*1. GENERATION OF ELEMENTARY DISCRETE TIME SEQUENCES*

x1=-10:10;

y1=[zeros(1,10),ones(1,1),zeros(1,10)];

subplot(3,2,1);

plot2d3(x1,y1);

xlabel('sequence n  ');

ylabel('x(n)');

title('UNIT IMPULSE');

x2=-20:20;

y2=[zeros(1,20),ones(1,1),ones(1,20)];

subplot(3,2,2);

plot2d3(x2,y2);

xlabel('sequence n ');

ylabel('u(n)');

title('UNIT STEP');

n=input('enter the value');

x3=0:n-1;

N=0:n-1;

subplot(3,2,3);

plot2d3(x3,N);

xlabel('sequence n ');

ylabel('Ur(n)');

title('UNIT RAMP');

a=0.8;

n=0:20;

x=a.^n;

subplot(3,2,4);

plot2d3(n,x);

xlabel('sequence n ');

ylabel('x(n)');

title('exponential signal');

a=0.8;

n=0:50;

x=a\*sin(2\*%pi\*1000\*n\*(1/20000));

subplot(3,2,6);

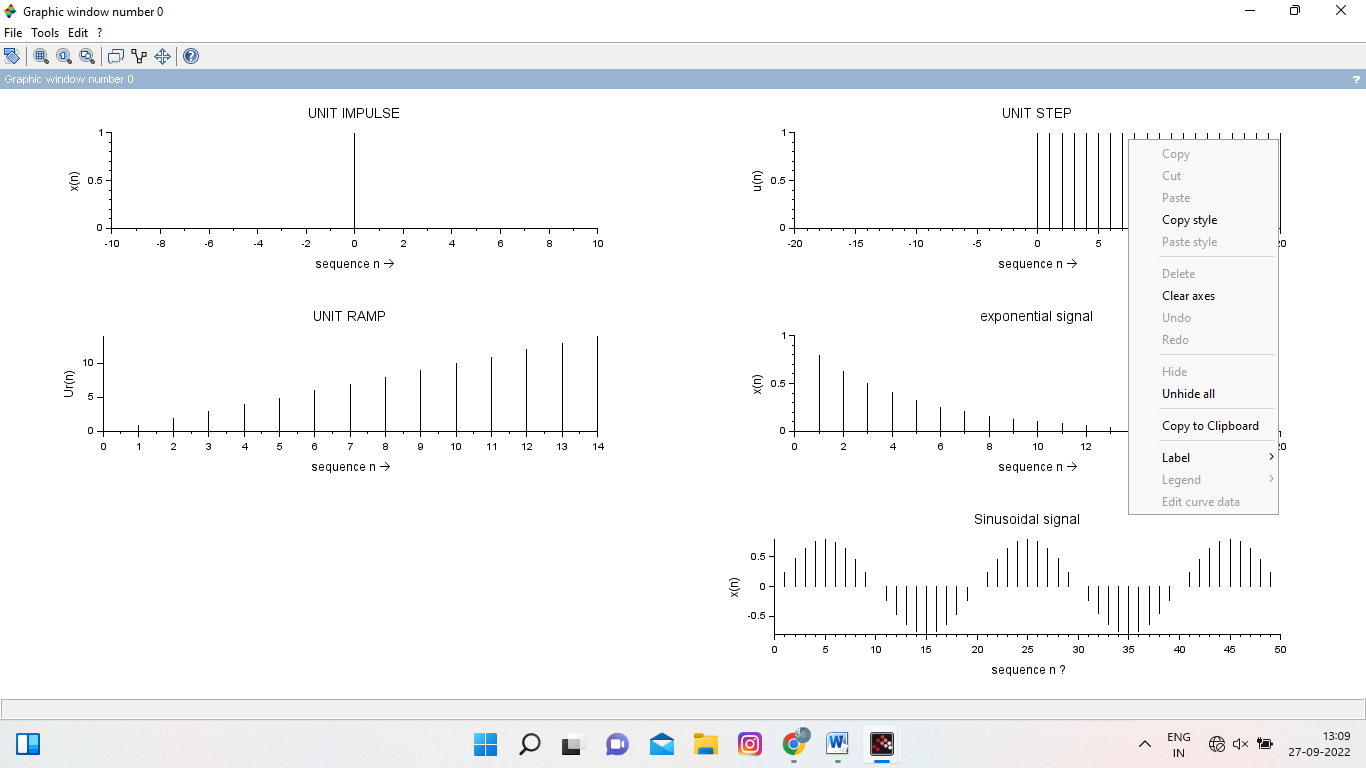
plot2d3(n,x);

xlabel('sequence n ?');

ylabel('x(n)');

title('Sinusoidal signal');

