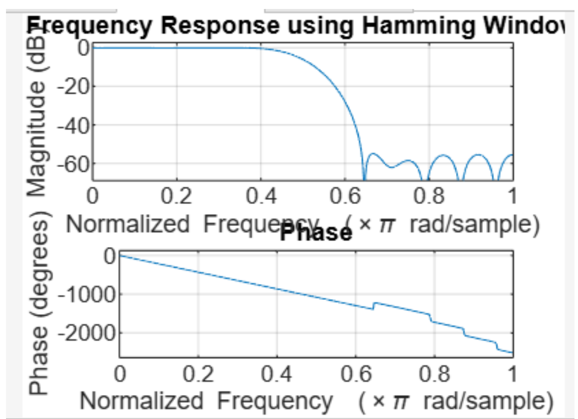
EXP 5: Design of FIR low pass, band pass, filter using different window method.

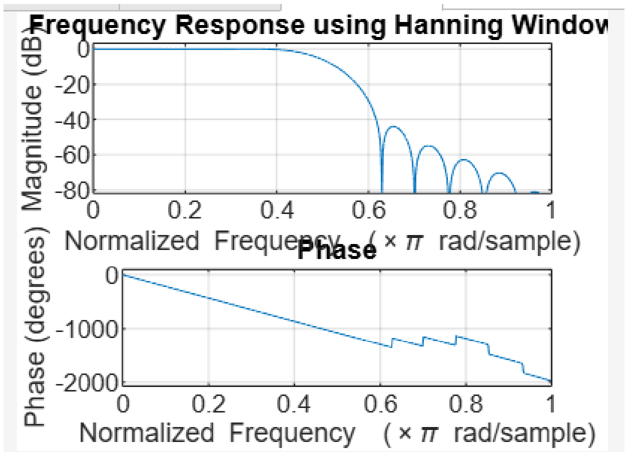
```
wc = 0.5*pi;
N = 25;
a = (N-1)/2;
eps = 0.01;
% Low pass filter
n = 0:1:N-1;
hd = (sin(wc*(n - a + eps))) ./ (pi*(n - a + eps));
% Rectangular Window
```

```
W_rect = boxcar(N);
Wt_rect = transpose(W_rect);
h_rect = hd .* Wt_rect;
figure;
freqz(h_rect);
title('Frequency Response using Rectangular Window');
% Hamming Window
W_ham = hamming(N);
Wt_ham = transpose(W_ham);
h_ham = hd .* Wt_ham;
figure;
freqz(h_ham);
title('Frequency Response using Hamming Window');
% Hanning Window
W_han = hanning(N);
Wt_han = transpose(W_han);
h_han = hd .* Wt_han;
figure;
freqz(h_han);
title('Frequency Response using Hanning Window');
```

Output:

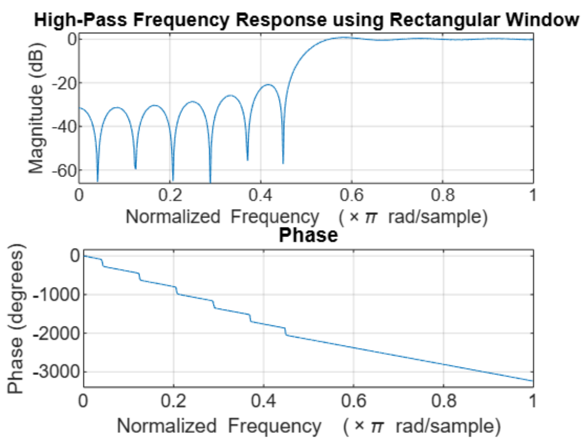


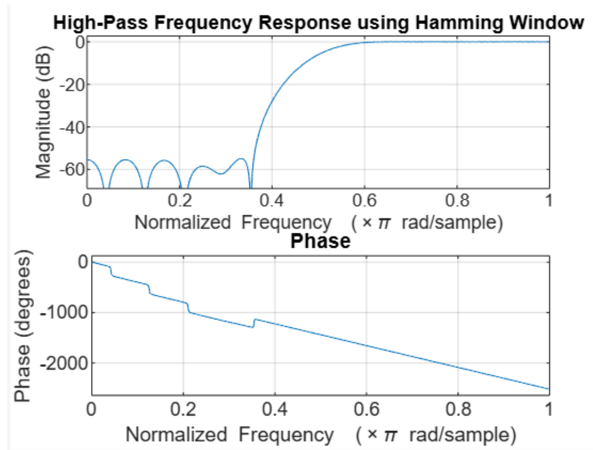




EXP 6: Design of FIR high pass, band stop filter using different window method.

