**EXP\_1 basic operations on discrete signals**

**Code :**

clc;

clear all;

close all;

Xn = input('enter input sequence');

N = length(Xn);

n = 0:N-1;

subplot(3,2,1);

stem(n,Xn);

title('Input Sequence');

xlabel('n');

ylabel('Xn');

d = input('enter delay value');

nd = 0:N-1+d

Xd = [zeros(1,d),Xn]

subplot(3,2,2)

stem(nd,Xd)

title('delay value');

xlabel('nd');

ylabel('Xd');

a = input('enter advance value');

na = -a:N-1

Xa = [Xn,zeros(1,a)]

subplot(3,2,3)

stem(na,Xa)

title('advance value');

xlabel('na');

ylabel('Xa');

s = input('enter scaling value');

Xs = [s\*Xn]

subplot(3,2,4)

stem(Xs)

title('scaling value');

xlabel('Xs');

ylabel('Xn');

c = input('enter compression value');

nc = n\*c

subplot(3,2,5)

stem(nc,Xn)

title('compression value');

xlabel('nc');

ylabel('Xn');

e = input('enter expansion value');

ne = n/e

subplot(3,2,6)

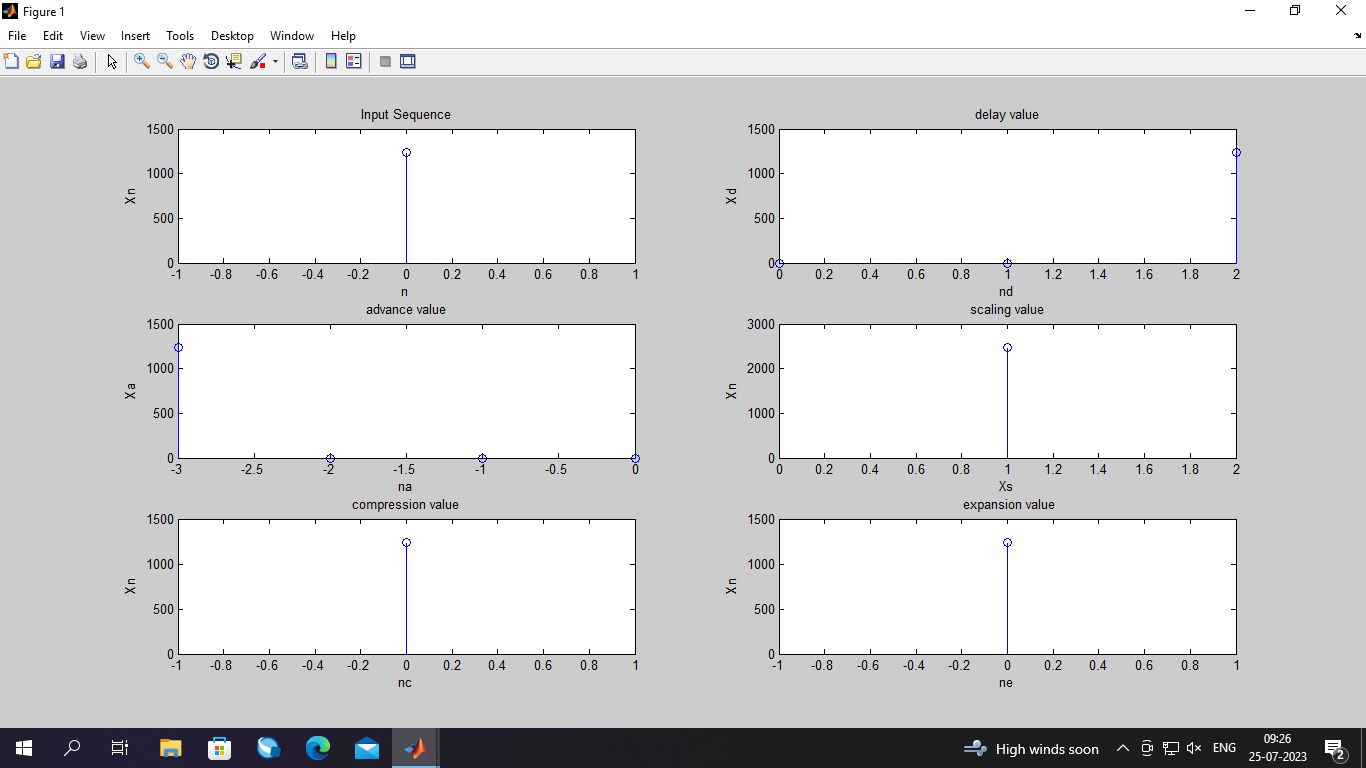
stem(ne,Xn)

title(' expansion value');

xlabel('ne');

ylabel('Xn');

**Output :**

****

**EXP\_2 linear convolution y(n) of the given input sequence x(n) and impulse response h(n) using DFT and IDFT**

**Code :**

clc;

close all;

clear all;

xn = input('Enter input sequence');%[1,2,3,4]

hn = input('Enter input sequence');%[1,2,1]

L = length (xn);

M = length (hn);

N = L+M-1;

xn1 = [xn, zeros(1,N-L)]

hn1 = [hn, zeros(1,N-M)]

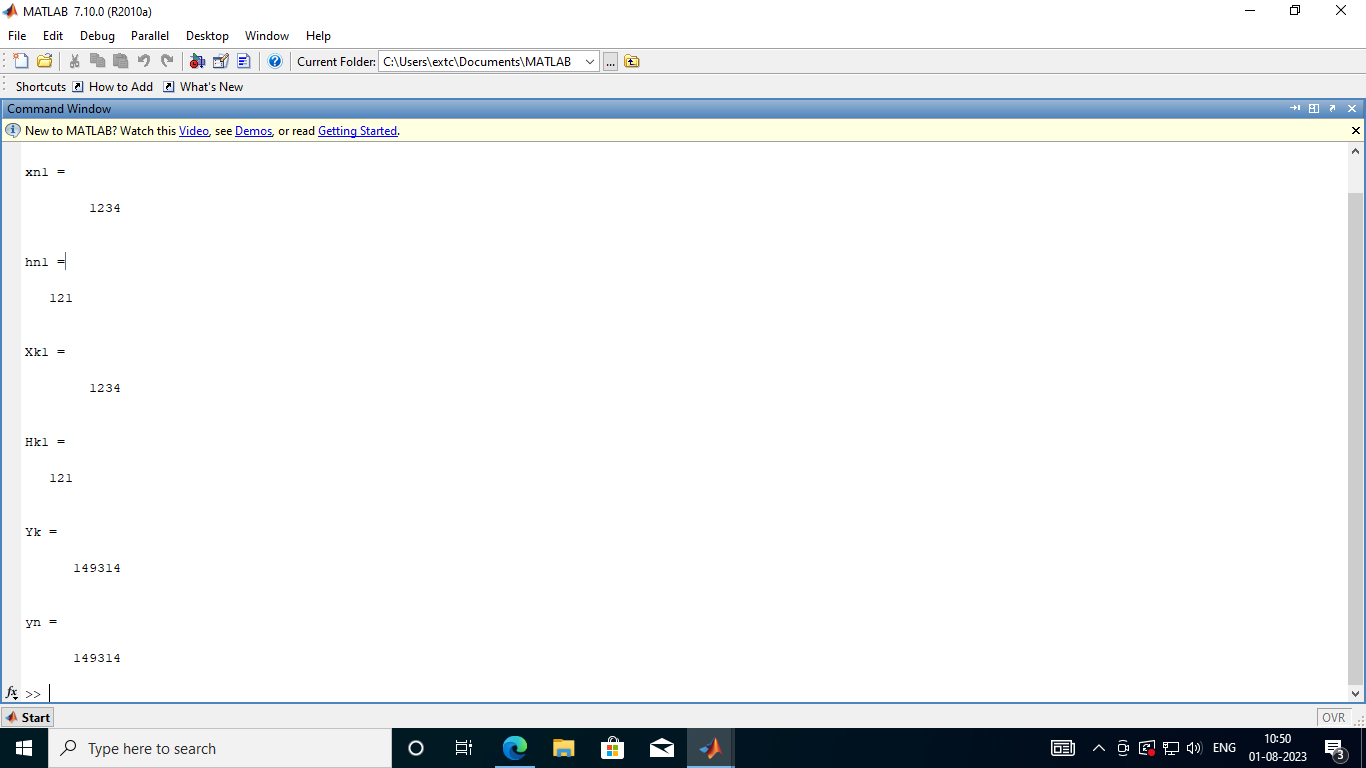
Xk1 = fft(xn1)