*OUTPUT:*



------------------------------------------------------------------------------------------------------------------------

2*) Linear convolution*

x=input('enter the four point input sequence :');

h=input('enter the four point impulse sequence:');

y=conv(x,h,"full");

disp(y);

subplot(3,1,1);

plot2d3(x);

xlabel('sequence n--> :');

ylabel('x(n)');

title('1.input sequence');

subplot(3,1,2);

plot2d3(h);

xlabel('sequence n--> :');

ylabel('h(n)');

title('2.impulse sequence');

subplot(3,1,3);

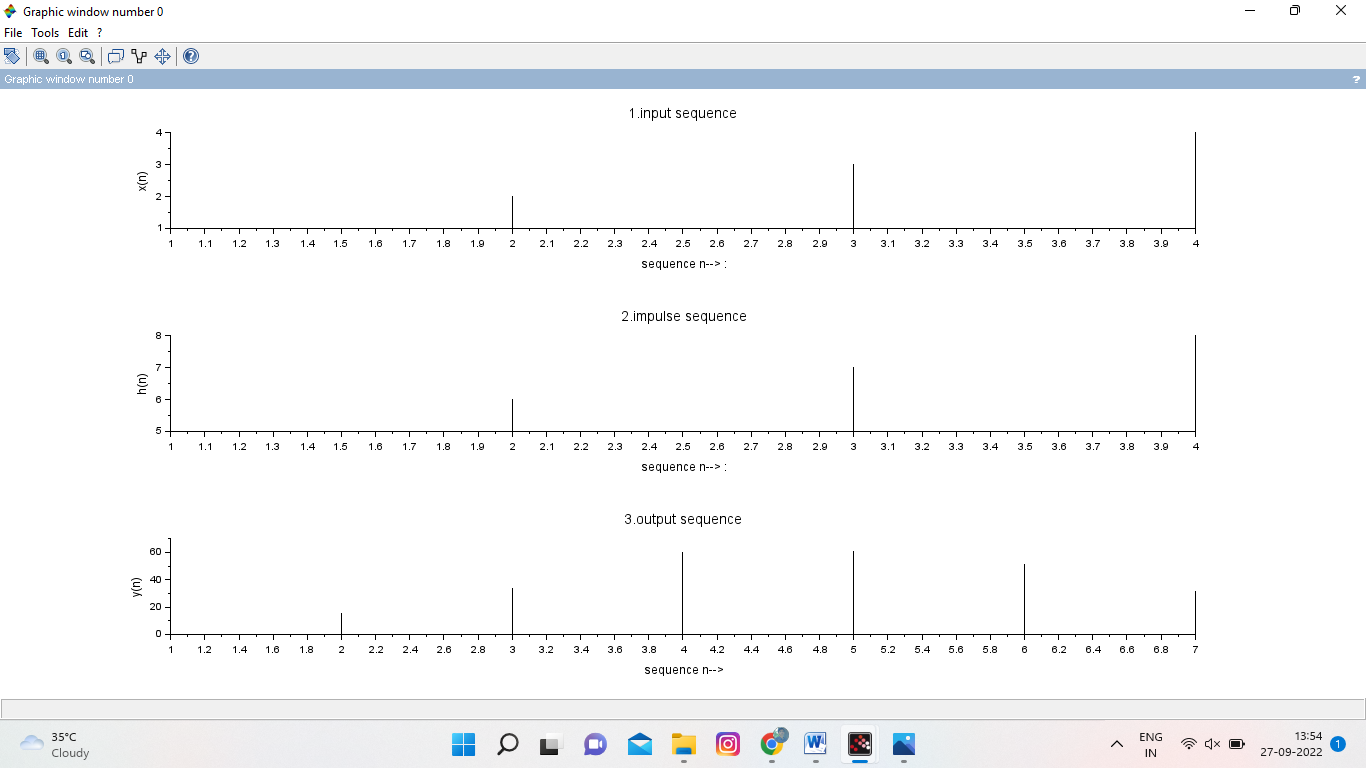
plot2d3(y);

xlabel('sequence n-->');

ylabel('y(n)');

title('3.output sequence');

OUTPUT:



-------------------------------------------------------------------------------------------------------

3)*circular covolution*

x=input('enter the four point input sequence :');

h=input('enter the four point impulse sequence:');

y=conv(x,h,"same");

disp(y);

subplot(3,1,1);

plot2d3(x);

xlabel('sequence n--> :');

ylabel('x(n)');

title('1.input sequence');

subplot(3,1,2);

plot2d3(h);

xlabel('sequence n--> :');

ylabel('h(n)');

title('2.impulse sequence');

subplot(3,1,3);

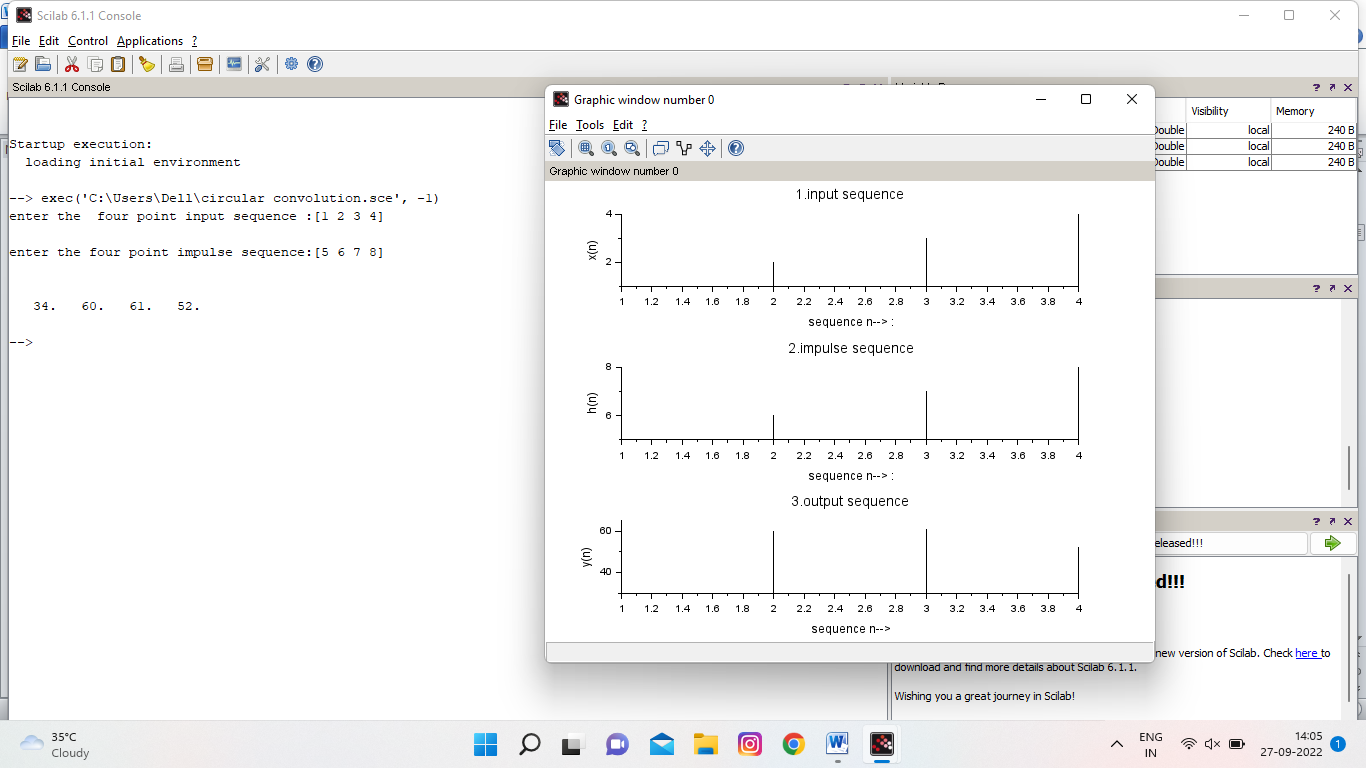
plot2d3(y);

xlabel('sequence n-->');

ylabel('y(n)');

title('3.output sequence')

OUTPUT:



4)*Auto & cross correlation*

clc;

clear;

x= input("Enter any sequence x(n)=");

figure(1);

subplot(2,1,1);

plot2d3(x);

xlabel("Time period");

ylabel("Amplitude");

title("Input sequence");

Rxx=xcorr(x)'

disp(Rxx);

subplot(2,1,2);

plot2d3(Rxx);

xlabel("Time period");

ylabel("Amplitude");

title("Auto correlation");

//cross correlation%

x=input("Enter any sequence x(n)=");

figure(2);

subplot(3,1,1);

plot2d3(x);

xlabel("Time period");

ylabel("Amplitude");

title("Input sequence");

h=input('Enter any sequence h(n)=');

subplot(3,1,2);

plot2d3(h);

xlabel("Time period");

ylabel("Amplitude");

title("Impulse sequence");

Rxy =xcorr(x,h);

disp(Rxy);

subplot(3,1,3);

plot2d3(Rxy);

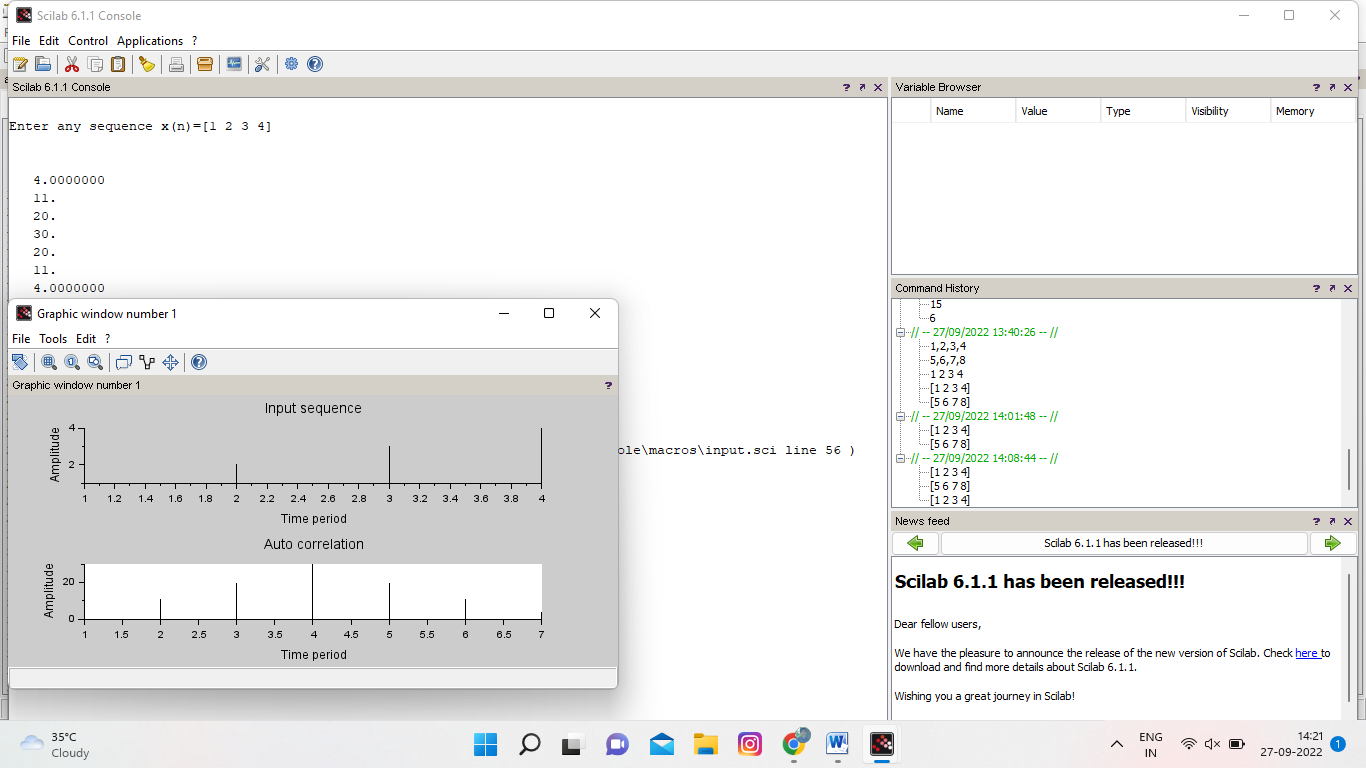
xlabel("Time period");

ylabel("Amplitude");

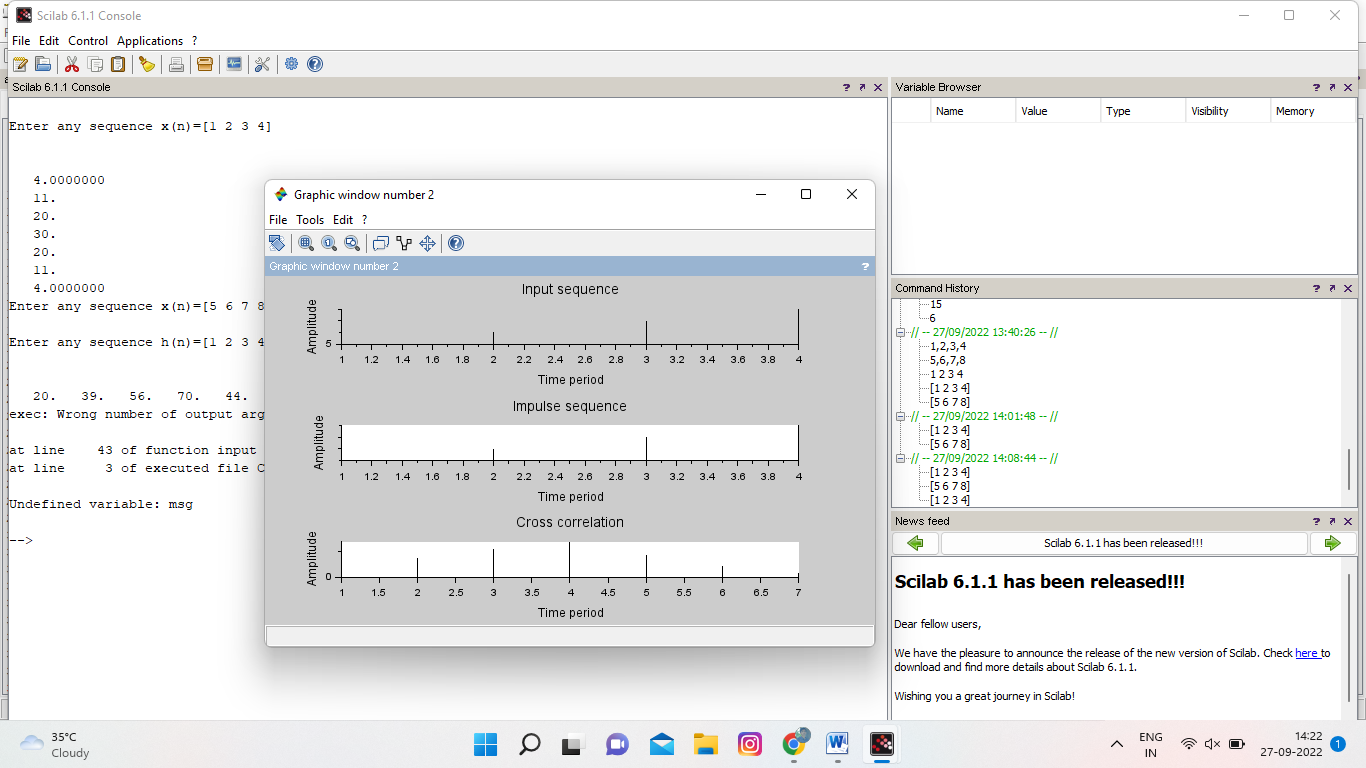
title("Cross correlation");

OUTPUT:

**AUTO CROSS CORRELATION**



**CROSS CORRELATION**



**5)frequency analysis using DFT**

clc;

xn =[1 2 3 4 4 3 2 1]

n1=0:1:length(xn)-1;

subplot(2,2,1);

plot2d3(n1,xn);

xlabel('Time n');

ylabel('Amplitude');

title('Input Sequence');

Xk = fft(xn);

disp(Xk);

K1=0:1:length(Xk)-1;

magnitude=abs(Xk)

disp(magnitude);

subplot(2,2,2);

plot2d3(K1,magnitude);

xlabel('frequency(Hz)');

ylabel('magnitude(gain)');

title('magnitude spectrum');

angle = atan(imag(Xk),real(Xk))

disp(angle)

subplot(2,2,3);

plot2d3(K1,angle);

xlabel('frequency(Hz)');

ylabel('Phase');

title('Phase spectrum')

y= ifft(Xk)

disp(y);

n2=0:1:length(y)-1;

subplot(2,2,4)

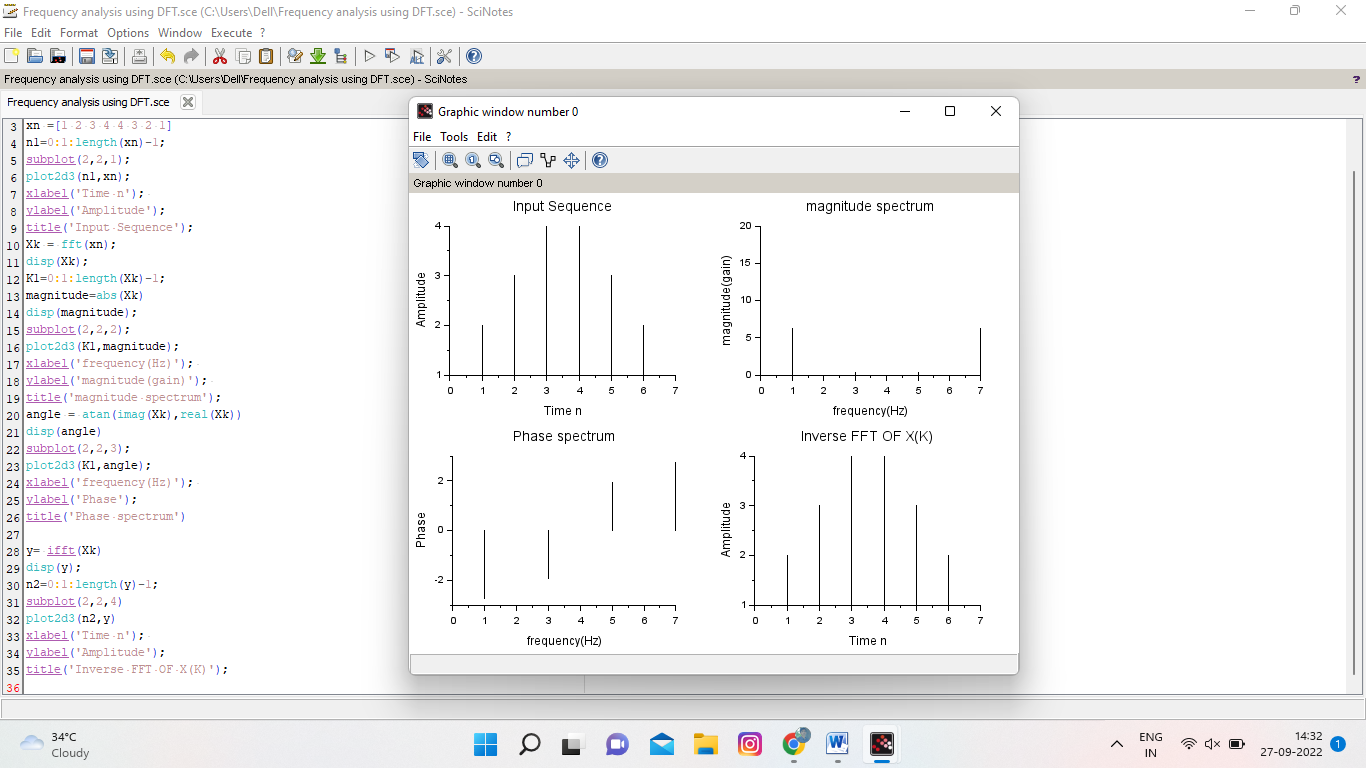
plot2d3(n2,y)

xlabel('Time n');

ylabel('Amplitude');

title('Inverse FFT OF X(K)');

OUTPUT:



**6)computation of DFT & IDFT using FFT**

clc;

close;

//Computation of DFT by using FFT

x=input('Enter the input sequence');

n=0:1:length(x)-1;

subplot(3,1,1);

plot2d3(n,x);

xlabel('Time n');

ylabel('Amplitude');

title('Input Sequence');

Xk = fft(x);

disp(Xk);

K1=0:1:length(Xk)-1;

subplot(3,1,2);

plot2d3(K1,Xk);

xlabel('freq k');

ylabel('X(K)');

title('DFT of x(n)')

//computation of IDFT from FFT%

y=fft(Xk,1);

disp(y);

subplot(3,1,3);

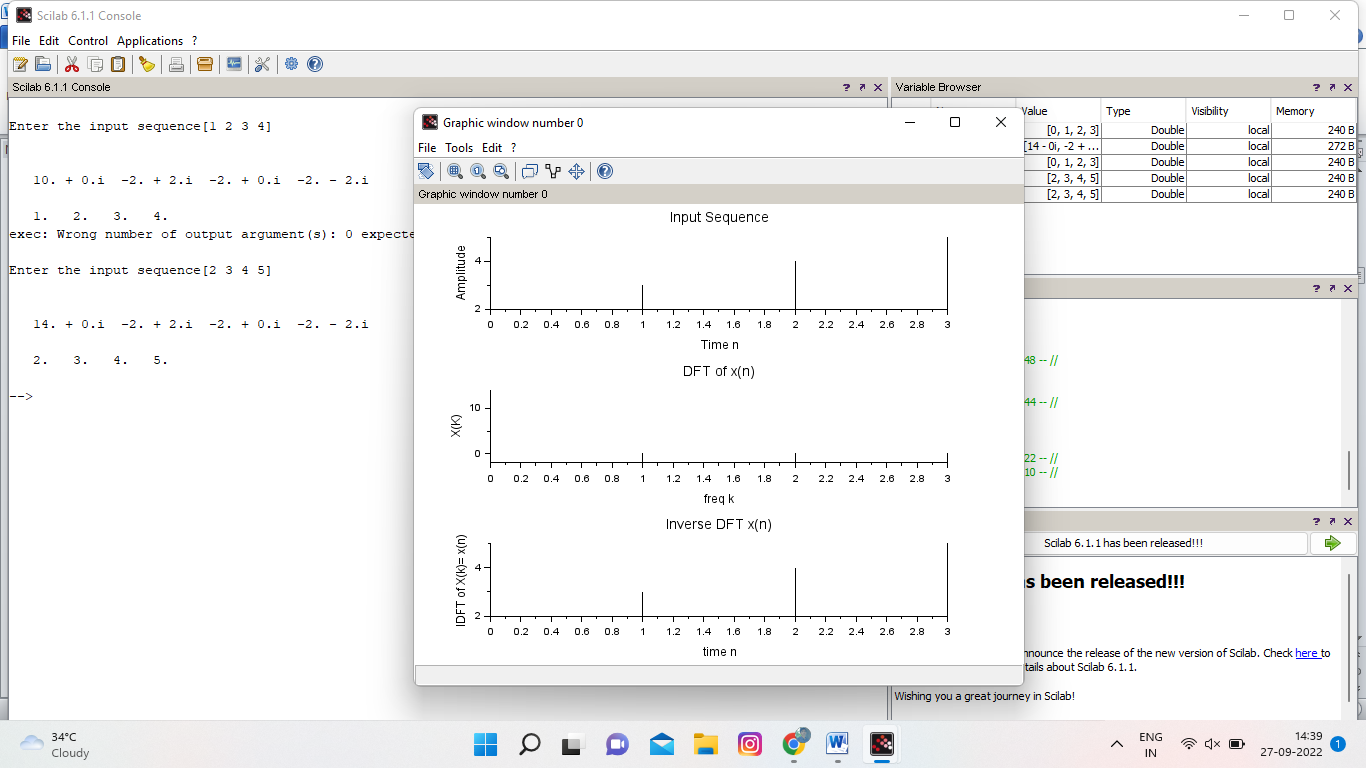
plot2d3(n,y);

xlabel('time n');

ylabel('IDFT of X(k)= x(n)');

title('Inverse DFT x(n)');

OUTPUT:



7)design of IIR butterworth filter

//Butterworth Lowpass Filter using bilinear transformation

clc ;

close ;

wp=input('Enter the pass band frequency (Radians )= ' );

ws=input('Enter the stop band frequency (Radians )= ' );

alphap=input( ' Enter the pass band attenuation (dB)=' );

alphas=input( ' Enter the stop band attenuation(dB)=' );

T=input('Enter the Value of sampling Time=');

//Pre warping- Bilinear Transformation

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

disp('omegap=',omegap);

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

disp('omegas=',omegas);

//Order of the filter

N=log10(sqrt(((10^(0.1\*alphas))-1)/((10^(0.1\*alphap))-1)))/(log10(omegas/omegap));

disp('N=',N);

N=ceil(N);

disp('Round off value of N=',N);

//Cut off frequency

omegac=omegap/(((10^(0.1\*alphap)) -1)^(1/(2\* N)));

disp('omegac=',omegac);

disp('Normalised Analog LPF Transfer function H(S)=');

hs\_Normalised = analpf(N,'butt',[0,0],1);

disp('Analog LPF Transfer function H(S)=');

disp(hs\_Normalised);

hs= analpf(N,'butt',[0,0],omegac);

disp(hs);

z=poly(0,'z');//Defining variable z

Hz=horner(hs,(2/ T)\*((z -1)/(z+1)))// Bilinear Transformation

disp('Digital LPF Transfer function H(Z)=');

disp(Hz);

HW=frmag(Hz,512); // Frequency response

w=0:%pi/511:%pi ;

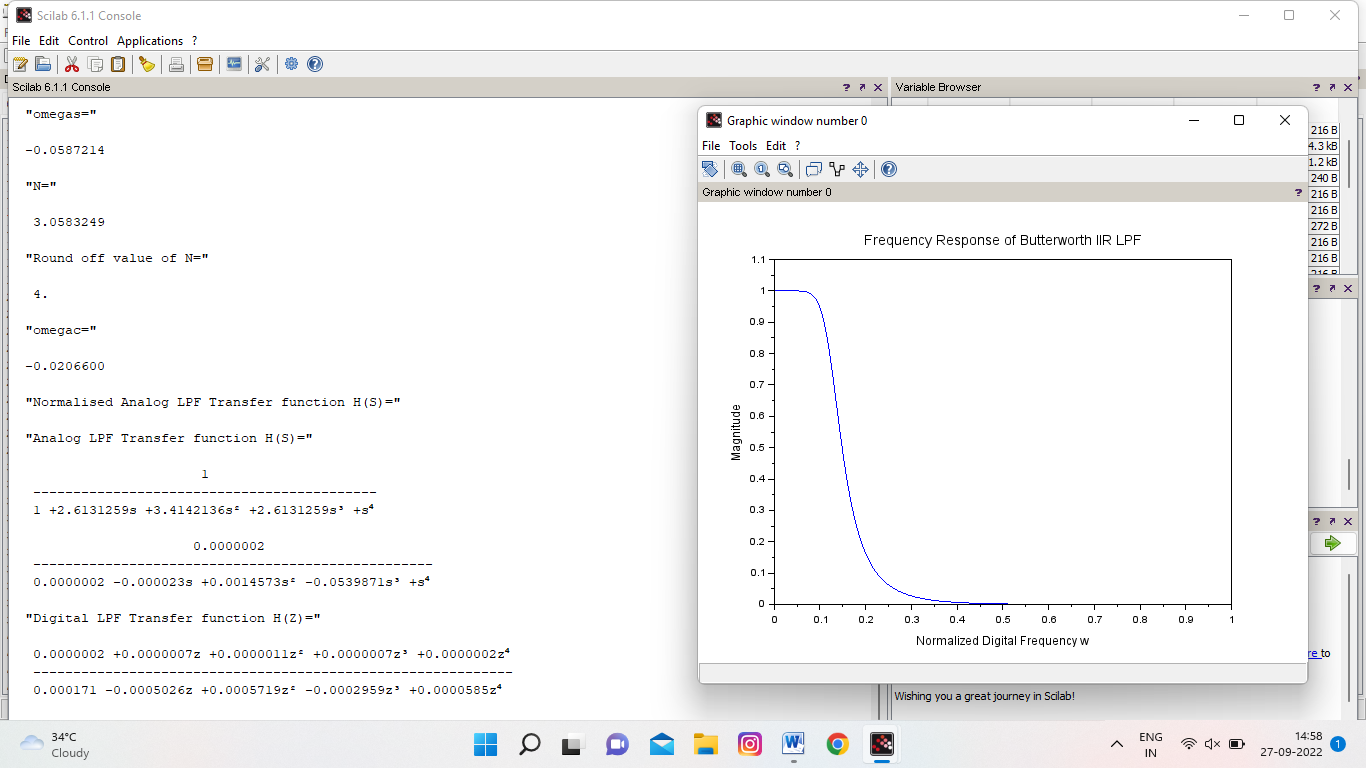
plot(w/%pi,abs(HW));

xlabel(' Normalized Digital Frequency w');

ylabel('Magnitude ');

title(' Frequency Response of Butterworth IIR LPF');

OUTPUT:



**8) design of IIR chebyshev filter**

clc ;

close ;

wp=input('Enter the pass band frequency (Radians )= ' );

ws=input('Enter the stop band frequency (Radians )= ' );

alphap=input( ' Enter the pass band attenuation (dB)=' );

alphas=input( ' Enter the stop band attenuation(dB)=' );

T=input('Enter the Value of sampling Time=');

//Pre warping- Bilinear Transformation

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

disp('omegap=',omegap);

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

disp('omegas=',omegas);

//Order of the filter

N=acosh(sqrt(((10^(0.1\*alphas))-1)/((10^(0.1\*alphap))-1)))/(acosh(omegas/omegap));

disp('N=',N);

N=ceil(N);

disp('Round off value of N=',N);

//Cut off frequency

omegac=omegap/(((10^(0.1\*alphap)) -1)^(1/(2\* N)));

disp('omegac=',omegac);

Epsilon = sqrt ((10^(0.1\*alphap))-1);

disp('Epsilon=',Epsilon);

[pols ,gn] = zpch1(N, Epsilon,omegap );

disp('Gain',gn);

disp('Poles',pols);

hs=poly(gn,'s','coeff')/real(poly(pols,'s'));

disp('Analog Low pass Chebyshev Filter Transfer function',hs);

z=poly(0,'z');//Defining variable z

Hz=horner(hs,(2/ T)\*((z -1)/(z+1)))// Bilinear Transformation

disp('Digital LPF Transfer function H(Z)=',Hz);

HW=frmag(Hz,512); // Frequency response

w=0:%pi/511:%pi ;

plot(w/%pi,abs(HW));

xlabel(' Normalized Digital Frequency w');

ylabel('Magnitude ');

title(' Frequency Response of Chebyshev IIR LPF');

OUTPUT:

