Task : 01

1) No, the order of the input doesn't matter in convolution. It is always commutative because it simply requires the integration of a product. And multiplication is always commutative i.e AxB = BxA.
2) The output is a quadratic. Which makes sense because the rectpuls is 1 throughout the integration region and tripuls is t and then -t, which upon integration would give t^2. Also the width of the output increases as the width of the overlapping part increases.

Task: 02

1) The relation is the sum of the lengths of the two signals of the input -1.
2) I think the output length will then be related to the larger length because one signal is moved over the other signal, so the overlapping length will be related to the length of the larger input.

Task: 03

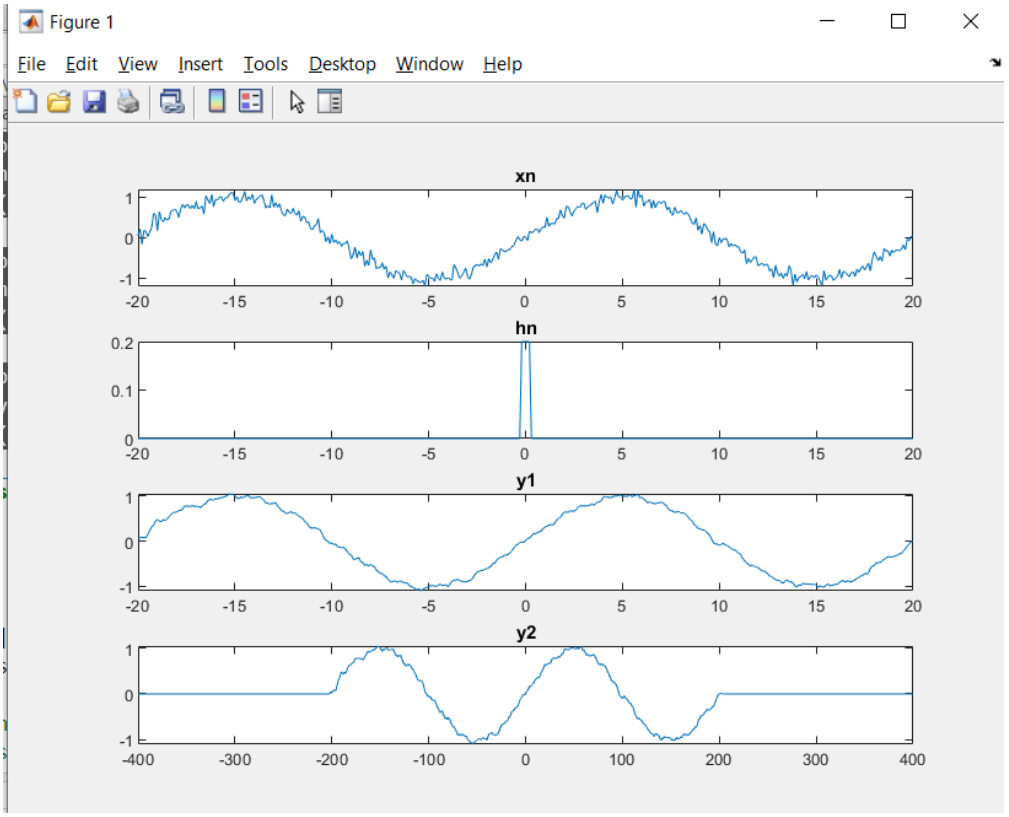
- The shape of y1 and y2 is same but there is a difference in scaling. This is due to two reasons:
    - During convolution we start the integration from the left
    - of the starting area of the input signal and go till the right side, due to which there are extra 0s at the left and the right of the output.
    - There is a difference in length of the x axis vector of y1 and y2. Because y1 is the same size as n, where as the size of y2 is 2n -1 .

