3.)

```
#include<stdio.h>
#define npt 4
int n, k, y[2 * npt - 1];
int x[npt], h[npt];


main() {
        printf("enter% point input sequence 1;\n", npt);
        for (n = 0; n < npt; n++) {
                scanf("%d", &x[n]);



        }
        printf("enter %d point input sequence2:\n",npt);
        for(n=0;n<npt;n++)
        {
                scanf("%d",&h[n]);
        }
        printf("convolution sequence y[n]=SUM(k) x(k)*y[n-k]:\n");
        for (n = 0; n < (2 * npt - 1); n++) {
                y[n] = 0;
                for (k = 0; k < npt; k++) {
                        if ((n - k) >= 0)
                                if ((n - k) < npt)
                                        y[n] = y[n] + x[k] * h[n - k];
```

```
            }
            printf("%d\t", y[n]);


        }
        printf("\n");
        return 0;
```

SCILAB

```
clc;
clear;
a1=input("enter 1st sequence");
b1=input("enter 2nd sequence");
c1=conv(a1,b1);
m2=length(c1)-1;
disp("output sequence");
disp(c1);
n1=0:m2;
plot2d3(n1,c1);
xlabel("time index n");
ylabel("amplitude");
title("linear convolution");
```

4.)

```
#include<stdio.h>
#define npt 4
int n, m, k, l, y[npt];
int x[npt], h[npt];
main() {
```

```
        printf("enter% point input sequence 1:\n", npt);
        for (n = 0; n < npt; n++) {
                scanf("%d", &x[n]);


        }
        printf("enter %d point input sequence2:\n",npt);
        for(n=0;n<npt;n++){
                scanf("%d",&h[n]);
        }
        printf("circular convolution sequence x3[m]=SUM(n)x1(n)*x2(m-n)modulon:\n");
        for (m = 0; m < npt; m++) {
                y[m] = 0;
                for (n = 0; n < npt; n++) {
                        if ((m-n) < 0)
                                        y[m] = y[m] + x[n] * h[npt+(m-n)];
                        else
                                y[m]=y[m]+x[n]*h[m-n];
                }
                printf("%d\t", y[m]);


        }
        printf("\n");
        return 0;SCILAB:
clc;
clear;
x=input("1st sequence=");
```

```
y=input("2nd sequence=");
a=fft(x);
b=fft(y);
c=a.*b;
disp("output signals");
disp(ifft(c));
plot2d3(ifft(c));
xlabel("time index n");
ylabel("amplitude");
title("circular convolution");
```

5.)

```
#include<stdio.h>
#define npt 4
int n, k, l, y[2*npt-1];
int x[npt], h[npt];
main() {
        printf("enter% point input sequence 1:\n", npt);
        for (n = 0; n < npt; n++) {
                scanf("%d", &x[n]);


        }
        printf("enter %d point input sequence2:\n",npt);
        for(n=0,l=(npt-1);n<npt;n++,l--){
                scanf("%d",&h[n]);
        }
        printf("correlation sequence Rxy[1]=SUM(n) x[n]*y[n-1]:\n");
```

```
        for (l =-(npt-1),k=0;l<npt;l++, k++) {
                y[k] = 0;
                for (n = 0; n < npt; n++){
                        if ((n-l) >= 0)
                                if((n-l)<npt)
                                        y[k]=y[k]+x[n]*h[n-l];
                }
                printf("%d\t", y[k]);


        }
        printf("\n");
        return 0;SCILAB:
clear;
clc;
x=input("enter the value of x=");
h=input("enter the value of h=");
subplot(3,1,1);
plot2d3(x);
xlabel("time index n");
ylabel("amplitude");
title("sequence of x(n)");
subplot(3,1,2);
plot2d3(h);
xlabel("time index n");
ylabel("amplitude");
title("sequence of h(n)");
```

```
b=xcorr(x,h);

disp(b);

subplot(3,1,3);

plot2d3(b);

xlabel("time index n");

ylabel("amplitude");

title("cross correlation");
```

6.)

```
//AMPLITUDE SCALING 2.0

clear;

clc;

t=0:.01:1;

A=2 ;

f= 1;

x=A*sin (2*%pi *f*t);

x1=2*x;

figure(1)

subplot (2,1, 1) ;

plot (t, x, 'r'); title( 'Plot of sinusoid '); xlabel ( 'Time') ;

ylabel ( 'Amplitude');

subplot (2, 1,2) ;

plot (t, x1, 'g'); title( 'Plot of its amplitude scaled version, scaling factor=2, notice doubling of amplitude');

xlabel ( 'Time');

ylabel ( 'Amplitude');

//AMPLITUDE SCALING 0.5

clear;
```

```
clc;

t=0:.01:1;

A=2;

f=1;

x=A*sin(2*%pi*f*t);

x1=2*x;

figure(1);

subplot (2, 1, 1) ;

plot (t, x, 'r'); title ('Plot of sinusoid '); xlabel ( 'Time') ;

ylabel ('Amplitude') ;

subplot (2, 1,2) ;

plot (t, x1, 'g'); title('Plot of its amplitude scaled version, scaling factor=2,notice doubling of amplitude');

xlabel ('Time');

ylabel ( 'Amplitude') ;

t=0:.01:1;

A=2;

f=1;

x=A*sin(2*%pi*f*t);

x1=5*x;

figure(2) ;

subplot (2,1, 1) ;

plot (t, x, 'r'); title ('Plot of sinusoid ');

xlabel ('Time') ;

ylabel ( 'Amplitude') ;

subplot (2, 1,2) ;

plot (t, x1, 'g'); title('Plot of its amplitude scaled version, scaling factor=0.5, notice halving of amplitude');
```

```
xlabel ( 'Time');
ylabel ( 'Amplitude') ;
//ADDITIN OF SIGNALS
clear;
clc;
t=0:.01:1;
A=2;
f=1;
x=A*sin (2*%pi*f*t) ;
x1=A*sin(2*%pi*2*t);
figure (3) ;
subplot (3, 1, 1) ;
plot (t,x, 'r');title ('Plot of sinusoid x'); xlabel ('Time') ;
ylabel ('Amplitude') ;
subplot (3, 1,2) ;
plot (t, x1, 'g'); title( 'Plot of sinusoid x1');
xlabel ('Time');
ylabel ('Amplitude') ;
addn=x+x1;
subplot (3, 1, 3) ;
plot (t, addn, 'r');title ('Plot of addn=x+x1') ;
xlabel ('Time') ;
ylabel ('Amplitude') ;
//SUBSTRACTION
clear;
clc;
t=0:.01:1;
```

```
A=2; f=1;
x=A* sin (2*%pi*f*t) ;
x1=A* sin (2*%pi*2*t) ;
figure (4) ; subplot (3,1, 1) ;
plot (t,x, 'r') ;title ('Plot of sinusoid x') ;
xlabel ('Time') ;
ylabel ( 'Amplitude') ;
subplot (3, 1,2) ;
plot (t, x1, 'g') ;title ('Plot of sinusoid x1') ;
xlabel ( 'Time') ;
ylabel ( 'Amplitude') ;
sub_sig=x-x1;
subplot (3, 1, 3) ;
plot (t, sub_sig, 'r');
title ('Plot of subtracted signal=x-x1'); xlabel ( 'Time');
ylabel ( 'Amplitude') ;
//MULT
clear;
clc;
t=0:.01:1;
A=2 ;
f=1;
x=A*sin (2*%pi*f*t) ;
x1=A*sin (2*%pi*2*t) ;
figure (5) ;
subplot (3, 1, 1) ;
plot (t,x, 'r');title ( 'Plot of sinusoid x');
```

```
xlabel ('Time' ) ;
ylabel ( 'Amplitude') ;
subplot (3, 1,2) ;
