1. linear convolution matlab

```
clc;
x=input('enter input signal-1[]')
N1=length(x);
n1=0:1:N1-1;
subplot(3,1,1)
stem(n1,x)
xlabel('n')
ylabel('x(n)')
h=input('enter input sig 2')
N2=length(h);
n2=0:1:N2-1;
subplot(3,1,2)
stem(n2,h)
xlabel('n')
ylabel('h(n)')
y=conv(x,h)
n3=1:N1+N2-1;
subplot(3,1,3)
stem(n3,y)
xlabel('n');
ylabel('y(n)');
title('linear convolution')
```

2.circular convolution in C language

```
#include<stdio.h>
#include<conio.h>
#define pf printf
#define sf scanf
void main(void)
{
int a,i,j,k,x[10],h[20],c,t[10][10],sum=0,y[10];
clrscr();
pf("\nenter the length of 2 signals");
sf("%d",&a);
pf("\nenter the sequence of x");
for(i=0;i<a;i++)
sf("%d",&x[i]);
pf("\nenter the sequence h");
for(i=0;i<a;i++)
{
sf("%d",&h[i]);
}
for(i=0;i<a;i++)
{
for(j=0;j<a;j++)
t[j][i]=h[j];
c=h[a-1];
for(k=a-2;k>=0;k--)
h[k+1]=h[k];
h[0]=c;
}
pf("Circular convolution is");
for(i=0;i<a;i++)
{
```

```
sum=0;
for(j=0;j<a;j++)
sum=sum+(t[i][j]*x[j]);
y[i]=sum;
printf("\n\n%d",y[i]);
}
getch();
}
```

3 dft without using direct function

```
x = input('enter i/p seq. in square bracket')
N=length(x);
k=0:1:N-1;
n=0:1:N-1;
Wn=exp(-i*2*pi/N);
nk=n'*k;
Wnk=Wn.^nk;
x=x*Wnk;
subplot(2,1,1)
stem(abs(x))
disp(x);
title "amplitude"
subplot(2,1,2)
stem(angle(x))
title"phase"
```

4 idft without using direct function

```
x = input('enter i/p sequence in square bracket')
N=length(x);
k=0:1:N-1;
n=0:1:N-1;
Wn=exp(2*pi*i/N);
nk=n'*k;
Wnk=Wn.^nk;
x=(x*Wnk)/N;
subplot(2,1,1)
stem(abs(x))
disp(x);
title "amplitude"
subplot(2,1,2)
stem(angle(x))
title "phase"
```

5 autocorrelation

```
 x=input ('Enter sequence x(n)=');
  rxx= conv(x,fliplr(x));
  disp('rxx=');
  disp(rxx);
  figure(1);
  stem(rxx,'filled');
  title('Autocorrelation output');
  xlabel('lag index');
  ylabel('amplitude');
```

Result:

```
Enter sequence x(n)=[1 2 3 4]
rxx=
     4    11    20    30    20    11     4
```

6 same as 2nd experiment

7 circular convolution using matlab

```
clc;
x=input('enter input sig 1')
N1=length(x);
h=input('enter input sig 2')
N2=length(h);
n1=0:1:N1-1;
subplot(3,1,1);
stem(n1,x);
xlabel('n');
ylabel('x(n)');
n2=0:1:N2-1;
subplot(3,1,2);
stem(n2,h);
xlabel('n');
ylabel('h(n)');
y=cconv(x,h,4)
n3=0:1:3;
subplot(3,1,3);
stem(n3,y);
xlabel('n3');
ylabel('y(n)');
title('circular convolution')
```

8 dft with using direct form

```
x=input('enter i/p seq. in square bracket')
x=fft(x)
stem(abs(x))
stem(angle(x))
i/p [3 2 1 1]
```

9 IIR filter using impulse invariant method

```
clc;
clear all;
b=input('enter numerator coefficients :');
a=input('enter denorenter coefficients :');
f=input('enter the sampling frequency :');
[bz,az]=impinvar(b,a,f)
```

Output:

```
enter numerator coefficients :[1 2]
enter denorenter coefficients :[1 3 2]
enter the sampling frequency :5

bz =

    0.2000   -0.1341
```

```
az =

    1.0000   -1.4891    0.5488
```

10 IIR filter using bilinear transformation method

```
clc;
clear all;
b=input('enter numerator coefficients :');
a=input('enter denorenter coefficients :');
f=input('enter the sampling frequency :');
[bz,az]=bilinear(b,a,f)
```

Output:

```
enter numerator coefficients :[1 2]
enter denorenter coefficients :[1 3 2]
enter the sampling frequency :5

bz =0.0909    0.0303   -0.0606
az =1.0000   -1.4848    0.5455
```

11 circular convolution in c same as 2nd

12 implement a program to generate exponential signal , sine wave , and unit ramp signal

```
n1=0:0.1:1;
x3=exp(n1);
subplot(3,3,3);
stem(n1,x3);
title('exponential wave');

n=0:0.05:1;
x1=sin(2*pi*n);
subplot(3,3,1);
stem(n,x1);
title('sine wave');

n2=0:1:10;
x3=n2;
subplot(3,3,4);
stem(n2,x3);
title('unit ramp');
```

13 implement a program to generate sine wave , cosine wave , unit step signal

```
n=0:0.05:1;
x1=sin(2*pi*n);
subplot(3,3,1);
stem(n,x1);
title('sine wave');

x8=cos(2*pi*n);
subplot(3,3,9);
stem(n,x8);
title('cosine wave');
```

```
n9=-4:1:4;
x2=[zeros(1,4),ones(1,5)];
subplot(3,3,2);
stem(n9,x2);
title('unit step signal')
```

14 cross correlation

```
 x=input ('Enter sequence x(n)=');
  y=input ('Enter sequence y(n)=');
  rxy= conv(x,fliplr(y));
  disp('rxy=');
  disp(rxy);
  figure(1);
  stem(rxy,'filled');
  title('cross correlation output');
  xlabel('lag index');
  ylabel('amplitude');
```

Result:

```
Enter sequence x(n)=[1 2 3 4]
Enter sequence y(n)=[4 3 2 1]
rxy=
     1     4    10    20    25    24    16
```