CODE:

```
clc
clear all
close all
n = -20:0.1:20;
zn = (sin(0.3142.*n));
noise =  0.1*randn(size(n));
xn = zn + noise;
y1 = movmean(xn,5);
a = 0.001;
impulse = (1./(a*sqrt(pi))).*exp((-1.*(n.^2))./(a ^ 2));
impulse=impulse/(max(impulse));
hn = movmean(impulse,5);
y2 = conv(xn,hn);
Lengthy2 = length(y1) + length(hn) - 1;
y2n = -(Lengthy2-1)/2 : (Lengthy2-1)/2;
length(y2)
length(n)
subplot 411
plot(n,xn)
title('xn')
```

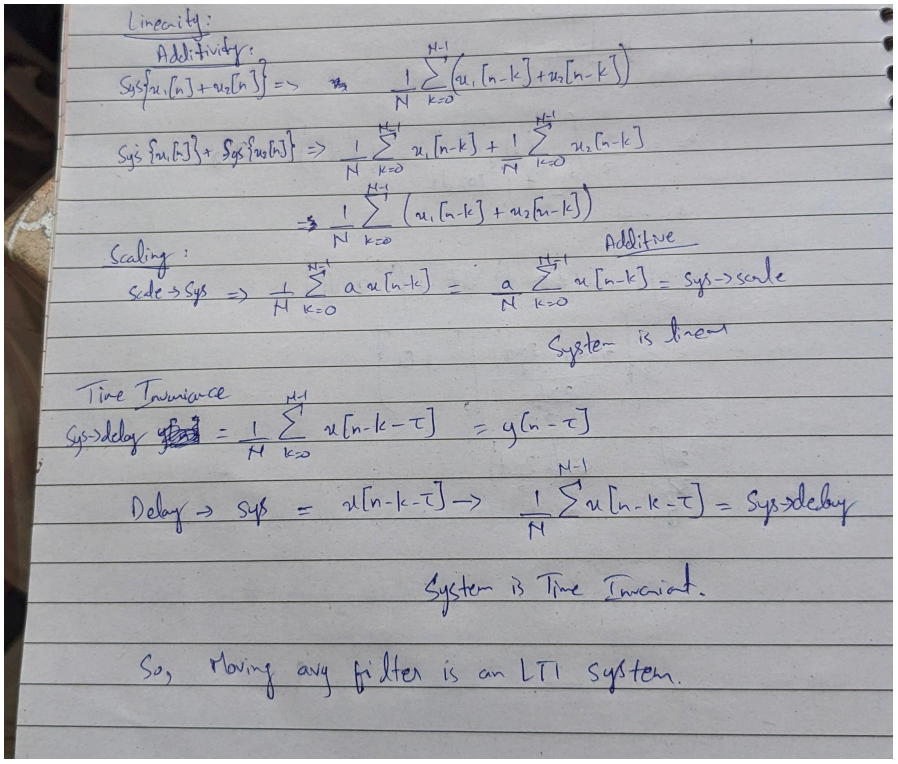
```
subplot 412
plot(n,hn)
title('hn')
subplot 413
plot(n,y1)
title('y1')
subplot 414
plot(y2n,y2)
title('y2')
```

SNAPSHOT:



Task: 04

3) The relation seems to be twice the length of the input -1.
4) I think the output length will then be related to the larger length because one signal is moved over the other signal, so the overlapping length will be related to the length of the larger input.