plot (t, x1, 'g'); title ('Plot of sinusoid x1') ;
xlabel ( 'Time') ;
ylabel ( 'Amplitude') ;
mul_sig=x.*x1;
subplot (3,1, 3) ;
plot (t, mul_sig, 'r');
title ('Plot of multiplied signal=x.*x1'); xlabel ('Time') ;
ylabel ('Amplitude ') ;
//OPERATION ON IND VARIABLE
clear;
t=0:.01:2;
A=2;
f=1 ;
x=A*sin(2*%pi*f*t);
//case-1: time delay
t1=t-0.2;//time delay =0.2 units
x1=A*sin(2*%pi*f*t1);//Time shifted version of x
figure(7) ;
subplot (211) ;
plot (t, x, 'r'); title ('Plot of sinusoid ');
xlabel ('Time') ;
ylabel ( 'Amplitude');
subplot (212) ;
plot (t,x1, 'g') ;title('Plot of its time delayed version, delay=0.2') ;
```

```
xlabel ('Time') ;
ylabel ( 'Amplitude') ;//case-2: time advance
t1=t+0.2;//time advance =0.2 units
x1=A*sin （2*%pi*f*t1）;//Time shifted version of x
figure (8) ;
subplot (211) ;
plot (t,x, 'r'); title('Plot of sinusoid ');
xlabel ('Time') ;
ylabel ('Amplitude') ;
subplot (212) ;
plot (t, x1, 'g') ;title('Plot of its time advanced version, advanced=0.2');
xlabel ( 'Time');
ylabel ( 'Amplitude') ;
//case-3: time scaling
t1=t*0.5;//scaling factor =0.5
x1=A*sin (2*%pi*f*t1);//Time shifted version of x figure (9) ;
subplot (211) ;
plot (t,x, 'r');title( 'Plot of sinusoid ');
xlabel ('Time' ) ;
ylabel ( 'Amplitude') ;
subplot (212) ;
plot (t, x1, '9'); title( 'Plot of its time scaled version, scaling factor=0.5, notice expansion' ) ;
xlabel ('Time' ) ;
ylabel ( 'Amplitude');
//case-4: time scaling
t1=t*2;//scaling factor =2
x1=A*sin (2*%pi*f*t1);//Time shifted version of x figure (10) ;. subplot (211) ;
```

```
plot (t, x, 'r'); title ('Plot of sinusoid ');
xlabel ('Time') ;
ylabel ( 'Amplitude') ;
subplot (212) ;
plot (t, x1, 'g'); title( 'Plot of its time scaled version, scaling factor=2, notice compression');
xlabel ( 'Time') ;
ylabel ( 'Amplitude');
//GENERATION OF SIGNALS(AM)
clc;
clear;
t=0:.001:3;
A=2;
f=1;
x=A*sin(2*%pi*f*t);
x=3+x;
x1=A*sin (4*2*%pi*50*t) ;
figure (6) ;
subplot (3, 1, 1) ;
plot (t, x, 'r') ;title('Plot of sinusoid, modulating wave ');
xlabel ('Time') ;
ylabel ( 'Amplitude') ;
subplot (3, 1, 2) ;
plot (t, x1, 'r') ;title ('Plot of sinusoid xl, Carrier');
xlabel ('Time' ) ;
ylabel ( 'Amplitude') ;
am_sig=x.*x1;
```

```
subplot (3,1, 3) ;plot(t, am_sig, 'r') ;title('Plot of multiplied signal=x*x1, notice the resemblance of shape of curve to AM wave');

xlabel ('Time' ) ;

ylabel ( 'Amplitude') ;
```

7.)LC OF 4 POINT SEQ USING FFT

```
clear; clc; 1MS22EI010

x=input('first sequence:');

y=input('second sequence:');

k=length(x);

m=length(y);

conv_length=k+m-1;

x=[x,zeros(1,conv_length-k)];

y=[y,zeros(1,conv_length-m)];

p=fft(x);

q=fft(y);

r=p.*q;

figure(1);

z=ifft(r);

disp('Linear convolution:');

disp(z);

plot2d3(z);

title('linear convolution using fft method');

xlabel('Time');

ylabel('Amplitude');
```

C PROGRAM:

```
#include<stdio.h>

#include<math.h>
```

```
#define pi 3.142857142857
#define NPT 8
#define NSTAGE 3
#define NPT_BY_2 4
float ffr[NPT],ffi[NPT];
float y3r[NPT],y3i[NPT];
void fft(float*,float*);
int main() {
	float a1r[NPT]={2,1,2,1,0,0,0,0},a1i[NPT]={0,0,0,0,0,0,0,0};
	float a2r[NPT]={1,2,3,4,0,0,0,0},a2i[NPT]={0,0,0,0,0,0,0,0};
	float y1r[NPT],y1i[NPT];
	float y2r[NPT],y2i[NPT];
	int i;
	fft(a1r,a1i);
	for (i=0;i<NPT;i++)
	{
		y1r[i]=ffr[i];
		y1i[i]=ffi[i];
	}
	fft(a2r,a2i);
	for (i=0;i<NPT;i++)
		{
			y2r[i]=ffr[i];
			y2i[i]=ffi[i];
		}
	for (i=0;i<NPT;i++)
	{
		y3r[i]=y1r[i]*y2r[i]-y1i[i]*y2i[i];
		y3i[i]=y1r[i]*y2r[i]+y1i[i]*y2i[i];
		y3i[i]=(-1)*y3i[i];
```

```
        }
        fft(y3r,y3i);
        for (i=0;i<NPT;i++)
        {
                y3r[i]=ffr[i]/NPT;
                y3i[i]=(-1)*ffi[i]/NPT
        }
        for (i=0;i<NPT-1;i++)
        {
                printf("%f\n",y3r[i]);
        }
        return 0;
}
void fft(float real[NPT],float imag[NPT])
{
        int i,j,k,p,g,tob,bob;
        float b,c,d;
        float tr[NPT_BY_2],ti[NPT_BY_2];
        int i1,work,j2,bb[NPT];
        for (i=0;i<NPT;i++)
        {
                tr[i]=cos(2*pi*i/NPT);
                ti[i]=-sin(2*pi*i/NPT);
        }
        for (k=0;k<NSTAGE;k++)
        {
                if (k==0)
                        g=1;
                else
                        g=pow(2,(k-1));
```

```
            p=pow(2,k);
            for(j=0;j<p;j++){
                    for(i=0;i<(NPT_BY_2/p);i++){
                            tob=i+j*NPT_BY_2/g;
                            bob=tob+(NPT_BY_2/p);
                            b=real[tob]+real[bob];
                            c=imag[tob]+imag[bob];
                            real[bob]=real[tob]-real[bob];
                            imag[bob]=imag[tob]-imag[bob];
                            real[bob]=b;
                            imag[tob]=c;
                            d=real[bob]*tr[p*i]-imag[bob]*ti[p*i];
                            imag[bob]=imag[bob]*tr[p*i]+real[bob]*ti[p*i];
                            real[bob]=d;
                    }
            }
    }
    for (i=0;i<NPT;i++)
    {
            work=0;
            i1=i;
            for (j=0;j<NSTAGE;j++)
            {
                    j2=i1%2;
                    work=2*work+j2;
                    i1=i1/2;
            }
            bb[i]=work;
    }
    for (i=0;i<npt;i++){
```

```
                ffr[i]=real[bb[i]];
                ffi[i]=imag[bb[i]];
        }
}
```

8.)MAG AND PHASE SPECTRUM 8 POINT SEQ.

SCILAB PROGRAM

```
clc;
clear;
x=[1 2 3 4 5 6 7 8];
y=fft(x);
mag=abs(y);
phase=atan(imag(y),real(y));
disp(mag);
disp(phase)
subplot(211);
plot2d3(mag);
title('magnitude spectrum');
xlabel('k--->');
subplot(212);
plot2d3(phase);
title('phase spectrum');
xlabel('k--->');
ylabel('phase');#include<stdio.h>
```

