Software required: Matlab

Program:

```
clc;
clear all;
close all;

num = input ('Enter numerator coefficients');
den = input ('Enter denominator coefficients');

w = 0: 0.01: 2*pi;

h = freqz (num, den, w);

subplot (2,1,1);
plot (w/pi , abs(h));
xlabel ('Frequency');
ylabel ('Magnitude');
title ('Magnitude response');

subplot (2,1,2);
plot(w/pi , angle(h));
xlabel ('Frequency');
ylabel ('Phase');
title ('Phase response');
```

I/p: numerator coeff's = [1, -1]

Denominator coeff's = [1, 1/4]

14/12/17

EXPERIMENT-2

IMPULSE RESPONSE OF FIRST AND SECOND ORDER SYSTEM

Aim: To simulate the impulse response of first and second order systems using matlab.

Software required: Matlab

Program:

```
clc;
clear all;
close all;
num = input ('Enter numerator coefficients');
den = input ('Enter denominator coefficients');
N = input ('Enter N value');

x = [zeros(1,N) 1 zeros(1,N)];
y = filter (num, den, x);

n = -N:1:N;
subplot(1,2,1); stem(n,x);
xlabel ('Time');
ylabel ('Amplitude');
title ('Input → Impulse signal');
```

```
subplot (1,2,2);
stem (n,y);
xlabel ('Time');
ylabel ('Amplitude');
title ('Impulse response ');
```

Result:

The impulse response of the first order and second order systems are verified.

14/12

14/12/17

EXPERIMENT - 32(b)

STEP RESPONSE OF FIRST AND SECOND ORDER SYSTEM

Aim: To simulate the step response of first and second order system using matlab.

Software required: Matlab

Program:

```
clear all;
close all;
clc;
num = input ('Enter numerator coeff');
den = input ('Enter denominator coeff');
N = input ('Enter N value');
x = [1 ones(1,N)];
y = filter (num, den, x);
n = 0:1:N;
subplot (1,2,1);
stem (n,x);
xlabel ('Time');
ylabel ('Amplitude');
title ('Input → Unit signal');
```

```
subplot (1, 2, 2);
stem (n, y);
xlabel ('Time');
ylabel ('Amplitude');
title ('Step response');
```

Result:

The step response of first order and second order systems are plotted.

14/12

21/12/19

EXPERIMENT - #3

DFT / IDFT OF DISCRETE TIME SIGNAL

Aim: To compute the N-point DFT / IDFT of the given discrete time signal.

Software required : Matlab.

Program:

```
clc;
clear all;
close all;
xn = input('Enter the input sequence');
N = input('Enter the length of DTFT');
L = length(xn);
if(N<L)
    error('N must be >=1');
end
x1 = [xn  zeros(1,N-1)];
for K=0:1:N-1
    for n = 0:1:N-1
        P = exp(-i*2*pi*n*K/N);
        x2(K+1,n+1) = P;
    end
end
xK = x1 * x2;
```

```
K = 0:1:N-1;
subplot(3,1,1);
stem(K, abs(xK));
xlabel('K');
ylabel('magnitude');
subplot(3,1,2);
stem(K, angle(xK));
xlabel('K');
ylabel('arg(xK))');
l = length(xK);
for K = 0:1:N-1
    for n = 0:1:N-1
        p = exp(i*2*pi*n*K/N);
        x2(K+1, n+1) = P;
    end
end
xn = (xK * x2)/N;
n = 0:1:N-1;
subplot(3,1,3);
stem(n, abs(xn));
xlabel('n');
ylabel('sequence');
```

Result:

The DFT/IDFT of a given sequence is calculated and observed.

21/12

28/12/14

EXPERIMENT-4

DETERMINATION OF POWER SPECTRUM OF A GIVEN SIGNAL (S)

Aim: To compute and plot the power spectrum of the given discrete signal.

Software required: Matlab

Program:

```
clc;
clear all;
close all;
f1 = input ('Enter first frequency');
f2 = input ('Enter second frequency');
fs = 100;
t = 0: 1/fs : 100;
x = sin(2*pi*f1*t) + sin(2*pi*f2*t);
N = input ('Enter length of DFT');
X = fft(x, N);
f = fs * (0:N-1)/N;
psd1 = (X .* conj(X))/N;
subplot (2,1,1);
plot (f, psd1);
```

```
xlabel('Frequency f');
ylabel('Power');
title('PSD through FT');

rxx = xcorr(x, x);
psd2 = abs(fft(rxx, N));
subplot(2,1,2);
plot(f, psd2);
xlabel('Frequency f');
ylabel('Power');
title('PSD through auto-correlation');
```

Result:

The power spectral density of a given signal is verified.

28/12

EXP-5

FFT OF GIVEN SEQUENCE

Aim: To compute the fourier transform of given sequence using FFT algorithm.

Software: Matlab

Program:

```
clc;
clear all;
close all;
xn = input ('Enter the sequence');
N = input ('Enter no.of pts in DFT');
L = length (xn);
if (N<L)
    disp (' N must be greater than or equal to L')
end
xK = fft(xn, N);
K = 0:N-1;
subplot (2, 2, 1);
stem (K, abs(xK));
xlabel ('K');
ylabel ('dft coefficients');
xn = ifft(xK);
n = 0:N-1;
```

```
subplot (2,2,2);
stem (n, abs(xn));
xlabel ('n');
ylabel ('sequence ~~xn~~');
```

Result: The required FFT & IFFT of given sequence are obtained.

4/1/18

11/1/18

EXPERIMENT - 6.

IMPLEMENTATION OF LP IIR FIETER OF SEQUENCE

Aim : To implement LP IIR filter of given sequence

Software : Matlab

Program: →IIRBUTTER WORTH LPF

```
clc;
clear all;
close all;
fp = input ('Enter pass band freq');
fs = input ('Enter stop band freq');
ap = input ('Enter pass band attenuation');
as = input ('Enter stop band attenuation');
sf = input ('Enter sampling freq');
wp = (2*fp)/sf;
ws = (2*fs)/sf;
[N,wn] = buttord (wp, ws, ap, as);
[b,a] = butter (N, wn);
w = input ('Enter vector freq');
[H,W] = freqz (b,a,w);
subplot (3,1,1);
plot (w/(2*pi), 20*log10(abs(H)));
n = 0 : 1/sf : 1;
x = cos(2*pi*200*n) + cos(2*pi*700*n);
subplot (3,1,2);
plot (abs (fft(x)));
y = filter (b, a, x);
subplot (3,1,3);
plot (abs (fft (y)));
```

CHEBYSHEV TYPE-1 LPF

```
clc;
clear all;
close all;
fp = input ('Enter pass band freq ');
fs = input ('Enter stop band freq');
ap = input ('Enter pass band attenuation');
as = input ('Enter stop band attenuation');
sf = input ('Enter sampling frequency');
wp = (2*fp)/sf;
ws = (2*fs)/sf;
[N,wn] = cheb1ord (wp, ws, ap, as);
R = input ('Enter ripple value');
[b,a] = cheby1 (N, R, wn);
w = input ('Enter vector frequency');
[H,W] = freqz(b,a, w);
subplot (3,1,1);
plot (w/(2*pi), 20*log10(abs(H)));
n = 0: 1/sf : 1;
x = cos (2*pi*200*n) + cos (2*pi*700*n);
subplot (3,1,2);
plot (abs(fft(x)));
y = filter (b, a, x);
subplot (3,1,3);
plot (abs(fft(y)));
```

CHEBYSHEV TYPE-2 LPF

```
clc;
clear all;
close all;
fp = input ('Enter pass band freq');
fs = input ('Enter stop band freq');
ap = input ('Enter pass band attenuation');
as = input ('Enter stop band attenuation');
sf = input ('Enter sampling freq');
wp = (2*fp)/sf;
ws = (2*fs)/sf;
[N, wn] = cheb2ord (wp, ws, ap, as);
[b,a] = cheby2 (N, as, wn);
w = input ('Enter vector frequency');
[H,W] = freqz (b, a, w);
subplot (3,1,1);
plot (w/(2*pi), 20*log10(abs(H)));
n=0:1/sf:1;
x = cos(2*pi*200*n) + cos(2*pi*400*n);
subplot (3,1,2);
plot(abs(fft(x)));
y = filter (b,a,x);
subplot(3,1,3);
plot(abs(fft(y)));
```

Result:

The LP IIR filter for a given sequence is implemented & plotted.

18/1/18

EXPERIMENT-7

IMPLEMENTATION OF HP IIR FILTER OF SEQUENCE

Aim: To Implement the HP IIR filter of given Sequence.

Software required: Matlab

Program: IIR BUTTER WORTH HPF

```
clc;
clear all;
close all;
fp = input ('Enter pass band freq');
fs = input ('Enter stop band freq');
ap: input ('Enter pass band attenuation');
as = input ('Enter stop band attenuation');
sf = input ('Enter sampling freq');
wp = (2*fp)/sf;
ws = (2*fs)/sf;
[N,wn] = buttord (wp, ws, ap, as);
[b,a] = butter (N, wn, 'high');
w = input ('Enter value of vector freq');
[H,W] = freqz (b,a,w);
subplot (3,1,1);
plot (W/(2*pi), 20*log10(abs(H)));
n = 0:1/sf:1;
x = cos(2*pi*200*n) + cos(2*pi*700*n);
subplot (3,1,2);
plot (abs(fft(x)));
y = filter (b,a,x);
subplot (3,1,3);
plot (abs(fft(y)));
```

CHEBYSHEV TYPE-1 HPF

```
clc;
clear all;
close all;
fp = input ('Enter pass band freq');
fs = input ('Enter stop band freq');
ap = input ('Enter pass band attenuation');
as = input ('Enter stop band attenuation');
Sf = input ('Enter sampling freq');
wp = (2*fp)/Sf;
ws = (2*fs)/Sf;
[N, wn] = cheb1ord (wp, ws, ap, as);
R = input ('Enter ripple value');
[b,a] = cheby1 (N, ap, wn, 'high');
w = input ('Enter vector freq');
[H,W] = freqz (b,a,w);
subplot (3,1,1);
plot (W/(2*pi), 20*log10(abs(H)));
n = 0 : 1/Sf : 1
x = cos(2*pi*200*n) + cos(2*pi*700*n);
subplot (3,1,2);
plot (abs(fft(x)));
y = filter (b,a,x);
subplot (3,1,3);
plot (abs(fft(y)));
```

CHEBYSHEV TYPE-2 HPF

```
clc;
clear all;
close all;
fp = input ('Enter pass band freq');
fs = input ('Enter stop band freq');
ap = input ('Enter pass band freq');
as = input ('Enter stop band freq');
sf = input ('Enter sampling freq');
wp = (2*fp)/sf;
ws = (2*fs)/sf;
[N,wn] = cheb2ord (wp, ws, ap, as);
[b,a] = cheby2 (N, as, wn, 'high');
w = input ('Enter vector freq');
[H,w] = freqz (b, a, w);
sub (3,1,1);
plot (w/(2*pi), 20*log10(abs(H)));
n = 0 : 1/sf : 1;
x = cos(2*pi*200*n) + cos(2*pi*700*n);
subplot (3,1,2);
plot ( abs(fft(x)));
y = filter (b, a, x);
subplot (3,1,3);
plot (abs(fft(y)));
```

Result: HPF is implemented using IIR Chebyshev (order 1, 2).

25/11/18

EXPERIMENT - 8

IMPLEMENTATION OF LP FIR FILTER OF SEQUENCE

-Aim: To implement LP FIR filter of given sequence.

Software: Matlab

Program:

```
clc;
clear all;
close all;
wc = 0.05 * ppi;
N = 25;
alpha = (N-1)/2;
eps = 0.001;
n = 0:1:N-1;
hd = sin(wc*(n-alpha+eps))./(pi*(n-alpha+eps));
wr = boxcar(N);
hn = hd.*wr';
w = 0:0.01:pi;
h = freqz(hn,1,w);
plot(w/pi, abs(h),'r');
hold on
wh = hamming(N);
wbr = black
hn = hd.*wh';
w = 0:0.01:pi;
h = freqz(hn,1,w);
plot(w/pi, abs(h),'blue');
hold on
wbl = blackman(N);
hn = hd.*wbl';
w = 0:0.01:pi;
```

```
h= freqz (hn, 1, w);
plot (w/pi , abs(h), 'K');
hold on
wb2 = bartlett(N);
hn = hd .* wb2';
w= 0: 0.01 : pi;
 h = freqz (hn, 1, w);
plot (w/pi, abs(h), 'g+');
hold on
wh1 = hanning (N);
hn = hd .* wh1';
w = 0: 0.01: pi ;
h= freqz(hn, 1, w);
plot (w/pi , abs(h), 'yellow *');
hold off
legend ('wr', 'wh', 'wb1', 'wb2', 'wh1');
xlabel ('frequency');
ylabel ('magnitude');
title ('Implementation of LP FIR');
fs=2000;
n=0:1/fs:1;
x=cos( 2* pi*200*n)+ cos(2* pi *700*n);
figure;
Subplot (2,1,1);
Plot(abs (fft(x)));
y= filter (hn,1,x);
Subplot(2,1,2);
plot (abs(fft(y)));
```

Result:

LP FIR filter is implemented & plotted.

8/10/18

EXPERIMENT - 9

IMPLEMENTATION OF HIP FIR FILTER IN SEQUENCE.

Aim: To implement HIP FIR filter of a given sequence.

Software: MATLAB

Program:

```
clc;
close all;
clear all;
wc = 0.85*pi;
N = 25;
alpha = (N-1)/2;
eps = 0.001;
n = 0:1:N-1;
hd = [sin(pi*(n-alpha+eps)) - sin(wc*(n-alpha+eps))]
                ./(pi*(n-alpha+eps));
wr = boxcar(N);
hn = hd.*wr';
w = 0:0.01:pi;
h = freqz(hn,1,w);
plot(w/pi, abs(h), 'r');
hold on
wh = hamming(N);
hn = hd.*wh';
w = 0:0.01:pi;
h = freqz(hn,1,w);
plot(w/pi, abs(h), 'blue0');
hold on
```

```
wb1 = blackman(N);
hn = hd.*wb1';
w = 0:0.01:pi;
h = freqz(hn,1,w);
plot(w/pi, abs(h), 'k');
hold on
wb2 = bartlett(N);
hn = hd.*wb2';
w = 0:0.01:pi;
h = freqz(hn,1,w);
plot(w/pi, abs(h), 'g+');
hold on
wh1 = hanning(N);
hn = hd.*wh1';
w = 0:0.01:pi;
h = freqz(hn,1,w);
plot(w/pi, abs(h), 'yellow*');
hold off
legend('wr','wh','wb1','wb2',
       'wh1');
xlabel('frequency');
ylabel('&magnitude');
title('Implementation of HP FIR');
```

```
fs = 2000;
n = 0 : 1/fs : 1;
x = cos(2*pi*200*n) + cos(2*pi*900*n);
figure;
subplot (2,1,1);
plot(abs(fft(x)));
y = filter(hn, 1, x);
subplot (2,1,2);
plot (abs (fft(y)));
```

Result:

HP FIR filter is implemented and plotted.

8/2/18

15/2/18

EXP - 10

IMPLEMENTATION OF DECIMATION PROCESS

Aim: To implement decimation process on the given sequence by factor M

Software: Matlab

Program:

```
clc;
clear all;
close all;
N = input ('Enter length of sequence');
D = input ('Enter decimation factor');
f1 = input ('Enter 1st sinusoidal freq');
f2 = input ('Enter 2nd sinusoidal freq');
n = 0:1:N-1;
x = sin((2*pi*f1*n)/1000) + sin((2*pi*f2*n)/1000);
subplot(2,1,1);
stem(n,x);
title('Input sequence');
n1 = 0:D:N-1;
x1 = sin((2*pi*f1*n1)/1000)+sin((2*pi*f2*n1)/1000);
subplot(2,1,2);
stem(n1,x1);
title('Output sequence');
```

Result: The given sequence is implemented using decimation process.

EXP - 11

IMPLEMENTATION OF INTERPOLATION PROCESS

Aim: To implement interpolation process of given sequence by factor L

Software: Matlab

Program:

```
clc;
clear all;
close all;
N = input ('Enter length of i/p signal');
I = input ('Enter interpolation factor');
f1 = input ("Enter 1st sinusoidal freq");
f2 = input ("Enter 2nd sinusoidal freq");
n = 0:1:N-1;
x = sin((2*pi*f1*n)/1000) + sin((2*pi*f2*n)/1000);
x1 = [zeros(1,I*N)];
j = 1:I:I*N;
x1(j) = x;
subplot (2,1,1);
stem (n,x);
title ('Input sequence');
n1 = 1:1:I*N;
subplot (2,1,2);
stem (n1-1, x1);
title ('Output sequence');
```

Result:

The given sequence is implemented using interpolation process.

22/2/18.

EXP-12

IMPLEMENTATION OF I/D SAMPLING RATE CONVERTE

Aim: To implement I/D sampling rate conversio

Software: Matlab

Program:

```
clc;
close all;
clear all;
N = input ('Enter length of I/P signal');
I = input ('Enter Interpolation factor');
f1 = input ('Enter 1st sine freq');
f2 = input ('Enter 2nd sine freq');
D = input ('Enter decimation factor');

n = 0:1:N-1;
x = sin((2*pi*f1*n)/1000) + sin((2*pi*f2*n)/1000);

x1 = [zeros(1,I*N)];
j = 1:I:I*N;
x1(j) = x;
subplot (3,1,1);
stem (n,x);
title ('Input sequence');
subplot (3,1,2);
n1 = 1:1:I*N;
stem (n1-1,x1);
title ('Interpolated signal');
```

```
x2 = x1(1:D:N*I);
n2 = 0:D:(N*I)-I;
subplot(3,1,3);
stem(n2,x2);
title('Output signal');
```

Result:

The output of I/D sampling rate converter is verified.

8/3/14

EXP-13 (C&Studio)

LINEAR CONVOLUTION

Aim: To find linear convolution b/w the sequences.

Program:

```
#include <stdio.h>
int a[15], h[15], y[15];
main()
{
int i, j, m, n;
printf("\n Enter m value");
scanf("%d", &m);
printf("\n Enter n value");
scanf("%d", &n);
printf("\n Enter I/P a(n)");
for (i=0; i<m; i++)
{ scanf("%d", &a[i]);
}
printf("\n Enter I/P h(n)");
for (i=0; i<n; i++)
{ scanf("%d", &h[i]);
}
for(i=m; i<=m+n-1; i++)
  a[i]=0;
for (i=n; i<=m+n-1; i++)
  h[i]=0;
```

```
for(i=0; i<m+n-1; i++)
{
    y[i]=0;
    for(j=0;j<=i; j++)
    {
        y[i]=y[i]+(x[j]*h[i-j]);
    }
}
for(i=0; i<m+n-1; i++)
  printf("\n Value of O/P y[%d]=%d",
                          i, y[i]);
}
```

Output:

Enter m value = 4

Enter n value = 4

Enter I/P x(n) = 1 2 3 4

Enter I/P h(n) = 4 3 2 1

Value of O/P y[0] = 4
y[1] = 11
y[2] = 20
y[3] = 30
y[4] = 20
y[5] = 11
y[6] = 4

3/3/18

EXP - 14

CIRCULAR CONVOLUTION

Aim: To find circular convolution b/w the sequences

Program:

```
# include <stdio.h>
int m,n, x[30], y[30], i, j, h[30], k,
    x2[30], a[30];
main()
{
  printf("\nEnter length of x sequence");
  scanf("%d", &m);
  printf("\n Enter length of h sequence");
  scanf("%d", &n);
  printf("\n Enter x sequence :");
  for(i=0; i<m; i++)
     scanf("%d", &x[i]);
  printf("\n Enter h sequence");
  for(i=0; i<n: i++)
     scanf("%d", &h[i]);
  if (m-n!=0)
  {
    if(m>n)
    {
      for(i=n; i<m; i++)
         h[i]=0;
    } n=m;
```

```
for (i = m; i<n; i++)
    x[i] = 0;
  m = n;
}
y[0] = 0;
a[0] = h[0];
for (j = 1; j<n; j++)
    a[j] = h[n-j];
for (i=0; i<n; i++)
    y[0] = y[0] + x[i] * a[i];
for (K=1; K<n; K++)
{
  y[K] = 0;
  for (j = 1; j<n; j++)
      x2[j] = a[j-1];
  x2[0] = a[n-1];
  for (i = 0; i<n; i++)
  {
    a[i] = x2[i];
    y[K] = y[K] + x[i] * x2[i];
  }
}
```

```
printf("Circular convolution is \n");
for(i=0; i<n; i++)
{
  printf("%d\t", y[i]);
}
}.
```

Output:

Enter length of x sequence = 4

Enter length of h sequence = 4

Enter x sequence = 1 2 3 4

Enter h sequence = 4 3 2 1

Circular convolution is :

24 22 24 30.

3/3/14

DISCRETE FOURIER TRANSFORM

Aim: To find DFT of the given sequence

Program:

```
# include < stdio.h>
# include < math.h>
float rw[15][15], iw[15][15], x[15],
      rx[15], ix[15]
main()
{
 int l, k, m, N;
 float p;
 p = 3.14;
 printf("\n Enter length of signal");
 scanf("%d", &m);
 printf("\n Enter sequence");
 for(l=0; l<m; l++)
    scanf("%f", &x[l]);
 printf("\n Enter length of dft signal");
 scanf("%d", &N);
if(N<m)
 { printf("\n Error");
 }
```

```
else
{
   for(l=0; l<N; l++)
   {
    for(K=0; K<N; K++)
    {
      rw[l][K] = cos(2*p*l*K/N);
      iw[l][K] =(-1)*sin(2*p*l*K/N);
    }
   }
   for (l=0; l<N; l++)
   {
    rX[l] =0;
    iX[l] =0;
    for (K=0; K<N; K++)
    {
      rX[l] = rX[l] + x[K] * rw[K][l];
      iX[l] = iX[l] + x[K] * iw[K][l];
    }
   }
   print+("\n");
   for (l=0; l<N; l++)
   {
    printf (" %.f + i(%.f)\t ",
                rX[l], iX[l]);
   }
```

```
    printf("\n");
}.
```

Output:

```
Enter length of signal = 5
Enter sequence = 1 3 5 7 9
Enter length of dft signal = 5
25.00 + i(0.00)   -5.02 + i(6.87)
-5.01 + i(1.59)   -4.98 + i(-1.66)
-4.90 + i(-6.91).
```

5/3/18

EXP - 16

INVERSE DISCRETE FOURIER TRANSFORM

Aim: To find the IDFT of the given sequence.

Program:

```
#include <stdio.h>
#include <math.h>
float rw[15][15], iw[15][15], x[15],
      rX[15], iX[15]
main()
{
 int l, k, m, N;
 float p;
 p = 3.14;
 printf("\n Enter length of signal");
 scanf("%d", &m);
 printf("\n Enter real sequence");
 for (l=0; l<m; l++)
     scanf("%f", &rX[l]);
 printf("\n Enter imag sequence");
 for (l=0; l<m; l++)
     scanf("%f", &iX[l]);
```

```
printf("\n Enter length of dft signal");
scanf("%d", &N);
if (N<m)
{
   printf("\n Error");
}
else
{
   for(l=0; l<N; l++)
   {
     for(K=0; K<N; K++)
     {
       rw[l][K] = cos(2*p*l*K/N);
       iw[l][K] = sin(2*p*l*K/N);
     }
   }
   for(l=0; l<N; l++)
   {
     x[l]=0;
     for(K=0; K<N; K++)
     {
       x[l] = x[l] + rx[K]* rw[K][l];
     }
   }
   for(l=0; l<N; l++)
   {
     for(K=0; K<N; K++)
     {
```

```
        x[l] = x[l] - (ix[k] * iw[k][l]);
      }
      x[l] = x[l]/N;
    }
    printf("\n");
    for (l=0; l<N; l++)
        printf("%.f\t", x[l]);
    }
}
```

Output:

Enter length of signal = 5

Enter real sequence = 1 3 5 7 9

Enter imag sequence = 2 4 6 8 10

Enter length of dft signal = 5

0.00 1.37 0.31 -0.33 -1.38.