Hk1 = fft(hn1)

Yk = Xk1.\*Hk1

yn = (ifft(Yk))

**Output :**

****

**EXP\_3 block convolution using overlap save method**

**Code :**

clc;

clear all;

close all;

% xn=[3,-1,0,1,3,2,0,1,2,1];

% hn=[1,1,1];

xn=input('enter input xn of len 10');

hn=input('enter input hn of len 3');

L=3;

M=length(hn);

N= L+M-1 ;

x1=[zeros(1,M-1),xn(1,1:L)];

x2=[x1(4:5),xn(L+1:2\*L)];

x3=[x2(4:5),xn(2\*L+1:3\*L)];

x4=[x3(4:5),xn(3\*L+1),zeros(1,2)]

%x4=[1,2,1,0,0]

%x5=[x4(4:5),zeros(1,3)]

h= [ hn,zeros(1,M-1)];

hx=[h;circshift(h,[0 1]);circshift(h,[0 2]);circshift(h,[0 3]);circshift(h,[0 4])];

hx=hx';

y1=hx\*x1';

y2=hx\*x2';

y3=hx\*x3';

y4=hx\*x4'

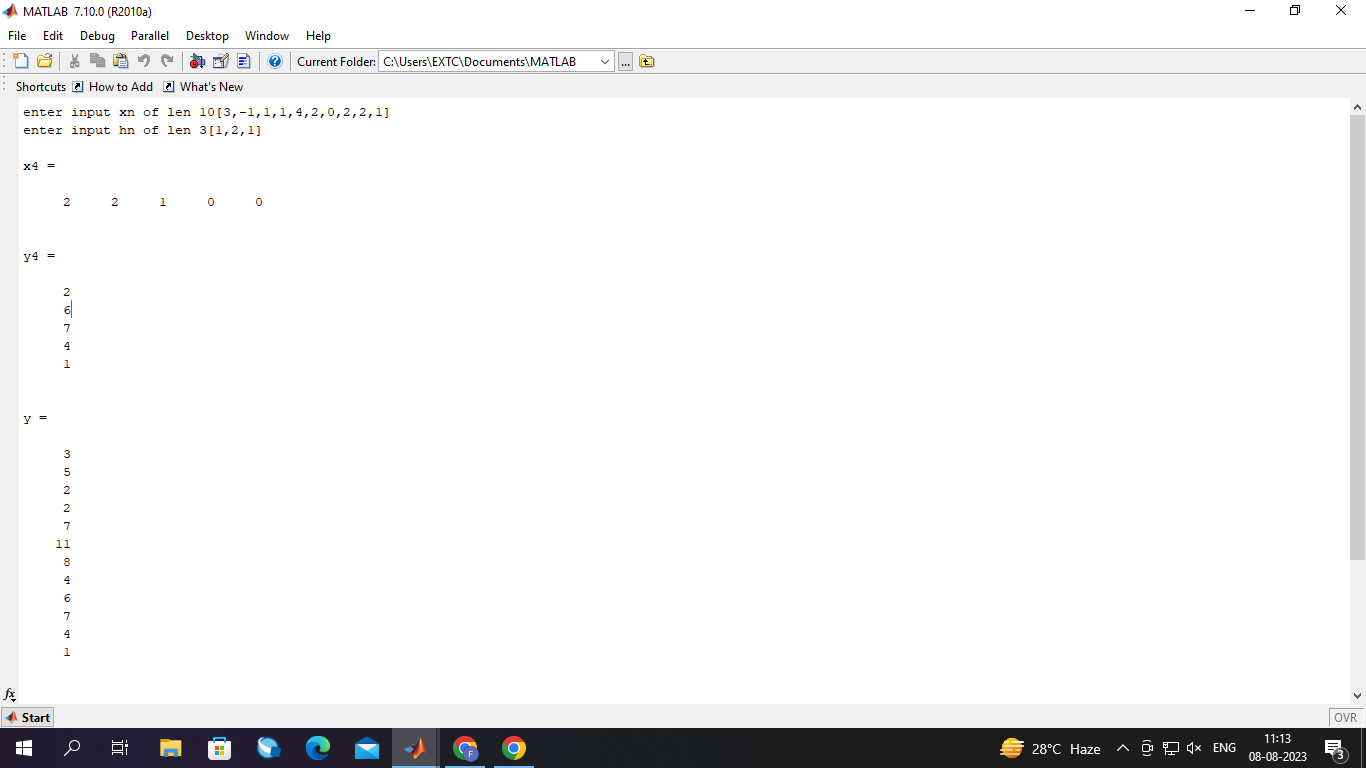
%y5=hx\*x5'

