Matlab code:CT Signal:

```
clc;
clear all;
close all;
fm=0.25;
t=-10:0.1:10;
a=5*sin(2*pi*fm*t);
plot(t,a);
xlabel('time');
ylabel('amplitude');
title('sinosidal signal');
```

Matlab code:DT Signal:

```
clc;
clear all;
close all;
fm=0.25;
n=-10:0.1:10;
a=5*sin(2*pi*fm*n);
stem(n,a);
xlabel('number of samples');
ylabel('amplitude');
title('sinosidal DT signal');
```

Matlab code: Unit Step CT Signal:

```
clc;
clear all;
close all;
t1= input('lower limit');
t2=input('higher limit');
t=t1:t2;
```

```
x=t>=0;
plot(t,x);
xlabel('time');
ylabel('amplitude');
title('unit step CT signal');
```

Matlab code: Unit Step DT Signal:

```
clc;
clear all;
close all;
n1= input('lower limit');
n2=input('higher limit');
n=n1:n2;
x=n>=0;
stem(n,x);
xlabel('number of samples');
ylabel('amplitude');
title('unit step DT signal');
```

Matlab code: CT Impulse Signal:

```
clear all;
close all;
n1= input('lower limit');
n2=input('higher limit');
n=n1:n2;
x=n>=0;
stem(n,x);
xlabel('number of samples');
ylabel('amplitude');
title('impules CT signal');
```

Matlab code for DT Impulse Signal:

```
clc;
clear all;
close all;
n1= input('lower limit');
n2=input('higher limit');
n=n1:n2;
x=n>=0;
stem(n,x);
xlabel('number of samples');
ylabel('amplitude');
title('impules DT signal');
```

Matlab code For Ramp Signal:

```
clc;
clear all;
close all;
t1=input('enter lower limit');
t2=input('enter higher limit');
t=t1:t2;
a=input('enter the value of a');
y=exp(a*t);
subplot(2,1,1);
plot(t,y,'r');
xlabel('time');
ylabel('amplitude');
title('continuous');
subplot(2,1,2);
stem(t,y,'b');
xlabel('number of samples');
ylabel('amplitude');
```

```
title('discrete');
clc;
clear all;
close all;
n=input('enter higher limit');
n=0:n;
subplot(2,1,1);
stem(n,n);
xlabel('number of samples');
ylabel('amplitude');
subplot(2,1,2);
plot(n,n);
xlabel('time');
ylabel('amplitude');
title('ramp signal');
```

Matlab code for Increasing Exponential Signal:

```
clc;
clear all;
close all;
t1=input('enter lower limit');
t2=input('enter higher limit');
t=t1:t2;
a=input('enter the value of a');
y=exp(a*t);
subplot(2,1,1);
plot(t,y,'r');
xlabel('time');
ylabel('amplitude');
title('continuous');
subplot(2,1,2);
```

```
stem(t,y,'b');
xlabel('number of samples');
ylabel('amplitude');
title('discrete');
```

Matlab code for Decreasing Exponential Signal:

```
clc;
clear all;
close all;
t1=input('enter lower limit');
t2=input('enter higher limit');
t=t1:t2;
a=input('enter the value of a');
y=exp(a*t);
subplot(2,1,1);
plot(t,y,'r');
xlabel('time');
ylabel('amplitude');
title('continuous');
subplot(2,1,2);
stem(t,y,'b');
xlabel('number of samples');
ylabel('amplitude');
title('discrete');
```

MATLAB code to perform basic operations on signals:-Experiment no 2

```
clc;
clear all;
close all;
t1 =input('Enter the amount to be delayed');
t2= input('Enter the amount to be adavanced');
```

```
t=0:6;
x=[1,1,1,1,1,1,1];
subplot(3,1,1);
plot(t,x);
title('signal x(t)');
m=t+t1;
y=x;
subplot(3,1,2);
plot(m,y)
 title('Delayed signal x(n-n1)');
 t=t-t2;
 z=x;
 subplot(3,1,3);
plot(t,z);
title(' signal t-t2');
xlabel('no of samples')
ylabel('amplitude');


clc;
clear all;
close all;
n1 =input('Enter the amount to be delayed');
n2= input('Enter the amount to be adavanced');
n=-2:2;
x=[3,2,1,3,4];
subplot(3,1,1);
stem(n,x);
title('signal x(n)');
m=n+n1;
y=x;
subplot(3,1,2);
```

```
stem(m,y);
title('Delayed signal x(n-n1)');
t=n-n2;
z=x;
subplot(3,1,3);
stem(t,z);
title(' signal n-n2');
xlabel('no of samples')
ylabel('amplitude');

clc;
clear all;
close all;
n1 =input('Enter the amount to be delayed');
n2= input('Enter the amount to be adavanced');
n=0:6;
x=[1,1,1,1,1,1,1];
subplot(3,1,1);
stem(n,x);
title('signal x(n)');
m=n+n1;
y=x;
subplot(3,1,2);
stem(m,y);
title('Delayed signal x(n-n1)');
t=n-n2;
z=x;
subplot(3,1,3);
stem(t,z);
title(' signal n-n2');
xlabel('no of samples')
```

```
ylabel('amplitude');
```

MULTIPLICATION OF TWO SIGNAL:

```
Clc;
Clear all;
Close all;
x1=[1 2 3 4];
subplot(3,1,1);
stem(x1);
xlabel('no. of samples');
ylabel('amplitude');
title('first signal');
x2=[4 3 2 1];
subplot(3,1,2);
stem(x2);
xlabel('no. of samples');
ylabel('amplitude');
title('second signal');
x3=x1.*x2;
subplot(3,1,3);
stem(x3);
xlabel('no. of samples');
ylabel('amplitude');
title('multiplied signal');
display(x1);
display(x2);
display(x3);
```

ADDTION OF TWO SIGNAL:

```
clc;
clear all;
```

```
close all;
x1=[1 2 3 4];
subplot(3,1,1);
stem(x1);
xlabel('no. of samples');
ylabel('amplitude');
title('first signal');
x2=[4 3 2 1];
subplot(3,1,2);
stem(x2);
xlabel('no. of samples');
ylabel('amplitude');
title('second signal');
x3=x1+x2;
subplot(3,1,3);
stem(x3);
 xlabel('no. of samples');
 ylabel('amplitude');
title('addition of signal');
display(x1);
display(x2);
display(x3);
```

SUBTRACTION OF TWO SIGNAL

```
clc;
clear all;
close all;
x1=[1 2 3 4];
subplot(3,1,1);
stem(x1);
xlabel('no. of samples');
```

```
ylabel('amplitude');
title('first signal');
x2=[4 3 2 1];
subplot(3,1,2);
stem(x2);
xlabel('no. of samples');
ylabel('amplitude');
title('second signal');
x3=x1-x2;
subplot(3,1,3);
stem(x3);
xlabel('no. of samples');
ylabel('amplitude');
title('substraction of signal');
display(x1);
display(x2);
display(x3);
```

Experiment No. 3

Verification of Sampling theorem:

```
clc;
clear all;
close all;
fm=0.25;
t=-10:0.1:10;
%construction of original signal
a=5*sin(2*pi*fm*t);
subplot(5, 1, 1);
plot(t,a);
xlabel('time');
ylabel('amp');
```

```
title('originl signal' );
%sampling when fs is greater than fm
fs=10*fm;
n= -50:50;
a_n=5*sin(2*pi*fm*n/fs);
subplot (5,1,2);
%figure;
stem(n,a_n);
xlabel('time');
ylabel('amp');
title('discrete time signal with fs>2fm' );
%sampling when fs is greater than fm
fs=1.2*fm;
n= -4:4;
a_n=5*sin(2*pi*fm*n/fs);
```

Experiment No.4

Linear convolution Using DFT and IDFT / Linear convolution using circular convolution:

```
clc;
clear all;
x1=input('enter the first sequence');
x2=input('enter the second sequence');
n=input('enter the no of points of the dft');
subplot(3,1,1);
stem(x1,'filled');
title('plot of first sequence');
subplot (3,1,2);
stem (x2, 'filled');
title('plot the second sequnce');
n1 = length(x1);
n2 = length(x2);
```

```
m = n1+n2-1;
% Length of linear convolution
x= [x1 zeros(1,n2-1)];
% Padding of zeros to make it of % length m
y = [x2 zeros(1,n1-1)];
x_fft = fft(x,m);
y_fft = fft(y,m);
dft_xy =x_fft.*y_fft;
y=ifft(dft_xy,m);
disp('the circular convolution result is ........')
disp(y);
subplot(3,1,3);
stem (y, 'filled');
title('plot of circularly convoluted sequence');
```

Experiment No.5

Circular convolution of two given sequences:

```
clc;
clear all;
x1=input('enter the first sequence');
x2=input('enter the second sequence');
n1 = length(x1);
n2 = length(x2);
subplot(3,1,1);
stem(x1,'filled');
title('plot of first sequence');
subplot(3,1,2);
stem(x2,'filled');
title('plot the second sequnce');
y1=fft(x1,n);
y2=fft(x2,n);
```

```
y3=y1.*y2;
y=ifft(y3,n);
disp('the circular convolution result is ..........................................................');
disp(y);
subplot(3,1,3);
stem(y,'filled');
title('plot of circularly convoluted sequence');
```