C PROGRAM:

```
#include<math.h>
#define pi 3.142857142857
```

```
#define NPT 8
#define NSTAGE 3
#define NPT_BY_2 4
float ffr[NPT], ffi[NPT];
float mag[NPT], angle[NPT];
void fft(float*, float*);
int main() {
	float ar[] = { 1, 2, 3, 4, 5, 6, 7, 8 };
	float ai[] = { 0, 0, 0, 0, 0, 0, 0, 0 };
	int i;
	for (i = 0; i < NPT; i++) {
		ffr[i] = 0;
		ffi[i] = 0;
	}
	fft(ar, ai);
	for (i = 0; i < NPT; i++) {
		mag[i] = sqrt((ffr[i] * ffr[i]) + (ffi[i] * ffi[i]));
		angle[i] = atan2(ffi[i], ffr[i]);
	}
	return (0);
}
void fft(float real[NPT], float imag[NPT]) {
	int i, j, k, p, g, tob, bob;
	float b, c, d;
	float tr[NPT_BY_2], ti[NPT_BY_2];
	int i1, work, j2, bb[NPT];
	/*TWIDLE FACTOR*/
```

```
        for (i = 0; i < NPT_BY_2; i++) {
        tr[i] = cos(2 * pi * i / NPT);
        ti[i] = -sin(2 * pi * i / NPT);
}
/*actual fft computation*/
        for (k = 0; k < NSTAGE; k++) {
        if (k == 0)
                g = 1;
        else
                g = pow(2, (k - 1));
        p = pow(2, k);
        for (j = 0; j < p; j++) {
                for (i = 0; i < (NPT_BY_2 / p); i++) {
                        tob = i + j * NPT_BY_2 / g;
                        bob = tob + (NPT_BY_2 / p);
                        b = real[tob] + real[bob];
                        c = imag[tob] + imag[bob];
                        real[bob] = real[tob] - real[bob];
                        imag[bob] = imag[tob] - imag[bob];
                        real[tob] = b;
                        imag[tob] = c;
                        d = real[bob] * tr[p * i] - imag[bob] * ti[p * i];
                        imag[bob] = imag[bob] * tr[p * i] + real[bob] * ti[p * i];
                        real[bob] = d;
                }
        }
}
```

```
/*to arrange bit reversed fft values*/for (i = 0; i < NPT; i++) {
        work = 0;
        i1 = i;
        for (j = 0; j < NSTAGE; j++) {
                j2 = i1 % 2;
                work = 2 * work + j2;
                i1 = i1 / 2;
        }
        bb[i] = work;
}
for (i = 0; i < NPT; i++) {
        ffr[i] = real[bb[i]];
        ffi[i] = imag[bb[i]];
}
}
```

9.)FIRST ORDER BUTTER LPF:

```
#include<stdio.h>
#include<math.h>
#define pi 3.14159264 float fs, T, t, x[256], y[256];
int i;
int main() {
fs = 10000.0;
T = 1/fs;
for (i = 0, t = 0.0; i < 256; i++,t += T) {
x[i] = sin(2*pi*117.1875*t) + sin(2*pi*3906.25*t) + sin(2*pi*585.9375*t);
} y[0] = 0.0;
for(i = 1; i < 256;i++)
```

```
{ y[i] = -0.1583844 * y[i-1] + 0.5791922 * (x[i] + x[i-1]);
}
return 0;
}
```

Scilab code:

```
clc;
clear;
N=1;
fs=10000;
fsb2=fs/2;
fc=3000;
t=0:(1/fs):0.1023;
x=sin(2*%pi*100*t)+sin(2*%pi*4000*t)+sin(2*%pi*500*t);
hz=iir(N,'lp','butt',[(fc/fs) 0],[])
[hzm,fr]=frmag(hz,256);
fr=fr*fs;
plot2d(fr',hzm');
xtitle('Discrete IIR filter: low pass');
hz
B=[0.5791922 0.5791922]
A=[1 0.1583844]
y=filter(B,A,x);
figure(1);
subplot(211);
plot(x);
title('time domain plot of input signal');
```

```
subplot(212);
plot(y);
title('time domain plot of filtered signal');
X=abs(fft(x));
Y=abs(fft(y));
figure(2);
subplot(211);
plot(X(1:512));
title('FREQUENCY domain plot of input signal');
subplot(212);
plot(Y(1:512));
title('FREQUENCY domain plot of filtered signal');
```