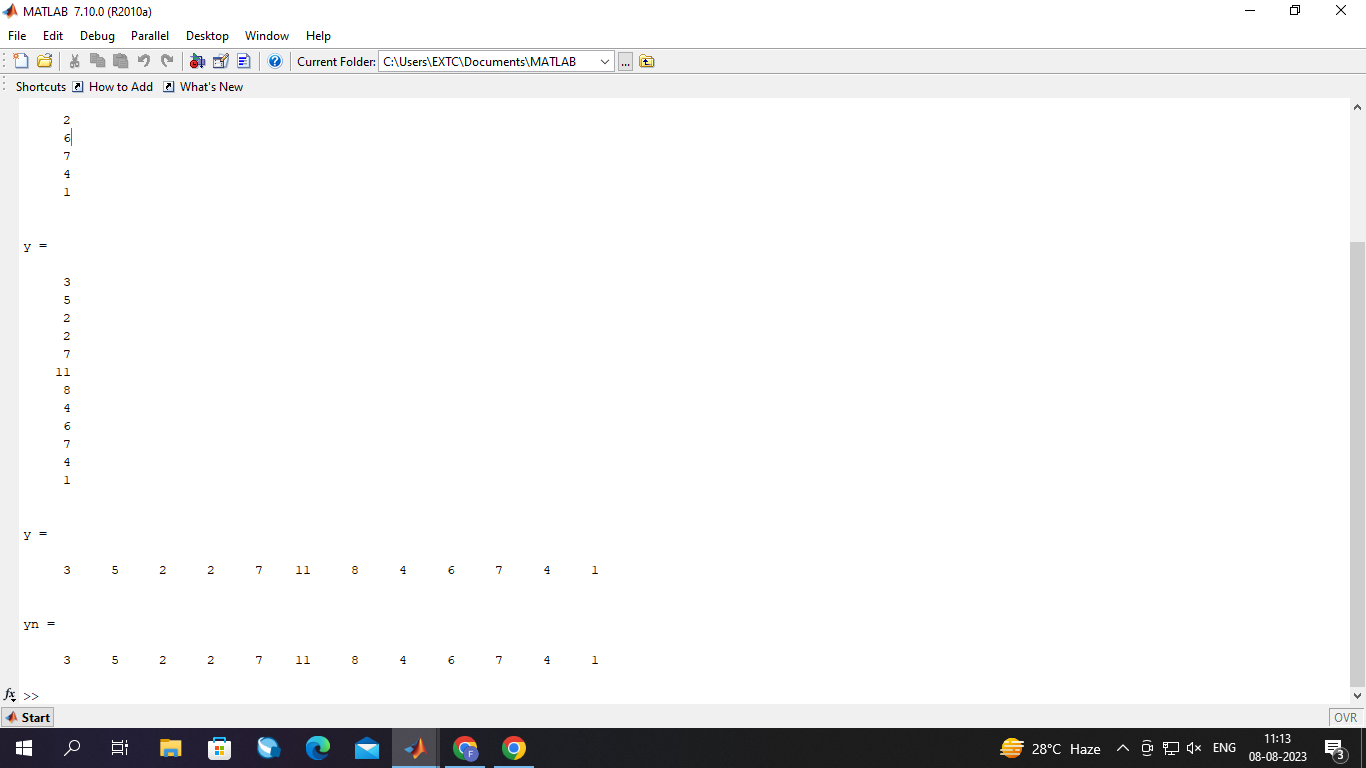
y=[y1(3:5);y2(3:5);y3(3:5);y4(3:5)]%;y5(3:5)]

y=y'

yn=conv(xn,hn)

**Output :**

****

****

**Exp\_4 linear convolution of long data sequence using overlap add method**

**Code:**

clc;

clear all;

close all;

xn=[3,-1,0,1,3,2,0,1,2,1]

hn=[1,1,1]

M=length(hn)

L=3

x1=[xn(1:L),zeros(1:M-1)]

x2=[xn(L+1:2\*L),zeros(1:M-1)]

x3=[xn(2\*L+1:3\*L),zeros(1:M-1)]

x4=[xn(3\*L+1),zeros(1,4)]

h1=[hn,zeros(1,M-1)]

hm=[h1;circshift(h1,[0 1]);circshift(h1,[0 2]);circshift(h1,[0 3]);

circshift(h1,[0 4])]

ht=hm'

y1=ht\*x1'

y2=ht\*x2'

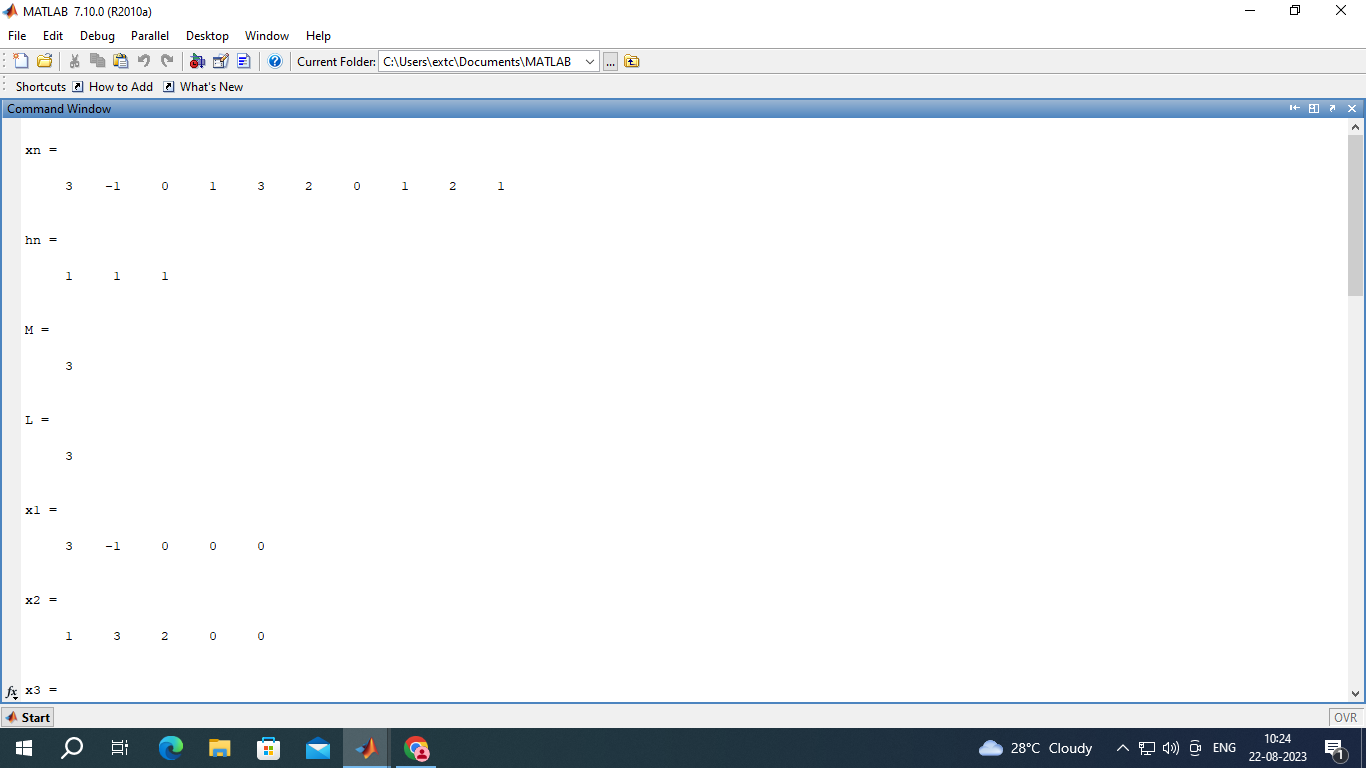
y3=ht\*x3'

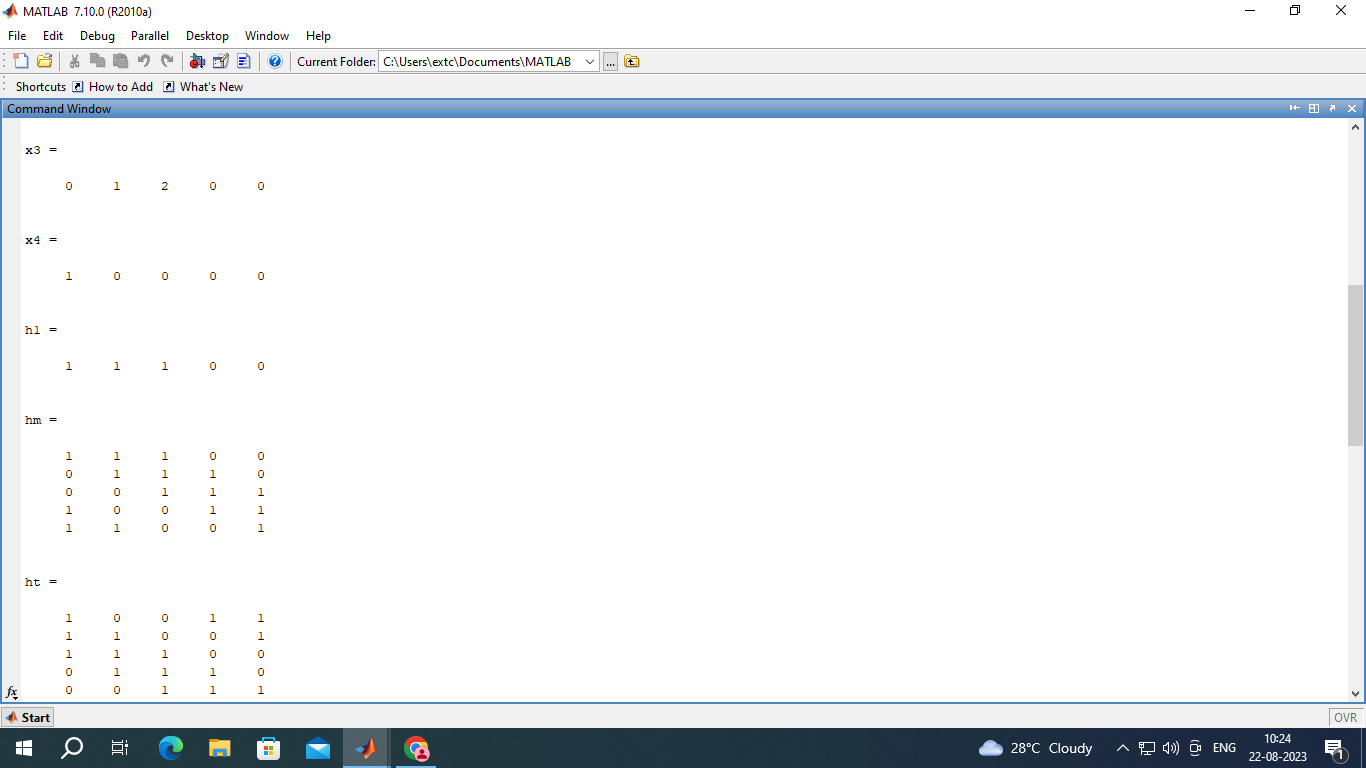
y4=ht\*x4'

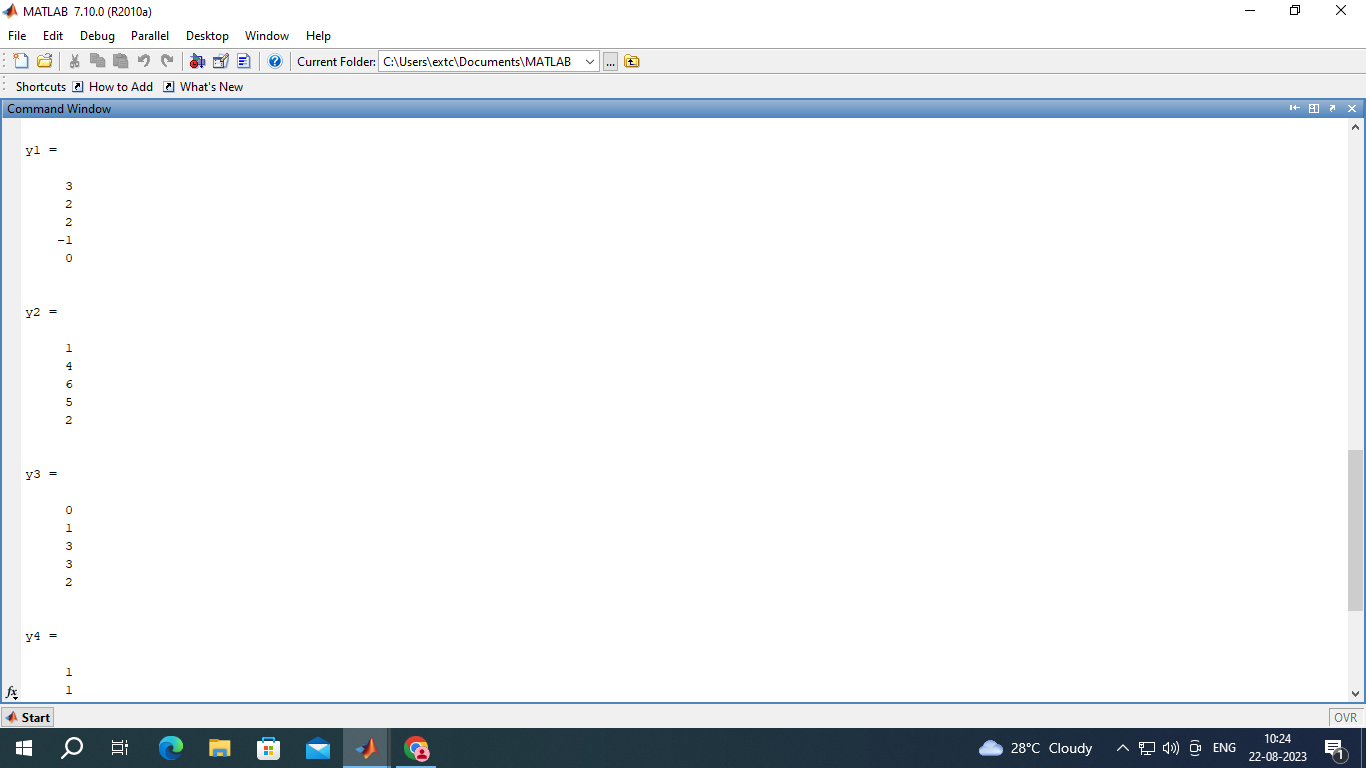
yn=[y1(1:3);y1(4:5)+y2(1:2);y2(3);y2(4:5)+y3(1:2);y3(3);y3(4:5)+y4(1:2);y4(3)]'

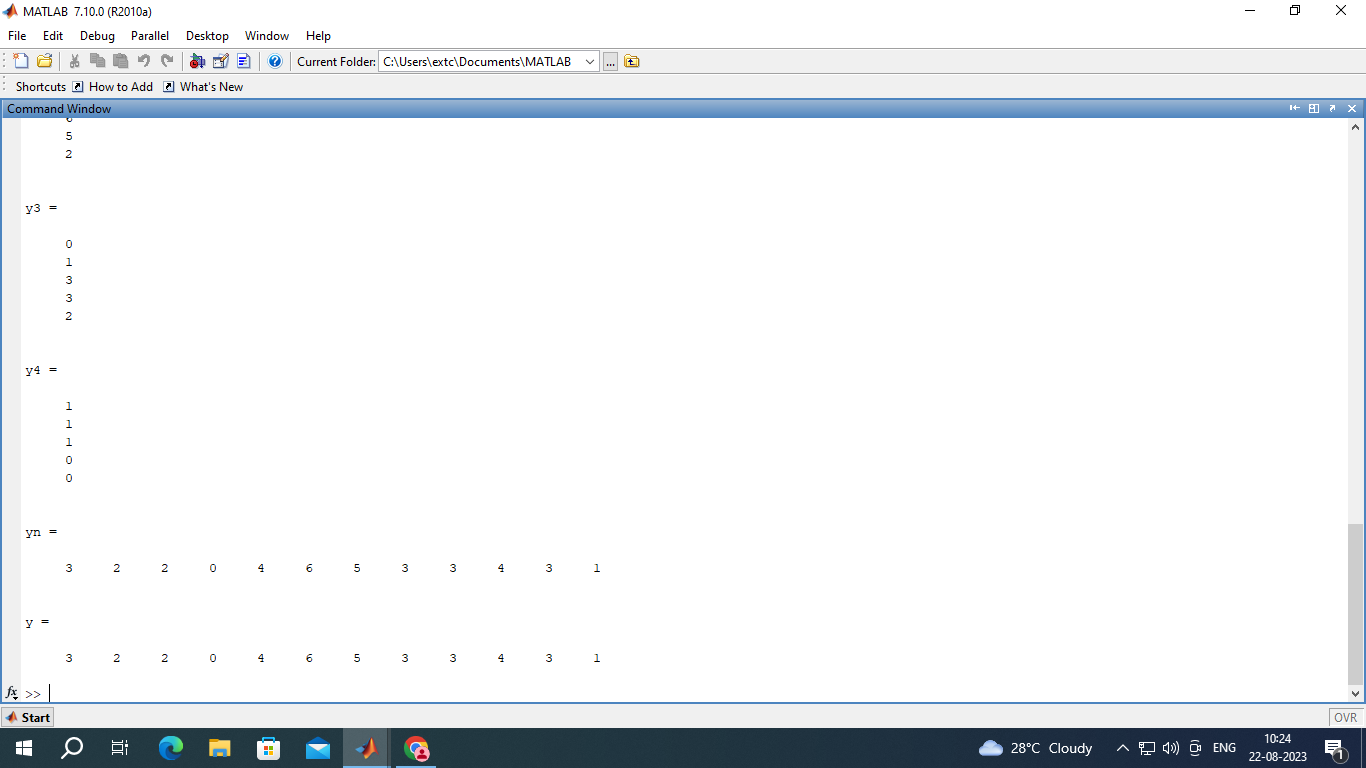
y=conv(xn,hn)

**Output :**

****

****

****

****

**Exp\_5 low pass butterworth IIR digital filter for given specification**

**Code:**

clc;

clear all;

close all;

fp = 4000;

fs = 8000;

T = 1/fsamp ;

fsamp = 24000;

rs = 0.108;

rp = 0.01;

w = 0:0.01:pi;

ws = 2\*pi\*(fs/fsamp);

wp = 2\*pi\*(fp/fsamp);

Op = 2/T\*tan\*(wp/2);

Os = 2/T\*tan\*(ws/2);

Ap = -20\*log\*(1-rp);

As = -20\*log\*(rs);

[N,Oc] = buttord (Op , Os , Ap , As , 's');

[ns ,ds] = Button(N,Oc,'s');

Hs = ts(ns,ds);

[Nz,Dz] = Bilinear(ns,ds,fsamp,Variable(z-1));

Hz = tf( nz , ds , fsamp);

[z,p,k] = tf2zp(ns,ds);

subplot(3,1,1)

zplane(z,p)

Hs = tf(ns,ds);

%digital

[nz,dz] = bilinear(ns,ds,fsamp);

Hz = tf(nz,dz,fsamp,'variable','z^-1');

%frequency response

Hw = freqz(nz,dz,w);

mag = abs(Hw);

subplot(3,1,2)

plot(mag)

title('Magnitude')

xlabel('Frequency(w)')

ylabel('Magnitude')

ph = angle(Hw);

subplot(3,1,3)

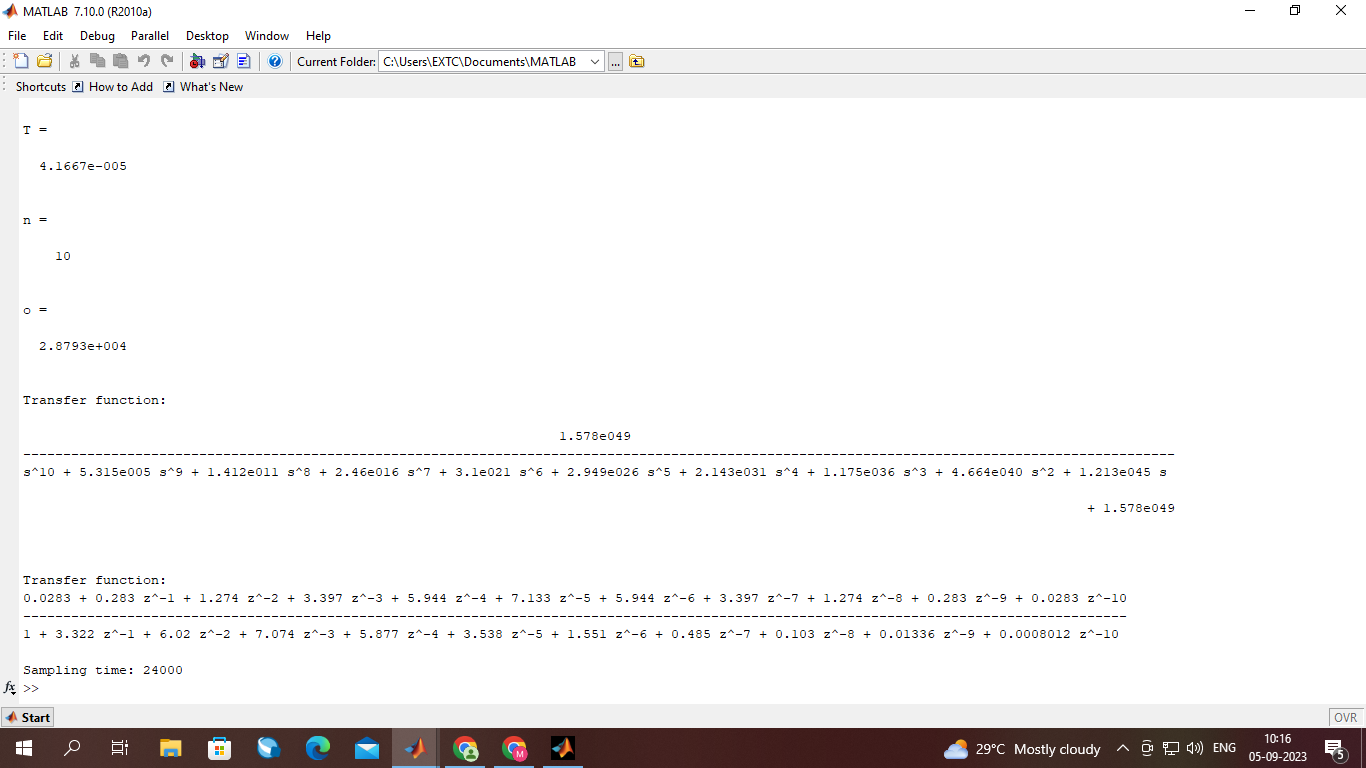
plot(ph)