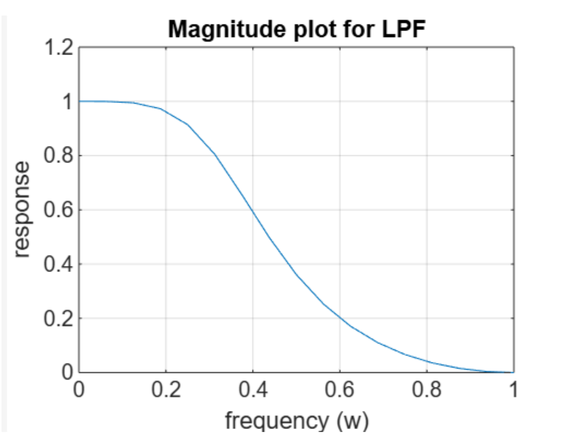
```
wc = 0.5*pi;        % Cut-off frequency
N = 25;             % Filter length
a = (N - 1) / 2;    % Center index
eps = 0.01;         % Small epsilon to avoid division by 0
% High-pass filter impulse response
n = 0:1:N-1;
hd = (sin(pi*(n - a + eps)) - sin(wc*(n - a + eps))) ./ (pi*(n - a + eps));
% Rectangular Window
W_rect = boxcar(N);
h_rect = hd .* W_rect';
figure;
freqz(h_rect);
title('High-Pass Frequency Response using Rectangular Window');
% Hamming Window
W_ham = hamming(N);
h_ham = hd .* W_ham';
figure;
freqz(h_ham);
title('High-Pass Frequency Response using Hamming Window');
% Hanning Window
W_han = hanning(N);
h_han = hd .* W_han';
figure;
freqz(h_han);
title('High-Pass Frequency Response using Hanning Window');
```





EXP7: Butterworth filter using Bilinear Transformation method for LPF

```
Ap=0.6;
As=0.1;
Wp=0.35*pi;
Ws=0.7*pi;
T=0.1;
rp=-20*log10(Ap)
rs=-20*log10(As)
wp=(2/T)*tan(Wp/2)
ws=(2/T)*tan(Ws/2)
[N,wc]=buttord(wp, ws, rp, rs, 's')
[num_analog, denom_analog]=butter(N, 1, 's')
[num1_analog, denom1_analog]=butter(N, wc, 's')
[num2_analog, denom2_analog]=bilinear(num1_analog, denom1_analog, 1/T)
w=0:pi/16:pi
H=freqz(num2_analog, denom2_analog, w)
H1=abs(H)
plot(w/pi, H1)
title("Magnitude plot for LPF")
xlabel("frequency (w)")
ylabel("response")
grid on
```



EXP 8 : Butterworth filter using Bilinear Transformation method for HPF.

```
Ap=0.6;
As=0.1;
Wp=0.35*pi;
Ws=0.7*pi;
T=0.1;
rp=-20*log10(Ap)
rs=-20*log10(As)
wp=(2/T)*tan(Wp/2)
ws=(2/T)*tan(Ws/2)
[N,wc]=buttord(wp, ws, rp, rs, 's')
[num_analog, denom_analog]=butter(N, 1, 's')
[num1_analog, denom1_analog]=butter(N, wc, 'high', 's')
[num2_analog, denom2_analog]=bilinear(num1_analog, denom1_analog, 1/T)
w=0:pi/16:pi
H=freqz(num2_analog, denom2_analog, w)
H1=abs(H)
plot(w/pi, H1)
title("Magnitude plot for HPF")
xlabel("frequency (w)")
ylabel("response")
grid on
```