Experiment No.6

Computation of N point DFT of a given sequence and to plot magnitude and phase spectrum:

```
clc;
clear all;
close all;
N = input('enter the value of N');
x = input('enter the sequence for which DFT is to be calculated');
n=(0:1:N-1);
k=(0:1:N-1);
WN =exp(-1j*2*pi/N);
nk=n'*k;
WNnk=WN.^nk;
Xk=x*WNnk;
MagX=abs(Xk);
%magnitue of calculated DFT
phaseX=angle(Xk)*180/pi;
%phase of the calculated DFTfigure(1);
subplot(2,1,1);
plot(k,MagX);
subplot(2,1,2);
plot(k,phaseX);
```

Experiment No.7

Impulse response of a given system:

```
clc;
clear all;
close all;
```

Experiment No.7

Impulse response of a given system:

```
% Difference equation of a second order system
% y(n) = x(n)+0.5x(n-1)+0.85x(n-2)+y(n-1)+y(n-2)
b=input('enter the coefficients of x(n),x(n-1)---------------------------------- ');
a=input('enter the coefficients of y(n),y(n-1)---------------------------------');
N=input('enter the number of samples of imp response ');
[h,t]=impz(b,a,N);
plot(t,h);
title('plot of impulse response');
ylabel('amplitude');
xlabel('time index------------------>N');
disp(h);
grid on;
```

Experiment No.8

Design and implementation ofIIR BUTTERWORTH filter to meet given specifications:

```
clc;
clear all;
close all;
wp=500;
% Enter the pass band frequency
ws=2000;
% Enter the stop band frequency
Rp=3;
% Enter the pass band ripple
Rs=20;
```

```
% Enter the stop band attenuation

Fs=8000;

% Enter the sampling frequency

Fn=Fs/2;

% Normalized sampling frequency

% Find the order n and cutt off frequency

[n,wc]=buttord(wp/Fn,ws/Fn,Rp,Rs);

% Find the filter co-efficients

[b,a]=butter(n,wc);

disp(n);

% Plot the frequency response

[======================

======================

===========h,f]=freqz(b,a,5

12,8000);

plot(f,20*log10(abs(h))) grid
```

Experiment No.9

Implementation of FIR digital filter using window (Rectangular, Hamming, Hanning, Bartlett) methods:

```
clc;

wc=.5*pi;

N=25;

w=0:0.1:pi;

b=fir1(N, wc/pi, blackman(N+1));

h=freqz(b,1,w);

subplot(3,2,1) plot(w/pi, abs(h)) grid;

xlabel('normalised frequency');

ylabel('magnitude in dB')

title('FIR LPF USING BLACKMAN WINDOW')

b=fir1(N, wc/pi, hamming(N+1));
```

```
h=freqz(b,1,w)

subplot(3,2,2) plot(w/pi, abs(h));

grid;

xlabel('normalised frequency');

ylabel('magnitude in dB')

title('FIR LPF USING HAMMING WINDOW')

b=fir1(N,wc/pi, hanning(N+1));

h=freqz(b,1,w);

subplot(3,2,3) plot(w/pi, abs(h));

grid; xlabel('normalised frequency');

ylabel('magnitude in dB')

title('FIR LPF USING HANNING WINDOW')

b=fir1(N,wc/pi, kaiser(N+1,3.5));

h=freqz(b,1,w);

subplot(3,2,4) plot(w/pi, abs(h));

grid;

xlabel('normalised frequency');

ylabel('magnitude in dB') title('FIR LPF USING KAISER WINDOW')


%FIR Filter design window techniques

clc;

clear all;

close all;

rp=input('enter passband ripple');

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

fp=input('enter passband freq'); fs=input('enter

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

beta=input('enter beta value'); 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;
c=input('enter your choice of window function 1. rectangular 2. triangular 3.kaiser: \n ');
if(c==1) y=rectwin(n1);
disp('Rectangular window filter response');end
if (c==2) y=triang(n1);
disp('Triangular window filter response');end
if(c==3) y=kaiser(n1,beta);
disp('kaiser window filter response');end
%HPF
b=fir1(n,wp,'high',y);
[h,o]=freqz(b,1,256);
m=20*log10(abs(h));
plot(o/pi,m);
title('HPF');
ylabel('Gain in dB-->');
xlabel('(b) Normalized frequency-->');
```

INPUT:

enter passband ripple:0.02

enter the stopband ripple:0.01

enter passband freq:1000

enter stopband freq:1500

enter sampling freq: 10000

enter beta value:5

OUTPUT WAVEFORM:

enter your choice of window function 1. rectangular 2. triangular 3.kaiser:2