**Experiment No 1:**

**Unit Sample Signal / Impulse Signal :**

1. Start the program

2. Get the inputs for signal generation

3. Use the appropriate library function

4. Display the waveform

**Matlab code:**

n=-10:10;

x=(n==0);

stem (n,x)

xlabel('Time');

ylabel('Amplitude');

title('Unit sample signal');

**Output:**



**Experiment No 2:**

**Unit Sample Delay Signal :**

1. Start the program

2. Get the inputs for signal generation

3. Use the appropriate library function

4. Display the waveform

**Matlab code:**

n=-10:10;

x=(n-2)==0;

stem(n,x)

xlabel('Time');

ylabel('Amplitude');

title('Unit Sample Delay');

**Output:**

****

**Experiment No 3:**

**Unit Step Signal :**

1. Start the program

2. Get the inputs for signal generation

3. Use the appropriate library function

4. Display the waveform

**Matlab code:**

n=-5:5;

x=(n>=0);

stem(n,x)

xlabel('Time');

ylabel('Amplitude');

title('Unit Step Signal');

**Output:**



**Experiment No 4:**

**Unit Delay Step Signal :**

1. Start the program

2. Get the inputs for signal generation

3. Use the appropriate library function

4. Display the waveform

**Matlab code:**

n=-5:5;

x=(n-2)>=0;

stem(n,x)

xlabel('Time');

ylabel('Amplitude');

title('Unit Step Delay Signal');

**Output:**



**Experiment No 5:**

**Unit Ramp Signal :**

1. Start the program

2. Get the inputs for signal generation

3. Use the appropriate library function

4. Display the waveform

**Matlab code:**

n=-5:5;

x=n.\*(n>=0);

stem(n,x);

xlabel('Time');

ylabel('Amplitude');

title('Unit Ramp Signal');

**Output:**



**Experiment No 6:**

**Real Valued Exponential Signal :**

1. Start the program

2. Get the inputs for signal generation

3. Use the appropriate library function

4. Display the waveform

**Matlab code:**

n=-20:20;

x=(0.9.^n).\*(n>=0);

stem(n,x)

xlabel('Time');

ylabel('Amplitude');

title('Real Valued Exponential Signal');

**Output:**



**Experiment No 7:**

**Delay Real Valued Exponential Signal :**

1. Start the program

2. Get the inputs for signal generation

3. Use the appropriate library function

4. Display the waveform

**Matlab code:**

n=-10:10;

x=(0.9.^n).\*((n-3)>=0);

stem(n,x)

xlabel('Time');

ylabel('Amplitude');

title('Delay Real valued Exponential Signal');

**Output:**



**Experiment No 8:**

**Complex Valued Exponential Signal :**

1. Start the program

2. Get the inputs for signal generation

3. Use the appropriate library function

4. Display the waveform

**Matlab code:**

n=-20:20;

x=(exp((3\*4j)\*n)).\*(n>=0);

y=real(x);

subplot(2,1,1);

stem(n,y)

z=imag(x);

subplot(2,1,2);

stem(n,z)

xlabel('Time');

ylabel('Amplitude');

title('Complex Valued Exponential Signal');

**Output:**



**Experiment No 9:**

**Sinusoidal Signal :**

1. Start the program

2. Get the inputs for signal generation

3. Use the appropriate library function

4. Display the waveform

**Matlab code:**

n=-5:10;

x=4\*cos(0.1\*pi\*n+pi/3);

stem(n,x);

xlabel('Time');

ylabel('Amplitude');

title('Sinusoidal Signal');

**Output:**



**Experiment No 10:**

**Signal Addition :**

1. Start the program

2. Get the inputs for signal generation

3. Use the appropriate library function

4. Display the waveform

**Matlab code:**

n1 = 0:4;

x1 = [0 1 2 3 4];

n2 = -2:2;

x2 = [2 2 2 2 2];

n = min(min(n1),min(n2)):max(max(n1),max(n2)); % duration of y(n) %MIN Smallest component %MAX Largest component

y1 = zeros(1,length(n)); y2 = y1; % initialization %ZEROS Zeros array %LENGTH Length of vector.

y1((n>=min(n1))&(n<=max(n1))==1)=x1; % x1 with duration of y

y2((n>=min(n2))&(n<=max(n2))==1)=x2; % x2 with duration of y

y = y1+y2;% sequence addition

stem(n,y)

xlabel('Time');

ylabel('Addition');

title('Addition of the signal');

**Output:**



**Experiment No 11:**

**Shifting :**

1. Start the program

2. Get the inputs for signal generation

3. Use the appropriate library function

4. Display the waveform

**Matlab code:**

n=0:8;

x=[0 1 2 3 4 5 6 7 8];

subplot(2,1,1);stem(n,x);

title('x(n) signal');

xlabel('n');

ylabel('x(n)');

m=n+2;

y=x;

subplot(2,1,2);stem(m,y);

title('y(n)=x(n-2) signal');

xlabel('n');

ylabel('y(n)');

**Output:**



**Experiment No 12:**

**Folding:**

1. Start the program

2. Get the inputs for signal generation

3. Use the appropriate library function

4. Display the waveform

**Matlab code:**

x=[0 1 2 3 4 5 6 7 8];

subplot(2,1,1);

stem(n,x)

title('x(n)signal');

xlabel('n');

ylabel('x(n)');

m=-fliplr(n);%FLIPLR Flip matrix in left/right direction.

% FLIPLR(X) returns X with row preserved and columns flipped

% in the left/right direction.

%

% X = 1 2 3 becomes 3 2 1

% 4 5 6 6 5 4

y=fliplr(x);

subplot(2,1,2);

stem(m,y);

title('y(n)= x(-n)signal');

xlabel('n');

ylabel('y(n)');

**Output:**



**Experiment No 13:**

**Cross correlation:**

1. Start the program

2. Enter the inputs for two sequence.

3. Use the xcorr and fliplrfunction.

4. Display the waveform.

**Matlab code:**

%of the sequences x=[1,2,3,4] and h=[4,3,2,1]

clc;

clear all;

close all;

x=input('enter the 1st sequence');

h=input('enter the 2nd sequence');

y=xcorr(x,h);

figure;

subplot(3,1,1);

stem(x);

ylabel('Amplitude-->');

xlabel('(a)n-->');

subplot(3,1,2)

stem(h);

ylabel('Amplitude-->');

xlabel('(b)n-->');

subplot(3,1,3)

stem(fliplr(y));

ylabel('Amplitude');

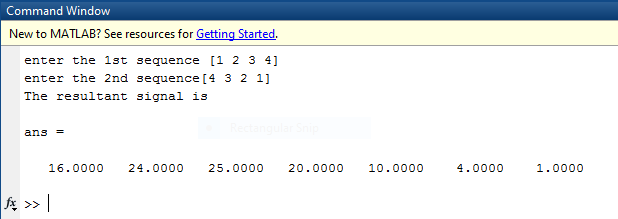
xlabel('(c)n-->');

disp('The resultant signal is');

fliplr(y)

**Output:**





**Experiment No 14:**

**Auto correlation:**

1. Start the program

2. Enter the inputs for two sequence.

3. Use the xcorr and fliplrfunction.

4.Use if else function.

5.Display the waveform.

**Matlab code:**

clc;

close all;

clear all;

% Read the signal

x=[1,2,3,6,5,4];

% define the axis

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

% plot the signal

stem(n,x);

xlabel('n');

% auto correlate the signal

Rxx=xcorr(x,x);

% the axis for auto correlation results

nRxx=-length(x)+1:length(x)-1;

% display the result

stem(nRxx,Rxx);

% properties of Rxx(0) gives the energy of the signal

% find energy of the signal

energy=sum(x.^2);

%set index of the centre value

centre\_index=ceil(length(Rxx)/2);

% Acces the centre value Rxx(0)

%Rxx\_0=Rxx(centre\_index);

Rxx\_0=Rxx(centre\_index);

% Check if the Rxx(0)=energy

if Rxx\_0==energy

disp('Rxx(0) gives energy proved');

else

disp('Rxx(0) gives energy not proved');

end

Rxx\_right=Rxx(centre\_index:1:length(Rxx));

Rxx\_left=Rxx(centre\_index:-1:1);

ifRxx\_right==Rxx\_left

disp('Rxx is even');

else

disp('Rxx is not even');

end

**Output:**



**Experiment No 15:**

**Convolution signal input output relation:**

1. Start the program

2. Enter the inputs for two sequence.

3. Use the conv and fliplrfunction.

4.Display the waveform.