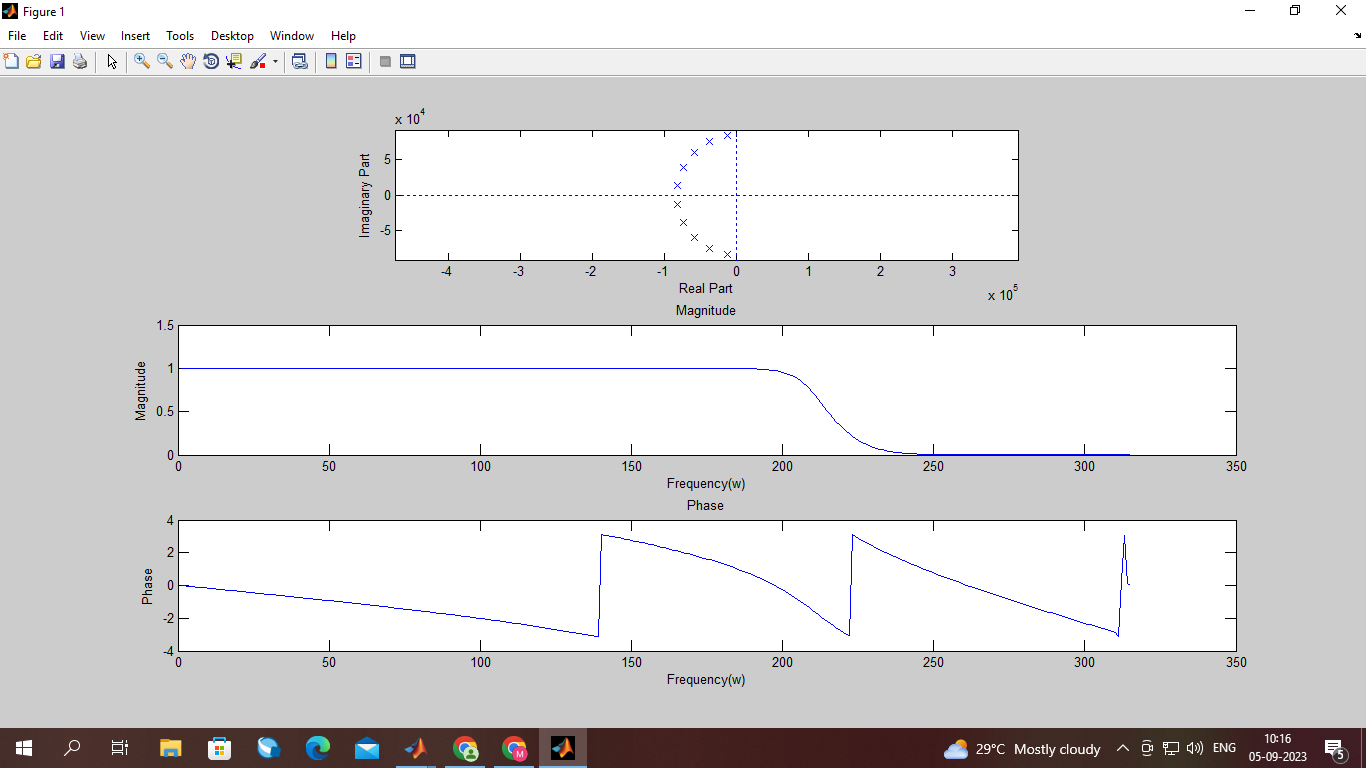
title('Phase')

xlabel('Frequency(w)')

ylabel('Phase')

**Output:**

****

****

**Exp 6 code to design and implement FIR filter using window technique.**

clc

clear all;

close all;

rp = 0.03;

rs = 0.01;

fp = 1400;

fs = 2000;

fsamp = 8000;

wp = 2\*(fp/fsamp);

ws = 2\*(fs/fsamp);

w = 0:0.01:pi;

nr = 20\*log(sqrt(rp\*rs))-13;

dr = 14.6\*((fp-fs)/fsamp);

n = nr/dr;

n = ceil(n)

N = n+1;

%Hamming Window.

y1 = hamming(N);

subplot(6,3,1); plot(y1)

title('Hamming Window.'); xlabel('N'); ylabel('Magnitude')

h = fir1(n,wp,y1);

H1 = freqz(h);

M1 = abs(H1);

subplot(6,3,2); plot(M1)

title('Magnitude Plot of Hamming Window.'); xlabel('Frequency'); ylabel('Magnitude')

P1 = angle(H1);

subplot(6,3,3); plot(P1)

title('Phase Plot of Hamming Window.'); xlabel('Frequency'); ylabel('Phase ')

%Rectwin Window.

y1 = rectwin(N);

subplot(6,3,4); plot(y1)

title('Rectwin Window.'); xlabel('N'); ylabel('Magnitude')

h = fir1(n,wp,y1);

H1 = freqz(h);

M1 = abs(H1);

subplot(6,3,5); plot(M1)

title('Magnitude Plot of Rectwin Window.'); xlabel('Frequency'); ylabel('Magnitude')

P1 = angle(H1);

subplot(6,3,6); plot(P1)

title('Phase Plot of Rectwin Window.'); xlabel('Frequency'); ylabel('Phase ')

%Hanning Window.

y1 = hanning(N);

subplot(6,3,7); plot(y1)

title('Hanning Window.'); xlabel('N'); ylabel('Magnitude')

h = fir1(n,wp,y1);

H1 = freqz(h);