10.)SECOND ORDER LPF:

```
clc;
clear;
N=2;
fs=10000;
fsb2=fs/2;
fc=3000;
t= 0:(1/fs):0.1023;
x=sin(2*%pi*100*t)+sin(2*%pi*4000*t) +sin (2*%pi*500*t);
hz=iir(N,'lp','butt', [(fc/fs) 0],[])
[hzm, fr] =frmag (hz, 256) ;
fr=fr*fs;
plot2d(fr',hzm');
xtitle('Discrete IIR filter: low pass');
```

```
B= [0.3913 0.7826 0.3913]

A= [1 0.369 0.1958]

y=filter (B,A,x);

figure(1) ;

subplot (211); plot (x);

subplot (212); plot (y);

X=abs(fft(x));

Y=abs(fft(y));

figure(2);

subplot (211); plot (X (1:512))

subplot (212) ; plot (Y (1:512))#include<stdio.h>

#include<math.h>

#define pi 3.14159264

float fs,T,t,x[256],y[256];

int i;

int main(){

        fs=10000.0;

        T=1 / fs;

        for(i=0,t=0.0;i<256;i++,t+=T){

                x[i] =sin(2*pi*117.1875*t)+sin(2*pi*3906.25*t)+sin(2*pi*585.9375*t);



        }

        y[0]=0.0;

        y[1]=0.0;

        for(i=2;i<256;i++){

                y[i]=-0.3695274*y[i-1]-0.1958157*y[i-2]+0.3913358*(x[i]+x[i-2])+0.7826715*x[i-1];

        }

        return 0;}
```

13.)LINEAR PHASE FIR LPF HAMMING WINDOW

```
#include<stdio.h>
#include<math.h>
#define pi 3.14159264
float x[256],y[256];
int i;
int main() {
float fs,T,t;
fs=10000.0;
T=1/fs;
for (i=0,t=0.0;i<256;i++,t+=T){
x[i]=sin(2*pi*117.182*t)+sin(2*pi*585.93*t)+sin(2*pi*2226.25*t);
}
for (i=6;i<256;i++)
{
y[i]=x[i]*0.0060021+0.049338*x[i-1]+0.1733109*x[i-2]+0.25*x[i-
3]+0.1733109*x[i-4]+0.049338*x[i-5]+x[i-6]*0.0060021;
}
return 0;
}
```

SCILAB

```
clc;
clear;
M=7;
Wc=%pi/4;
Tuo=(M-1)/2;
for n=1:M
if (n==Tuo+1)
hd(n)=Wc/%pi;
else
```

```
hd(n)=sin(Wc*((n-1)-Tuo)*%pi);
end
end
W=window('hm',M);
W=W';
h=hd.*W;
disp('Filter Coefficients are')
disp(h);
[hzm,fr]=frmag(h,256);
hzm_dB=20*log10(hzm)./max(hzm);
figure(1);
plot(fr,hzm_dB)
xlabel('Normalized digital frequency W');
ylabel('Frequency response of FIR LPF using rectangular window M=7')
fs=10000;
fsb2=fs/2;
fc=3000;
t=0:(1/fs):0.1023;
x=sin(2*%pi*100*t)+sin(2*%pi*4000*t)+sin(2*%pi*500*t);
hz=iir(N,'lp','butt',[(fc/fs) 0],[])
[hzm,fr]=frmag(hz,256);
fr=fr*fs;
plot2d(fr',hzm');
xtitle('Discrete IIR filter:low pass');
B=[0.3913 0.7826 0.3913]
A=[1 0.369 0.1958]
y=filter(B,A,x);
figure(1);
subplot(211);plot(x);title('Time domain plot of input signal');
subplot(212);plot(y);title('Time domain plot of filtered signal');
```

```
X=abs(fft(x));

Y=abs(fft(y));

figure(2);

subplot(211);plot(x(1:512));title('Frequency domain plot of input signal');

subplot(212);plot(y(1:512));title('Frequency domain plot of filtered signal');

fs=10000;

t=0:(1/fs):0.1023;

x=sin(2*%pi*100*t)+sin(2*%pi*4000*t)+sin(2*%pi*500*t);

B=h

A=[1]

y=filter(B,A,x);

figure(2);

subplot(211);plot(x);title('Time domain plot of input signal');

subplot(212);plot(y);title('Time domain plot of filtered signal');

X=abs(fft(x));

Y=abs(fft(y));

figure(3);

subplot(211);plot(X(1:512));title('Frequency domain plot of input signal');

subplot(212);plot(Y(1:512));title('Frequency domain plot of filtered signal');

end
```

)))))EXPERIMENT 14:(HANNING WINDOW)

CCS

```
#include<stdio.h>

#include<math.h>

#define pi 3.14159264

float x[256] ,y[256];

int main()

{

float fs,t,T;

int i;
```

```
fs = 10000.0;
T =1/fs;
for(i=0,t=0.0;i<256;i++,t+=T)
{
x[i] =sin(2*pi*117.182*t)+sin(2*pi*585.93*t)+sin(2*pi*2226.25*t);
}
for(i=5 ;i<256;i++)
{
y[0] = 0.0;
y[1] = 0.0;
y[2] = 0.0;
y[3] = 0.0;
y[4] = 0.0;
y[i] = 0.0397887*x[i-1]+0.1688093*x[i-2]+0.25*x[i-3]+0.1688093*x[i-4]+0.0397887*x[i-5];
}
return 0;
}
```

SCILAB

```
clear;
clc;
M=7
wc=%pi/4;
tuo=(M-1)/2;
for n=1:M
if(n==tuo+1)
hd(n)=wc/%pi;
else
hd(n)=sin(wc*((n-1)-tuo))/(((n-1)-tuo)*%pi);
```

```
end
end
w=window('hn',M);
w=w';
h=hd.*w;
disp('filter coefficients are')
disp(h);
[hzm,fr]=frmag(h,256);
hzm_db=20*log10(hzm)./max(hzm);
figure(1);
plot(fr,hzm_db);
xlabel('Normailsed digital frequency w');
ylabel('magnitude in db');
title('frequency response of FIR LPF using rectangular window M=7');
fs=10000;
t=0:(1/fs):0.1023;
x=sin(2*%pi*100*t)+sin(2*%pi*4000*t)+sin(2*%pi*500*t);
B=h
A=[1]
y=filter(B,A,x);
figure(2);
subplot(211);plot(x)
subplot(212);plot(y)
X=abs(fft(x));
Y=abs(fft(y));
figure(3);
subplot(211);plot(X(1:512))
subplot(212);plot(Y(1:512))
[(0:(M-1))' hd w h]
```

EXPERIMENT 15:(RECTANGULAR WINDOW)

SCILAB

```
clear;
clc;
M=6
Wc=1;
Tuo=(M-1)/2
for n=1:M
if (n == Tuo+1)
hd(n)=Wc/%pi;
else
hd(n)= sin(Wc*((n-1)-Tuo))/(((n-1)-Tuo)*%pi);
end
end
W=window('re',M);
W=W';
h=hd.*W;
disp('filter coefficients are')
disp(h);
[hzm,fr]=frmag(h,256);
hzm_dB=20*log10(hzm)./max(hzm);
figure(1);
plot(fr,hzm_dB)
xlabel('normalized digital frequency W');
ylabel('Magnitude in dB');
title('frequency response of FIR LPF using Rectangular window M=7');
fs=10000;
t=0:(1/fs):0.1023;
x=sin(2*%pi*100*t)+sin(2*%pi*4000*t)+sin(2*%pi*500*t);
B=h//numerator polynomial
```

```
A=[1]//denominator polynomial
y=filter(B,A,x);
figure(2);
subplot(211);plot(x)
subplot(212);plot(y)
X=abs(fft(x));
Y=abs(fft(y));
figure(3);
subplot(211);plot(X(1:512))
subplot(212);plot(Y(1:512))
[(0:(M-1))'hd W h]
```

C PROGRAM:

```
#include<stdio.h>
#include<math.h>
#define pi 3.1416
float x[256],y[256];
int main()
{
        float fs,t,T;
        int i;
        fs=10000.0;
        T=1/fs;
        for (i=0,t=0.0;i<256;i++,t+=T)
        {
                x[i]=sin(2*pi*117.182*t)+sin(2*pi*585.93*t)+sin(2*pi*226.25*t);
        }
        for(i=5;i<256;i++){
                y[i]=y[0] = 0.0;
                 y[1] = 0.0;
```

```
                y[2] = 0.0;
                y[3] = 0.0;
                y[4] = 0.0;
                y[i] = 0.0397887*x[i-1]+0.1688093*x[i-2]+0.25*x[i-3]+0.1688093*x[i-4]+0.0397887*x[i-5];
        }
}
```