**Matlab code:**

clc;

x1=input('enter the first sequence');

subplot(3,1,1);

stem(x1);

ylabel('amplitude');

title('plot of the first sequence');

x2=input('enter 2nd sequence');

subplot(3,1,2);

stem(x2);

ylabel('amplitude');

title('plot of 2nd sequence');

f=conv(x1,x2);

disp('output of linear conv is');

disp(f);

xlabel('time index n');

ylabel('amplitude f');

subplot(3,1,3);

stem(f);

title('linear conv of sequence');

**Output:**

**enter the first sequence[1 2 3 4]**

**enter 2nd sequence[4 3 2 1]**

**output of linear conv is**

**4 11 20 30 20 11 4**

****

**Experiment No 16:**

**FIR Lowpass Filter Using fir1() Function:**

1. Start the program

2. Enter the cutoff frequency.

3. Enter the sampling frequency.

4. Use fir1 and freqz functions.

4. Plot the waveform.

**Matlab code:**

clf; %Clear all figures

clear all; %Clear all variables

close all; %Close all figures

N = 512;

O = 64; %Define the order of the filter

fc = input('Enter the cutoff frequency');

fs = input('Enter the sampling frequency');

Wc=2\*fc/fs; %Normalizing the frequencies

b=fir1(O,Wc,'low'); %Calculation of filter coefficients

freqz(b,1,N,fs); %Plotting the filter response

TITLE('Magnitude and Phase Response of FIR Lowpass Filter');

**Output:**

****

**Experiment No 17:**

**FIR Highpass Filter Using fir1() Function:**

1. Start the program

2. Enter the cutoff frequency.

3. Enter the sampling frequency.

4. Use fir1 and freqz functions.

5. Plot the waveform.

**Matlab code:**

clf; %Clear all figures

clear all; %Clear all variables

close all; %Close all figures

N = 512;

O = 64; %Define the order of the filter

fc = input('Enter the cutoff frequency');

fs = input('Enter the sampling frequency');

Wc=2\*fc/fs; %Normalizing the frequencies

b=fir1(O,Wc,'high'); %Calculation of filter coefficients

freqz(b,1,N,fs); %Plotting the filter response

TITLE('Magnitude and Phase Response of FIR Highpass Filter');

**Output:**

****

**Experiment No 18:**

**FIR Bandpass Filter Using fir1() Function:**

1. Start the program

2. Enter the cutoff frequency.

3. Enter the sampling frequency.

4. Use fir1 and freqz functions.

5. Calculation of filter coefficients.

6. Plot the waveform.

**Matlab code:**

clf; %Clear all figures

clear all; %Clear all variables

close all; %Close all figures

N = 512;

O = 64; %Define the order of the filter

fn = input('Enter the passband range of frequencies');

fs = input('Enter the sampling frequency');

Wn=2\*fn/fs; %Normalizing the frequencies

b=fir1(O,Wn); %Calculation of filter coefficients

freqz(b,1,N,fs); %Plotting the filter response

TITLE('Magnitude and Phase Response of Bandpass Filter');

**Output:**

****

**Experiment No 19:**

**IIR Butterworth Lowpass Filter:**

1. Start the program

2. Enter the cutoff frequency.

3. Enter the sampling frequency.

4.Use butter function.

5.Plot the waveform.

**Matlab code:**

clf; %Clear all figures

clear all; %Clear all variables

close all; %Close all figures

N = 512;

n = 6; %Define the order of the filter

fc = input('Enter the cutoff frequency');

fs = input('Enter the sampling frequency');

Wc = 2 \* fc / fs; %Normalizing the frequencies

[b,a]=butter(n,Wc,'low'); %Calculation of filter coefficients

freqz(b,a,N,fs); %Plotting the filter response

TITLE('Magnitude and Phase response of Butterworth IIR Lowpass

filter');

**Output:**

****

**Experiment No 20:**

**IIR Chebyshev-Type-1 Lowpass Filter:**

**Matlab code:**

clf; %Clear all figures

clear all; %Clear all variables

close all; %Close all figures

N = 512;

n = 6; %Define the order of the filter

R = 0.5; %Define the Passband peak-to-peak ripple in decibels

fc = input('Enter the cutoff frequency');

fs = input('Enter the sampling frequency');

Wc = 2 \* fc / fs; %Normalizing the frequencies

[b,a]=cheby1(n,R,Wc,'low'); %Calculating filter coefficients

freqz(b,a,N,fs); %Plotting the filter response

TITLE('Magnitude and Phase Response of Chebyshev-Type-1 IIR

Lowpass Filter');

**Output:**

****

**Experiment No 21:**

**IIR Chebyshev-Type-2 Lowpass Filter:**

**Matlab code:**

clf; %Clear all figures

clear all; %Clear all variables

close all; %Close all figures

N = 512;

n = 6; %Define the order of the filter

R = 20; %Stopband ripple in decibles

fc = input('Enter the cutoff frequency');

fs = input('Enter the sampling frequency');

Wc = 2 \* fc / fs; %Normalizing the frequencies

[b,a]=cheby2(n,R,Wc,'low');%Calculation of filter coefficients

freqz(b,a,N,fs); %Plotting the filter response

TITLE('Magnitude and Phase Response of Chebyshev-Type-2 IIR

Lowpass Filter');