M1 = abs(H1);

subplot(6,3,8); plot(M1)

title('Magnitude Plot of Hanning Window.'); xlabel('Frequency'); ylabel('Magnitude')

P1 = angle(H1);

subplot(6,3,9); plot(P1)

title('Phase Plot of Hanning Window.'); xlabel('Frequency'); ylabel('Phase ')

%Blackman Window.

y1 = blackman(N);

subplot(6,3,10); plot(y1)

title('Blackman Window.'); xlabel('N'); ylabel('Magnitude')

h = fir1(n,wp,y1);

H1 = freqz(h);

M1 = abs(H1);

subplot(6,3,11); plot(M1)

title('Magnitude Plot of Blackman Window.'); xlabel('Frequency'); ylabel('Magnitude')

P1 = angle(H1);

subplot(6,3,12); plot(P1)

title('Phase Plot of Blackman Window.'); xlabel('Frequency'); ylabel('Phase ')

%Kaiser Window.

y1 = kaiser(N);

subplot(6,3,13); plot(y1)

title('Kaiser Window.'); xlabel('N'); ylabel('Magnitude')

h = fir1(n,wp,y1);

H1 = freqz(h);

M1 = abs(H1);

subplot(6,3,14); plot(M1)

title('Magnitude Plot of Kaiser Window.'); xlabel('Frequency'); ylabel('Magnitude')

P1 = angle(H1);

subplot(6,3,15); plot(P1)

title('Phase Plot of Kaiser Window.'); xlabel('Frequency'); ylabel('Phase ')

%Bartlett Window.

y1 = bartlett(N);

subplot(6,3,16); plot(y1)

title('Bartlett Window.'); xlabel('N'); ylabel('Magnitude')

h = fir1(n,wp,y1);

H1 = freqz(h);

M1 = abs(H1);

subplot(6,3,17); plot(M1)

title('Magnitude Plot of Bartlett Window.'); xlabel('Frequency'); ylabel('Magnitude')

P1 = angle(H1);

subplot(6,3,18); plot(P1)

title('Phase Plot of Bartlett Window.'); xlabel('Frequency'); ylabel('Phase ')

Output



**Exp 7 MATLAB code to realise IIR filters using FDA (Simulink) tools.**

**1.Direct form-I, II**

**2.Cascade form**

**3.Parallel form**

**clc;**

**clear all;**

**close all;**

**%DF-I Realization.**

**n1 = input('Enter values of coefficients of Numerator of TF1 : ');**

**d1 = input('Enter values of coefficients of Denominator of TF1 : ');**

**fs = 1000;**

**tf(n1,d1,fs,'variable','z^-1')**

**h1 = dfilt.df1(n1,d1);**

**realizemdl(h1)**

**h12 = dfilt.df2(n1,d1);**

**%DF-II Realization.**

**n2 = input('Enter values of coefficients of Numerator of TF2 : ');**

**d2 = input('Enter values of coefficients of Denominator of TF2 : ');**

**fs = 1000;**

**tf(n2,d2,fs,'variable','z^-1')**

**h2 = dfilt.df2(n2,d2);**

**realizemdl(h2)**

**h22 = h2;**

**%Cascade Realization.**

**hc = dfilt.cascade(h12,h22);**

**realizemdl(hc)**

**%Parallel Realization.**

**hp = dfilt.parallel(h12,h22);**

**realizemdl(hp)**

**Exp\_8 sequence x(n), plot and observe magnitude and phase spectrum for given sequence using DFT.**

**Code:**

clc;

clear all;

close all;

x=input('enter xn');

N=length(x);

for k=0:N-1

for n=0:N-1

w(n+1,k+1)=exp((-1j\*2\*pi\*n\*k)/N);

end

end

TF=w;

xk= x\*w;

magnitude=abs(xk)

phase=angle(xk)

n=0:1:N-1

subplot(2,1,1);

stem(n,magnitude)

title('Magnitude plot xk');

xlabel('n');

ylabel('phase');

subplot(2,1,2);

stem(n,phase)

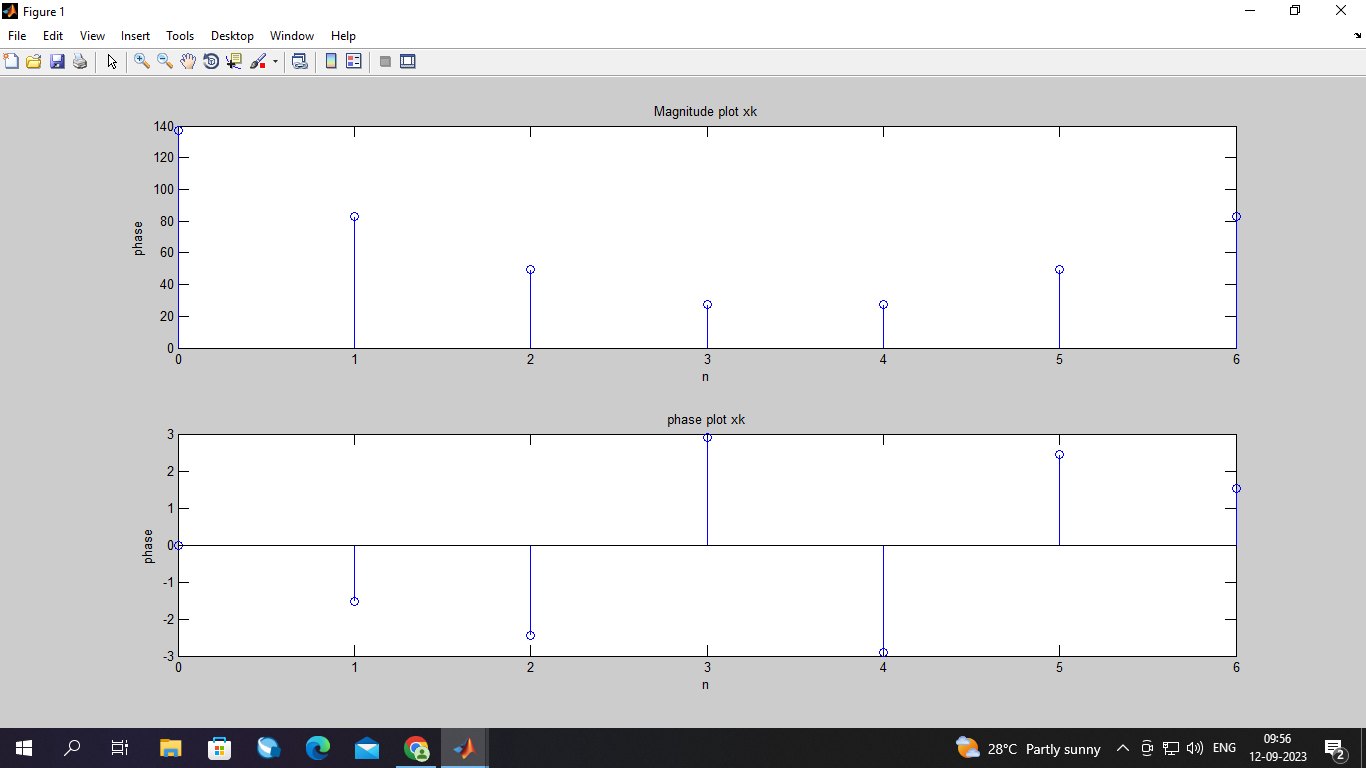
title('phase plot xk');

xlabel('n');

ylabel('phase');

**enter xn[2,56,45,15,12,4,3]**

**Output:**

****

**Exp 9 To plot and observe the following for different frequency selective filters:**

**▪ Pole Zero plot**

**▪ Magnitude Response**

**▪ Phase Response**

**clc**

**clear all;**

**close all;**

**%Digital Resonator.**

**r = 0.5;**

**w = -pi:0.01:pi;**

**h = 1/(1-r)\*(1-r\*exp(-2j\*w));**

**hmag = abs(h);**

**hpha = angle(h);**

**subplot(4,3,1); plot(w,hmag)**

**title('Magnitude Plot of Digital Resonator.'); xlabel('Frequeny.'); ylabel('Magnitude.')**

**subplot(4,3,2); plot(w,hpha)**

**title('Phase Plot of Digital Resonator.'); xlabel('Frequeny.'); ylabel('Phase.')**

**a = [1,0];**

**b = [0.5,0,-0.25];**

**[z,p,k] = tf2zp(a,b);**

**subplot(4,3,3); zplane(z,p)**

**title('Pole-Zero Plot of Digital Resonator.')**

**%Notch Filter.**

**h1 = 1-2\*exp(-1j\*w)\*cos(pi/4)+exp(-2j\*w);**

**hmag1 = abs(h1);**

**hpha1 = angle(h1);**

**subplot(4,3,4); plot(w,hmag1)**

**title('Magnitude Plot of Notch Filter.'); xlabel('Frequeny.'); ylabel('Magnitude.')**

**subplot(4,3,5); plot(w,hpha1)**

**title('Phase Plot of Notch Filter.'); xlabel('Frequeny.'); ylabel('Phase.')**

**a = [1,-2\*cos(pi/4),1];**

**b = [1,0,0];**

**[z,p,k] = tf2zp(a,b);**

**subplot(4,3,6); zplane(z,p)**

**title('Pole-Zero Plot of Notch Filter.')**

**%Comb Filter.**

**m = 6;**

**h2 = (1/m+1)\*exp(-0.5j\*w\*m).\*(sin(((m+1)/2)\*w)./sin(w/2));**

**hmag2 = abs(h2);**

**hpha2 = angle(h2);**

**subplot(4,3,7); plot(w,hmag2)**

**title('Magnitude Plot of Comb Filter.'); xlabel('Frequeny.'); ylabel('Magnitude.')**

**subplot(4,3,8); plot(w,hpha2)**

**title('Phase Plot of Comb Filter.'); xlabel('Frequeny.'); ylabel('Phase.')**

**a = [1,0,0,0,0,0,0,-1];**

**b = [1,-1,0,0,0,0,0,0];**

**[z,p,k] = tf2zp(a,b);**

**subplot(4,3,9); zplane(z,p)**

**title('Pole-Zero Plot of Comb Filter.')**

**%All Pass Filter.**

**m1 = 0.8;**

**h3 = (exp(-1j\*w)-inv(m1))./(exp(-1j\*w)-m1);**

**hmag3 = abs(h3);**

**hpha3 = angle(h3);**

**subplot(4,3,10); plot(w,hmag3)**

**title('Magnitude Plot of All Pass Filter.'); xlabel('Frequeny.'); ylabel('Magnitude.')**

**subplot(4,3,11); plot(w,hpha3)**

**title('Phase Plot of All Pass Filter.'); xlabel('Frequeny.'); ylabel('Phase.')**

**a = [1,-inv(m1)];**

**b = [1,-m1];**

**[z,p,k] = tf2zp(a,b);**

**subplot(4,3,12); zplane(z,p)**

**title('Pole-Zero Plot of All Pass Filter.')**

****

**Exp 10 dsp processor**

**1.Linear convolution**

**#include&lt;stdio.h&gt;**

**#define LENGTH1 6//length of input samples sequence**

**#define LENGTH2 4//length of impulse response coefficients**

**int x[2\*LENGTH1-1]={1,2,3,4,5,6,0,0,0,0,0};//input signal samples**

**int h[2\*LENGTH1-1]={1,2,3,4,0,0,0,0,0,0,0};//impulse response coefficients**

**int y[LENGTH1+LENGTH2-1];**

**void main()**

**{**

**int i=0,j;**

**for(i=0;i&lt;(LENGTH1+LENGTH2-1);i++)**

**{**

**y[i]=0;**

**for(j=0;j&lt;=i;j++)**

**y[i]+=x[j]\*h[i-j];**

**}**

**for(i=0;i&lt;(LENGTH1+LENGTH2-1);i++)**

**printf(&quot;%d\n&quot;,y[i]);**

**}**

****

**2. circular convolution**

**#include&lt;stdio.h&gt;**

**int m,n,x[30],h[30],y[30],i,j,temp[30],k,x2[30],a[30];**

**void main()**

**{**

**printf(&quot;Enter the length of the first sequence\n&quot;);**

**scanf(&quot;%d&quot;,&amp;m);**

**printf(&quot;Enter the length of the second sequence\n&quot;);**

**scanf(&quot;%d&quot;,&amp;n);**

**printf(&quot;Enter the first sequence\n&quot;);**

**for(i=0;i&lt;m;i++)**

**{**

**scanf(&quot;%d&quot;,&amp;x[i]);**

**}**

**printf(&quot;Enter the second sequence\n&quot;);**

**for(j=0;j&lt;n;j++)**

**{**

**scanf(&quot;%d&quot;,&amp;h[j]);**

**}**

**if((m-n)!=0)**

**{**

**if(m&gt;n)**

**{**

**for(i=n;i&lt;m;i++)**

**h[i]=0;**

**n=m;**

**}**

**for(i=m;i&lt;n;i++)**

**x[i]=0;**

**m=n;**

**}**

**y[0]=0;**

**a[0]=h[0];**

**for(j=1;j&lt;n;j++)**

**a[j]=h[n-j];**

**for(i=0;i&lt;n;i++)**

**y[0]+=x[i]\*a[i];**

**for(k=1;k&lt;n;k++)**

**{**

**y[k]=0;**

**/\*circular shift\*/**

**for(j=1;j&lt;n;j++)**

**x2[j]=a[j-1];**

**x2[0]=a[n-1];**

**for(i=0;i&lt;n;i++)**

**{**

**a[i]=x2[i];**

**y[k]+=x[i]\*x2[i];**

**}**

**}**

**/\*displaying the result\*/**

**printf(&quot;the circular convolution is\n&quot;);**

**for(i=0;i&lt;n;i++)**

**printf(&quot;%d\t&quot;,y[i]);**

**}**