- 

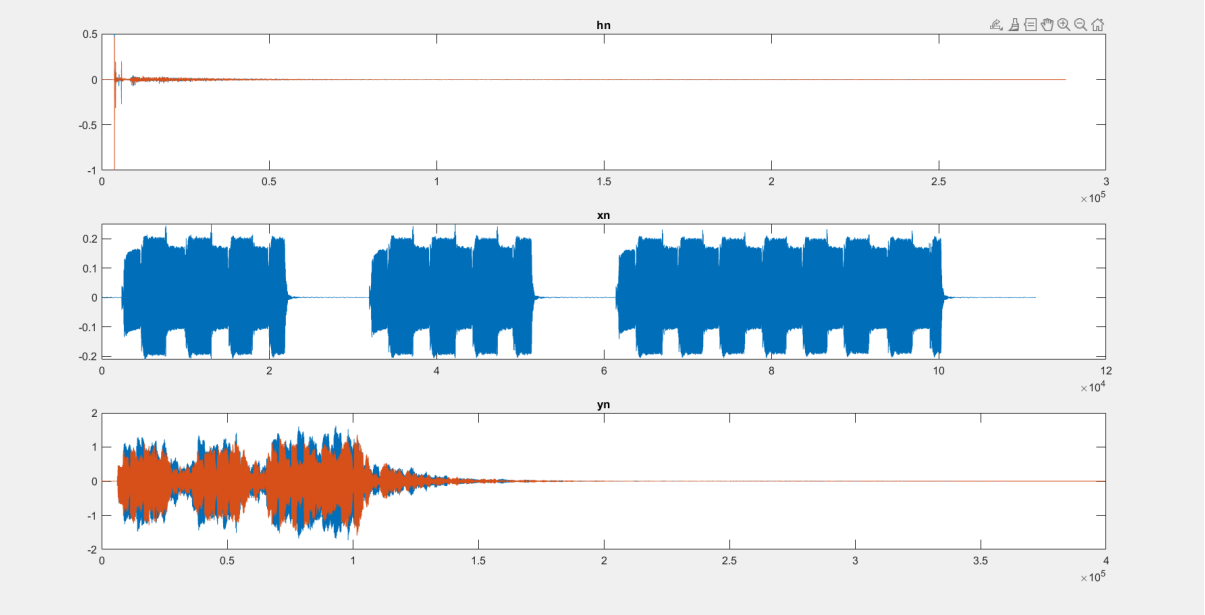
TASK:05

CODE:

```
%% Task 05
clc
clear all
close all
[h,Fh] = audioread("1st_baptist_nashville_balcony.wav");
disp(size(h));
% soundsc(h,Fh)
% pause(length(h)/Fh)
[x,Fx] = audioread("ringtone.wav")
disp(size(x));
% soundsc(x,Fx)
```

```
% pause(length(x)/Fx)
%Extracting the first and second channel from the audio
%in simple words h1 will have all rows of first column and h2 will have all
%rows of second column
h1 = h(:,1);
h2 = h(:,2);
y1 = conv(h1,x);
y2 = conv(h2,x);
y = [y1,y2];
% soundsc(y,Fx)
% pause(length(y)/Fx)
subplot 311
plot(h)
title('hn')
subplot 312
plot(x)
title('xn')
subplot 313
plot(y)
title('yn')
```

SNAPSHOT:



Task: 06

CODE:

```
function [yn]=discrete_convo(xn,hm,n,m)
% hm = [1, 2, -1];
% m=3;
% xn = [4,1,2,5];
% n = 4;
%flip hm and make size of the arrays same
hm = flip(hm);
while n<m
```

```
    xn = [xn 0];
    n=n+1;
end
while m<n
    hm = [hm 0];
    m=m+1;
end
yn = [];  %resultant matrix
sizeyn = 0; %size of resultant(initially set to 0)
%copy matrices to be later used in left loop
lxn = xn
ln = n
lhm=hm
lm=m
% right loop
for i = 0:n
    %sum of the inner product of the two matrices
    val = sum(xn.*hm);
    yn = [yn val]; %append the resultant
    sizeyn = sizeyn+1;
    hm = [0 hm];
    m= m+1;
    xn = [xn 0];
    n=n+1;
end
% left loop
for i = 0: lm-2
    %append 0s too shift left hm
    lhm = [lhm 0]
    lm= lm+1;
    lxn = [0 lxn]
    ln=ln+1;
    %sum of the inner product of the two matrices
    val = sum(lxn.*lhm);
    yn = [val yn]; %append the resultant
    sizeyn = sizeyn+1; %increase the size

end
yn
```

SNAPSHOT: