Problem 1:

Matlab function "circonv" is included at the end of this report under Appendix A

```
circonv(x,h,N);
```

This functions has 3 inputs:

1- x, the input signal

2- h, the impulse response of the filter

3- N, the length of the circular convolution

Also, input "N" can be used to alter the function to linear convolution. Since length of circular convolution cannot be zero, if N is set to zero, my "circonv" function will do a circular convolution of length M+P-1 (M is length of x and P is length of h), which is equivalent to linear convolution.

Problem 2:

a)

Stem plot of the iutput is illustrated in Figure 1.

```
xa = [1 1 1 1];
ha = [1 2 3];
N=4;
ya = circonv(xa,ha,N);

disp('circonv(x,h,N)')
```

```
circonv(x,h,N)
```

```
disp(['x = [' num2str(xa) ']'])
```

```
x = [1  1  1  1]
```

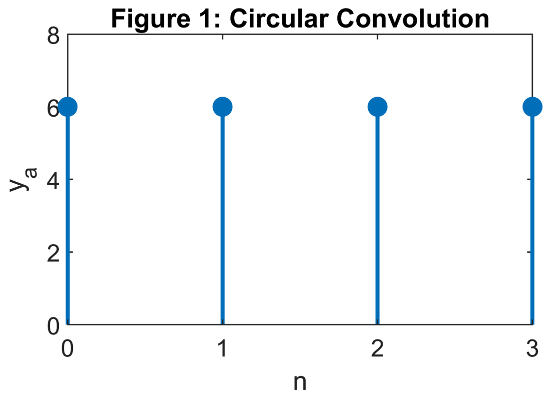
```
disp(['h = [' num2str(ha) ']'])
```

```
h = [1  2  3]
```

```
disp(['N = ' num2str(N)])
```

```
N = 4
```

```
figure
stem((1:4)-1,ya,'filled','LineWidth',1.5)
ylim([0,8])
xlabel('n')
ylabel('y_a')
set(gcf,'position',[50,50,300,200])
title('Figure 1: Circular Convolution')
```



b)

Stem plot of the iutput is illustrated in Figure 2.

```
xb = [1 1 1 1];
hb = [1 2 3];
yb = circonv(xb,hb,0);

disp('circonv(x,h,N)')
```

```
circonv(x,h,N)
```

```
disp(['x = [' num2str(xb) ']'])
```

```
x = [1  1  1  1]
```

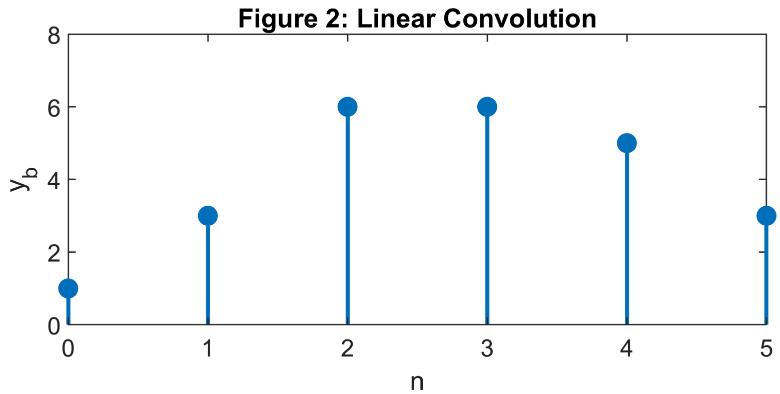
```
disp(['h = [' num2str(hb) ']'])
```

```
h = [1  2  3]
```

```
disp('N = 0' )
```

```
N = 0
```

```
figure
stem((1:6)-1,yb,'filled','LineWidth',1.5)
ylim([0,8])
xlabel('n')
ylabel('y_b')
set(gcf,'position',[50,50,450,200])
title('Figure 2: Linear Convolution')
```



c)

Stem plot of the iutput is illustrated in Figure 3.

```
xc = [1 3 6 2 8];
hc = [2 9 5 3];
N=6;
yc = circonv(xc,hc,N);

disp('circonv(x,h,N)')
```

```
circonv(x,h,N)
```

```
disp(['x = [' num2str(xc) ']'])
```

```
x = [1   3   6   2   8]
```

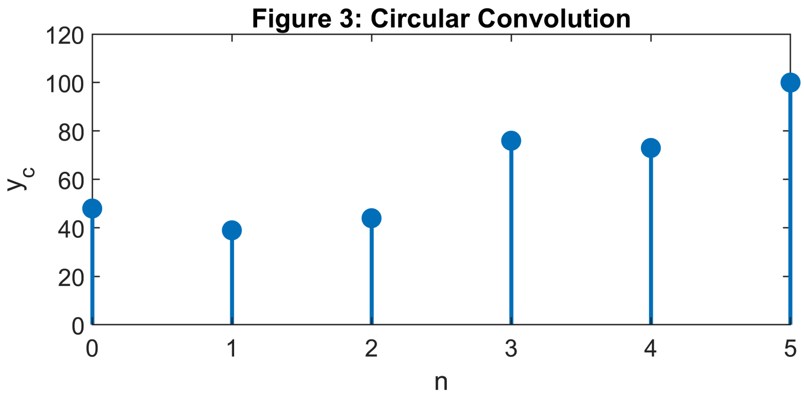
```
disp(['h = [' num2str(hc) ']'])
```

```
h = [2   9   5   3]
```

```
disp(['N = ' num2str(N)])
```

```
N = 6
```

```
figure
stem((1:6)-1,yc,'filled','LineWidth',1.5)
ylim([0,120])
xlabel('n')
ylabel('y_c')
set(gcf,'position',[50,50,450,200])
title('Figure 3: Circular Convolution')
```



Problem 3:

Matlab function "ola" is included at the end of this report under Appendix B.

```
load('MP2data')
N=100;
ola(x,h,N);
```

This functions has 3 inputs:

1- x, the input signal

2- h, the impulse response of the filter

3- N, the length of the DFT

Length of the blocks are internally calculated as L=N+1-P, in which P is the length of impulse response.

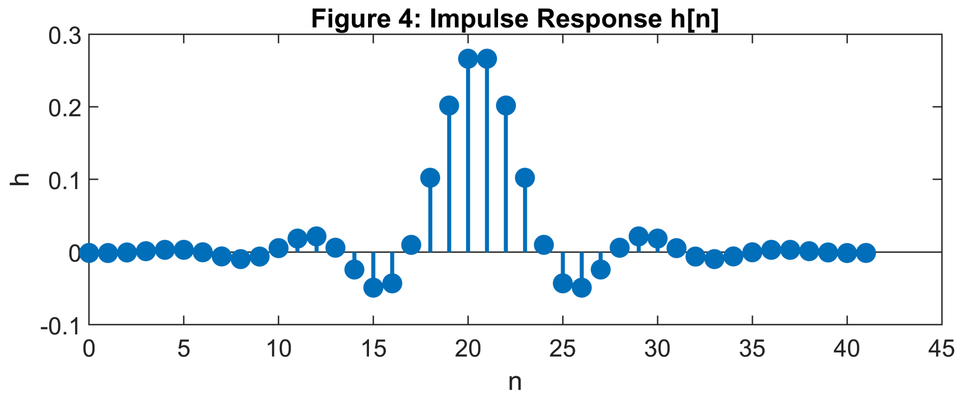
Problem 4:

a)

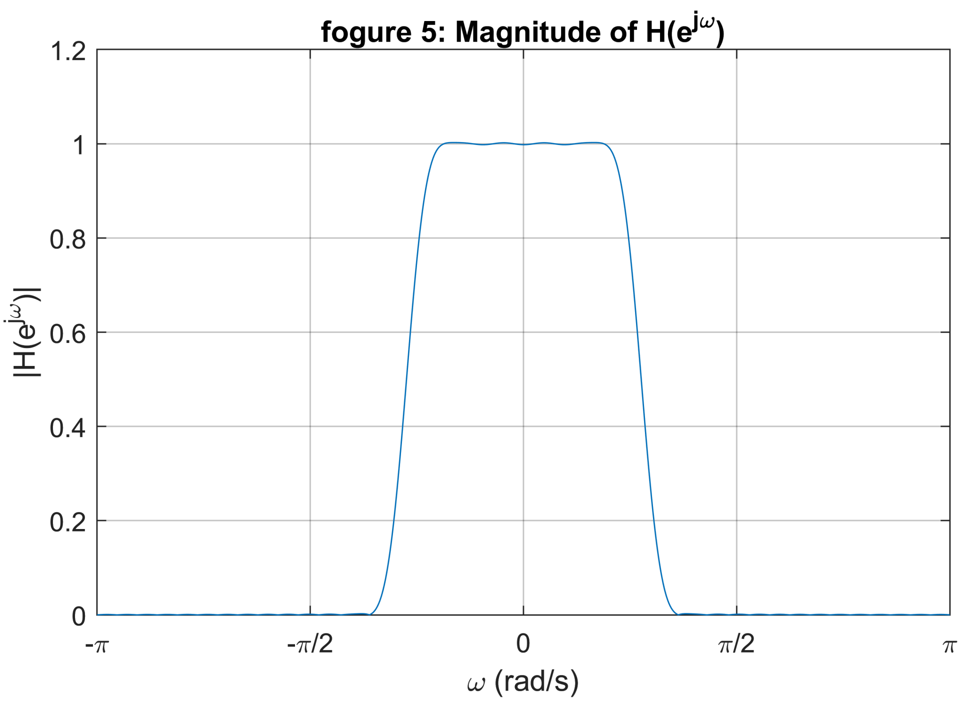
h[n] is a delayed sinc function (figure 4), as a result it approximates a lowpass filter (figure 5).

```
load('MP2data')

figure
stem((1:42)-1,h,'filled','LineWidth',1.5)
ylim([-0.1,0.3])
xlim([0 45])
xlabel('n')
ylabel('h')
set(gcf,'position',[50,50,550,200])
title('Figure 4: Impulse Response h[n]')
```




```
H = fft(h,1024);
figure,
plot(linspace(-pi,pi,1024),fftshift(abs(H)))
xlim([-pi,pi])
grid on
xticks([-pi -pi/2 0 pi/2 pi])
set(gca,'XTickLabel',{'-\pi','-\pi/2','0','\pi/2','\pi'})
xlabel('\omega (rad/s)')
ylabel('|H(e^j^\omega)|')
title('fogure 5: Magnitude of H(e^j^\omega)')
ylim([0,1.2])
set(gcf,'position',[50,50,550,350])
```



b)

h[n] is a de

c)

```
y_ola=ola(x,h,100);
y_conv=conv(x,h);

error = mean((y_ola-y_conv).^2);
disp(['Average error between "conv" and "ola": ' num2str(error)])
```

```
Average error between "conv" and "ola": 6.7514e-33
```

Small error confirm that "ola" is working as intended.

Appendix A:

```
function y=circonv(x,h,N)

if N == 0
    N = length(x)+length(h)-1;
end

X = fft(x,N);
H = fft(h,N);
Y = X.*H;
y = ifft(Y,N);

end
```

Appendix B:

```
function y=ola(x,h,N)

M = length(x);
P = length(h);
L = N+1-P;
r = mod(M,L);
if r ~= 0
    x = [x zeros(1,L-r)];
    M = length(x);
end
y = zeros(1,M+P-1);

H = fft(h,L+P-1);
i = 1;
while i<=M
    Xr = fft(x(i:i+L-1),L+P-1);
    Yr = H.*Xr;
    yr = ifft(Yr,L+P-1);
    y(i:i+L+P-2) = y(i:i+L+P-2)+yr;
    i=i+L;
end
if r ~= 0
    y(end-(L-r-1):end) = [];
end
end
```