**Output:**

****

**Experiment No 22:**

**A PROGRAM FOR DISCRETE CONVOLUTION:**

1. Start the program

2. Enter the inputs for two sequence.

3. Use the conv and fliplrfunction.

4. Display the waveform.

**Matlab code:**

clc;

x1=input('enter the first sequence');

subplot(3,1,1);

stem(x1);

ylabel('amplitude');

title('plot of the first sequence');

x2=input('enter 2nd sequence');

subplot(3,1,2);

stem(x2);

ylabel('amplitude');

title('plot of 2nd sequence');

f=conv(x1,x2);

disp('output of linear conv is');

disp(f);

xlabel('time index n');

ylabel('amplitude f');

subplot(3,1,3);

stem(f);

title('linear conv of sequence');

**Output:**



**Experiment No 23:**

**A PROGRAM FOR DISCRETE CORRELATION:**

1. Start the program

2. Enter the inputs for two sequence.

3. Use the xcorr and fliplrfunction.

4. Display the waveform.

**Matlab code:**

%program for computing cross correlation of the sequences x=[1 2 3 4] and

%h[4 3 2 1]

clear all;

close all;

x=input('enter the 1st sequence');

h=input('enter the 2nd sequence');

y=xcorr(x,h);

figure;

subplot(3,1,1);

stem(x);

ylabel('Amplitude-->');

xlabel('(a)n-->');

subplot(3,1,2);

stem(h);

ylabel('Amplitude-->');

xlabel('(b)n-->');

subplot(3,1,3);

stem(fliplr(y));

ylabel('Amplitude-->');

xlabel('(c)n-->');

disp('The resultant signal is');

fliplr(y)

**Output:**

****

**Experiment No 24:**

**A PROGRAM FOR STABILITY TEST:**

**1.**Enter the denominator coefficient of the filter.

2.Usepoly,fliplr,abs function.

3.Use if else for stability test.

**Matlab code:**

%program for stability test

%[1 -1 .5]

clc;

clear all;

close all;

b=input('enter the denominator coefficient of the filter');

k=poly(b);

knew=fliplr(k);

s=all(abs(knew));

if(s==1)

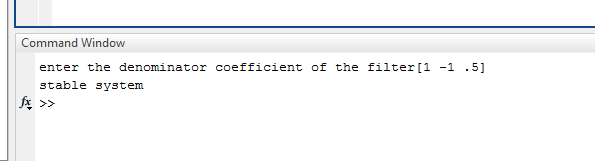
disp('stable system');

else

disp('non-stable system');

end

**Output:**



**Experiment No 25:**

**A PROGRAM FOR UNDERSTAND SAMPLING THEOREM:**

1.Start the program.

2.Use size function.

3.Display the figure.

**Matlab code:**

clc;

T=0.04; % Time period of 50 Hz signal

t=0:0.0005:0.02;

f = 1/T;

n1=0:40;

size(n1)

xa\_t=sin(2\*pi\*2\*t/T);

subplot(2,2,1);

plot(200\*t,xa\_t);

title('Verification of sampling theorem');

title('Continuous signal');

xlabel('t');

ylabel('x(t)');

ts1=0.002;%>niq rate

ts2=0.01;%=niq rate

ts3=0.1;%<niq rate

n=0:20;

x\_ts1=2\*sin(2\*pi\*n\*ts1/T);

subplot(2,2,2);

stem(n,x\_ts1);

title('greater than Nq');

xlabel('n');

ylabel('x(n)');

n=0:4;

x\_ts2=2\*sin(2\*sym('pi')\*n\*ts2/T);

subplot(2,2,3);

stem(n,x\_ts2);

title('Equal to Nq');

xlabel('n');

ylabel('x(n)');

n=0:10;

x\_ts3=2\*sin(2\*pi\*n\*ts3/T);

subplot(2,2,4);

stem(n,x\_ts3);

title('less than Nq');

xlabel('n');

ylabel('x(n)');

**Output:**



**Experiment No 26:**

**DESIGN FIR FILTERS USING WINDOWS TECHNIQUE:**

1.Start the program.

2.Enterstopband ripple, passband frequency, stopband frequency, and sampling frequency.

3.Use ceil,sqrt,fir1,freqz,abs function.

4.Display the figure.

**Matlab code:**

%Program for the design of FIR

%the passband ripple0.04

%enter the stopband ripple0.02

%enter the passband freq1500

%enter the stopband freq2000

%enter the sampling freq9000

%rectangular window

clc;

clear all;

close all;

rp=input('enter the passband ripple');

rs=input('enter the stopband ripple');

fp=input('enter the passbandfreq');

fs=input('enter the stopbandfreq');

f=input('enter the sampling freq');

wp=2\*fp/f;

ws=2\*fs/f;

num=-20\*log10(sqrt(rp\*rs))-13;

dem=14.6\*(fs-fp)/f;

n=ceil(num/dem);

n1=n+1;

if (rem(n,2)~=0)

n1=n;

n=n-1;

end

y=boxcar(n1);

%low pass filter

b=fir1(n,wp,y);

[h,o]=freqz(b,1,256);

m=20\*log10(abs(h));

subplot(2,2,1);

plot(o/pi,m);

ylabel('Gain in db-->');

xlabel('(a) Normalised frequency-->');

%high pass filter

b=fir1(n,wp,'high',y);

[h,o]=freqz(b,1,256);

m=20\*log10(abs(h));

subplot(2,2,2);

plot(o/pi,m);

ylabel('Gain in db-->');

xlabel('(b) Normalised frequency-->');

%band pass filter

wn=[wpws];

b=fir1(n,wn,y);

[h,o]=freqz(b,1,256);

m=20\*log10(abs(h));

subplot(2,2,3);

plot(o/pi,m);

ylabel('Gain in db-->');

xlabel('(c) Normalised frequency-->');

%band stop filter

b=fir1(n,wn,'stop',y);

[h,o]=freqz(b,1,256);

m=20\*log10(abs(h));

subplot(2,2,4);

plot(o/pi,m);

ylabel('Gain in db-->');

xlabel('(d) Normalised frequency-->');

**Output:**

****

**Experiment No 27:**

**A PROGRAM TO COMPARE DIRECT REALIZATION VALUES OF IIR DIGITAL FILTER:**

**Matlab code:**

%program for computing direct realization of IIR digital filter

%function y= direct(typ,b,a,x);

x=input('enter the input sequnce');

b=input('enter the numberator polynomials');

a=('enter the denominator polynomials=');

typ = input('type of realization');

p=length(a)-1;

q=length(b)-1;

pq=max(p,q);

a=a(2:p+1);

u=zeros(1,pq);

%u is the internal state

if(typ==1)

fori=1:length(x),

unew=x(i)-sum(u(1:p).\*a);

u=[unew,u];

y(i)=sum(u(1:q+1).\*b);

u= u(1:pq)

end

elseif(typ==2)

fori=1:length(x)

y(i)=b(1)\*x(i)+u(1);

u=u((2:pq),0);

u(1:q)=u(1:q)+b(2:q+1)\*x(i);

u(1:p)=u(1:p)-a\*y(i);

end

end

**Experiment No 28:**

**DEVELOP A PROGRAM FOR COMPUTING PARALLEL REALIZATION VALUES OF IIR DIGITAL FILTER :**

%program for computing direct realization of IIR digital filter

%function y= direct(typ,b,a,x);

x=input('enter the input sequnce');

b=input('enter the numberator polynomials');

a=('enter the denominator polynomials=');

typ = input('type of realization');

p=length(a)-1;

q=length(b)-1;

pq=max(p,q);

a=a(2:p+1);

u=zeros(1,pq);

%u is the internal state

if(typ==1)

fori=1:length(x),

unew=x(i)-sum(u(1:p).\*a);

u=[unew,u];

y(i)=sum(u(1:q+1).\*b);

u= u(1:pq)

end

elseif(typ==2)

fori=1:length(x)

y(i)=b(1)\*x(i)+u(1);

u=u((2:pq),0);

u(1:q)=u(1:q)+b(2:q+1)\*x(i);

u(1:p)=u(1:p)-a\*y(i);

end

end

**Experiment No 29:**

**A PROGRAM FOR COMPUTING INVERSE Z-TRANSFORM OF A RATIONAL TRANSFER FUNCTION**:

1.start the program.

2.Use impz,freqz,real,imag,abs function.

3.Display the figure.

**Matlab code:**

%Inverse Z-Transform using impz

%definition of numerator and denominator coefficients

num=[0.1+.4\*i 5 .05];

den=[1 .9+0.3\*i .12];s

%Finding the inverse z transform of G(z)

[a,b]=impz(num,den);

s%Evaluating on Unit Circle i.e. Fourier Transform

[c,d]=freqz(num,den);

% Plotting of x[n] and it's fourier transform

subplot(2,2,1)

stem(b,real(a))

title('Real Part of g[n]')

xlabel('Samples'); ylabel('Magnitude')

grid on

subplot(2,2,2)

stem(b,imag(a))

title('Imaginary Part of g[n]')

xlabel('Samples'); ylabel('Magnitude')

grid on

subplot(2,2,3)

plot(d/pi,abs(c))

title('Magnitude Spectrum of g[n]')

xlabel('\omega/\pi'); ylabel('Magnitude')

grid on

subplot(2,2,4)

plot(d/pi,angle(c))

title('Phase Spectrum of g[n]')

xlabel('\omega/\pi'); ylabel('Phase, radians')

**Output:**



**Experiment No 30:**

**Verification of time invariant system:**

1.Enter the signal parameters (amplitude, time and frequency)

2.Perform the sampling rate conversion on the input by using upsample, downsample and resample

3. Perform interpolation and decimation on the input

4. Find the spectrum of all the signals

5.Plot the waveforms.

**Matlab code:**

x1=input('enter input sequence x1:');

a=input('enter a scaling a:');

n0=input('enter shift n0:');

x2=[zeros(1,n0),x1];

y1=a.\*x1;

y2=a.\*x2;

y3=[zeros(1,n0),x1];

if(y2==y3)

display(' the system is time invariant');

else

display(' the system is time variant');

end;

subplot(2,2,1);

stem(x1);title('input signal');

subplot(2,2,2);

stem(x2);title(' signal after shifting');

subplot(2,2,3);

stem(y2);title(' L.H.S');

subplot(2,2,4);

stem(y1);title(' R.H.S');

**output:**

****

**Experiment No 31:**

**Verification of causality and non-causality:**

1.Enter the signal parameters (amplitude, time and frequency)

2.Perform the sampling rate conversion on the input by using upsample, downsample and resample

3. Perform interpolation and decimation on the input

4. Find the spectrum of all the signals

5.Plot the waveforms

**Matlab code:**

%enter input sequence x1:[1 2 3 4]

%enter lower limit n1:1

%enter higher limit n2:3

%the system is non\_causal

x1=input('enter input sequence x1:');

n1=input('enter lower limit n1:');

n2=input('enter higher limit n2:');

Flag=0;

for n=n1:n2

arg=n;

if arg>=0;

Flag=1;

end;

end;

y1=x1(n)+x1(n+1);

if(flag==1)

display(' the system is causal');

else

display(' the system is causal');

end;

subplot(2,2,1);

stem(x1);title(' Input signal');

xlabel('Time');ylabel('Amplitude');

subplot(2,2,2);

stem(y1);title('Output signal');

xlabel('Time');ylabel('Amplitude');

**Output:**



**Experiment No 32:**

**Verification of Linearity and non-Linearity:**

1.Enter the signal parameters (amplitude, time and frequency)

2.Perform the sampling rate conversion on the input by using upsample, downsample and resample

3. Perform interpolation and decimation on the input

4. Find the spectrum of all the signals

5.Plot the waveforms

**Matlab code:**

x1=input('enter first input signal x1:');

x2=input('enter second input signal x2:');

a=input('enter first scaling constant a:');

b=input('enter second scaling constant:');

subplot(2,2,1);

stem(x1);title('first input signal');

xlabel('Time');ylabel('Amplitude');

subplot(2,2,2);

stem(x2);title('Second input signal ');

xlabel('Time');ylabel('Amplitude');

y1=x1;

y2=x2;

RHS=a\*y1+a\*y2;

x3=a\*(x1+x2);

LHS=x3;

if(RHS==LHS)

display('The system is Linear.');

else

display('The system non\_Linear.');

end;

subplot(2,2,3);

stem(LHS);title(' L.H.S');

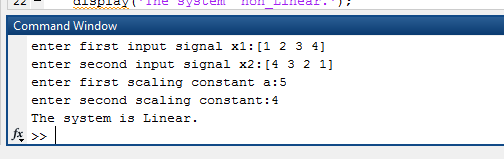
xlabel('Time');ylabel('Amplitude');

subplot(2,2,4);

stem(RHS);title(' R.H.S');

xlabel('Time');ylabel('Amplitude');

**Output:**





**Experiment No 33:**

**Verification of Stability and unstability:**

1.Enter the signal parameters (amplitude, time and frequency)

2.Perform the sampling rate conversion on the input by using upsample, downsample and resample

3. Perform interpolation and decimation on the input

4. Find the spectrum of all the signals

5.Plot the waveforms

**Matlab code:**

disp('Stability')

nr=input ('enter the numerator co\_efficient:');

dr=input ('enter the denomintorco\_efficient:');

z=tf(nr,dr,1);

[r,p,k]=residuez(nr,dr);

figure

zplane(nr,dr);

if abs(p)<1

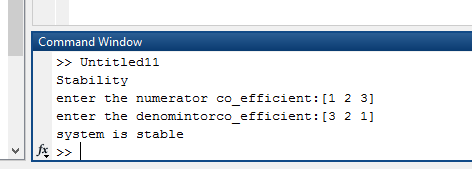
disp('system is stable');

else

disp('system is unstable');

end;

**Output:**





**Experiment No 34:**

**A Program for Discrete Time Fourier Transform:**

1.Enter the sequence .

2.Use abs,angle function.

3.Plot the waveforms

**Matlab code:**

% Input:

%Enter the sequence :[1 2 3 4]

%Mat lab code:

clc;

clear all;

close all;

a=input('Enter the sequence :');

N=length(a);

disp('The length of the sequence is:');N

for k=1:N

y(k)=0;

fori=1:N

y(k)=y(k)+a(i)\*exp((-2\*pi\*j/N)\*((i-1)\*(k-1)));

end;

end;

k=1:N

disp('The result is:');y

figure(1);

subplot(211);

stem(k,abs(y(k)));

grid;

xlabel('sample values n-->');

ylabel('Amplitudes-->');

title('Magnitude response of the DFT of given sequence');

subplot(212);

stem(angle(y(k))\*180/pi);

grid;

xlabel('sample values n-->');

ylabel('phase-->');

title('Phase response of the DFT of given sequence');

**Output:**



**Experiment No 35:**

**Butterworth high pass analog filter:**

1.Start the program.

2.Enter the stopband ripple,passband ripple,stop frequency,passband frequency,sampling frequency.

3.Use buttord,butter,zp2tf,freqs,abs,angle function.

4.Display the figure.

**Matlab code:**

clc;

close all;

clear all;

format long

rs=input('enter the stopband ripple..');

rp=input('enter the passband ripple..');

ws=input('enter the stop freq..');

wp=input('enter the passband freq..');

fs=input('enter the sampling freq..');

w1=2\*wp/fs;

w2=2\*ws/fs;

[n,wn]=buttord(w1,w2,rp,rs,'s');

%[z,p,k]=butter(n,wn);

%[b,a]=zp2tf(z,p,k);

[b,a]=butter(n,wn,'high','s');

w=0:0.01:pi;

[h,om]=freqs(b,a,w);

m=20\*log10(abs(h));

an=angle(h);

subplot(2,1,1);

plot(om/pi,m);

ylabel('GaiindB-->');

xlabel('(a)Normalized frequency-->');

subplot(2,1,2);

plot(om/pi,an);

xlabel('(b)Normalised frequnecy-->');

ylabel('Phase in radians-->');

**Output:**

****

**Experiment No 36:**

**Butterworth band pass analog filter:**

1.Start the program.

2.Enter the stopband ripple,passband ripple,stop frequency,passband frequency,sampling frequency.

3.Use buttord,butter,zp2tf,freqs,abs,angle function.

4.Display the figure

**Matlab code:**

clc;

close all;

clear all;

format long

rp=input('enter the passband ripple...');

rs=input('enter the stopband ripple...');

wp=input('enter the passband freq...');

ws=input('enter the stopband freq...');

fs=input('enter the sampling freq...');

w1=2\*wp/fs;

w2=2\*ws/fs;

[n]=buttord(w1,w2,rp,rs,'s');

wn=[w1 w2];

[b,a]=butter(n,wn,'stop','s');

w=0:0.01:pi;

[h,om]=freqz(b,a,w);

m=20\*log10(abs(h));

an=angle(h);

subplot(2,1,1);

plot(om/pi,m);

ylabel('Gain in dB-->');

xlabel('(a)Normalised frequency-->');

subplot(2,1,2);

plot(om/pi,an)

xlabel('(b)Normalised frequency-->');

ylabel('Phase in radians-->');

**Output:**

****

**Experiment No 37:**

**Chebyshev low pass filter:**

1.Start the program.

2.Enter the stopband ripple,passband ripple,stop frequency,passband frequency,sampling frequency.

3.Use buttord,butter,zp2tf,freqs,abs,angle function.

4.Display the figure.

**Matlab code:**

%enter the stopband ripple...45

%enter the passband ripple...0.2

%enter the stopband freq...1500

%enter the passband freq...1300

%enter the sampling freq...10000

%program for the design of chebyshev low pass filter

clc;

close all;

clear all;

format long

rs=input('enter the stopband ripple..');

rp=input('enter the passband ripple..');

ws=input('enter the stop freq..');

wp=input('enter the passband freq..');

fs=input('enter the sampling freq..');

w1=2\*wp/fs;

w2=2\*ws/fs;

[n,wn]=cheb1ord(w1,w2,rp,rs);

[b,a]=cheby1(n,rp,wn);

w=0:0.01:pi;

[h,om]=freqz(b,a,w);

m=20\*log10(abs(h));

an=angle(h);

subplot(2,1,1);

plot(om/pi,m);

ylabel('GaiindB-->');

xlabel('(a)Normalized frequency-->');

subplot(2,1,2);

plot(om/pi,an);

xlabel('(b)Normalised frequnecy-->');

ylabel('Phase in radians-->');

**Output:**

****

**Experiment No 38:**

**Chebyshev high pass filter :**

1.Start the program.

2.Enter the stopband ripple,passband ripple,stop frequency,passband frequency,sampling frequency.

3.Use buttord,butter,zp2tf,freqs,abs,angle function.

4.Display the figure.

**Matlab code:**

%enter the passband ripple...0.3

%enter the stopband ripple...60

%enter the passband freq.....1500

%enter the stopband freq.....2000

%enter the sampling freq......9000

%program for the design of chebyshev high pass filter

clc;

close all;

clear all;

format long

rp=input('enter the passband ripple...');

rs=input('enter the stopband ripple...');

wp=input('enter the passband freq.....');

ws=input('enter the stopband freq.....');

fs=input('ente the sampling freq......');

w1=2\*wp/fs;

w2=2\*ws/fs;

[n,wn]=cheb1ord(w1,w2,rp,rs,'s');

[b,a]=cheby1(n,rp,wn,'high','s');

w=0:0.01:pi;

[h,om]=freqs(b,a,w);

m=20\*log10(abs(h));

an=angle(h);

subplot(2,1,1);

plot(om/pi,m);

ylabel('GainindB-->');

xlabel('(a) Nomralised frequency-->');

subplot(2,1,2);

plot(om/pi,an);

xlabel('(b) Normalised frequency')

ylabel('Phase in radians')

**Output:**

****

**Experiment No 39:**

**Chebyshev band pass filter:**

1.Start the program.

2.Enter the stopband ripple,passband ripple,stop frequency,passband frequency,sampling frequency.

3.Use buttord,butter,zp2tf,freqs,abs,angle function.

4.Display the figure.

**Matlab code:**

%enter the passband ripple...0.4

%enter the stopband ripple...35

%enter the passband freq...2000

%enter the stopband freq...2500

%enter the sampling freq...10000

%program for the design of chebyshevbandpass filter

clc;

close all;

clear all;

format long

rp=input('enter the passband ripple...');

rs=input('enter the stopband ripple...');

wp=input('enter the passband freq...');

ws=input('enter the stopband freq...');

fs=input('enter the sampling freq...');

w1=2\*wp/fs;

w2=2\*ws/fs;

[n]=cheb1ord(w1,w2,rp,rs,'s');

wn=[w1 w2];

[b,a]=cheby1(n,rp,wn,'bandpass','s');

w=0:0.01:pi;

[h,om]=freqs(b,a,w);

m=20\*log10(abs(h));

an=angle(h);

subplot(2,1,1);

plot(om/pi,m);

ylabel('GainindB-->');

xlabel('(a)Normalised frequency-->');

subplot(2,1,2);

plot(om/pi,an)

xlabel('(b)Normalised frequency-->');

ylabel('Phase in radians-->');

**Output:**



**Experiment No 40:**

**Chebyshev band stop filter:**

1.Start the program.

2.Enter the stopband ripple,passband ripple,stop frequency,passband frequency,sampling frequency.

3.Use buttord,butter,zp2tf,freqs,abs,angle function.

4.Display the figure.

**Matlab code:**

%enter the stopband ripple...40

%enter the passband ripple...0.25

%enter the stopband freq...2750

%enter the passband freq...2500

%enter the sampling freq...7000

%program for the design of chebyshevbandstop filter

clc;

close all;

clear all;

format long

rs=input('enter the stopband ripple...');

rp=input('enter the passband ripple...');

ws=input('enter the stopband freq...');

wp=input('enter the passband freq...');

fs=input('enter the sampling freq...');

w1=2\*wp/fs;

w2=2\*ws/fs;

[n]=cheb1ord(w1,w2,rp,rs);

wn=[w1 w2];

[b,a]=cheby1(n,rp,wn,'stop');

w=0:0.1/pi:pi;

[h,om]=freqz(b,a,w);

m=20\*log10(abs(h));

an=angle(h);

subplot(2,1,1);

plot(om/pi,m);

ylabel('GainindB-->');

xlabel('(a)Normalised frequency-->');

subplot(2,1,2);

plot(om/pi,an)

xlabel('(b)Normalised frequency-->');

ylabel('Phase in radians-->');

**Output:**

****