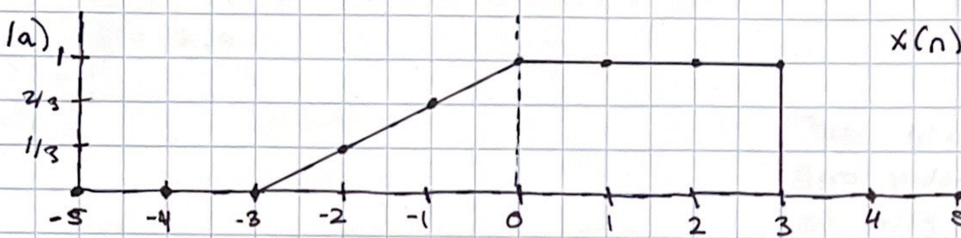
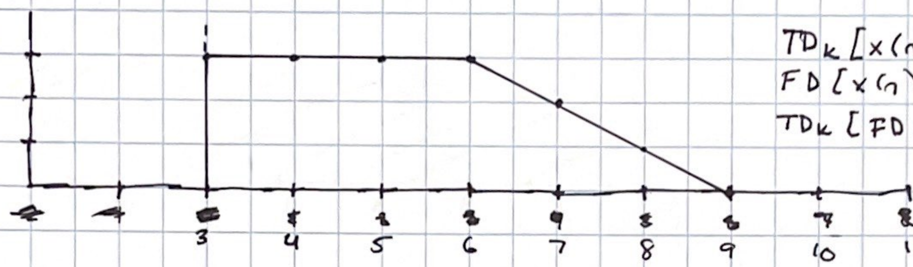
DSP Module 2 Hw probl

1a)



$$x(n) = \begin{cases} 1 + \frac{n}{3} & -3 \le n \le 1 \\ 1 & 0 \le n \le 3 \\ 0 & \text{else} \end{cases}$$

bi)



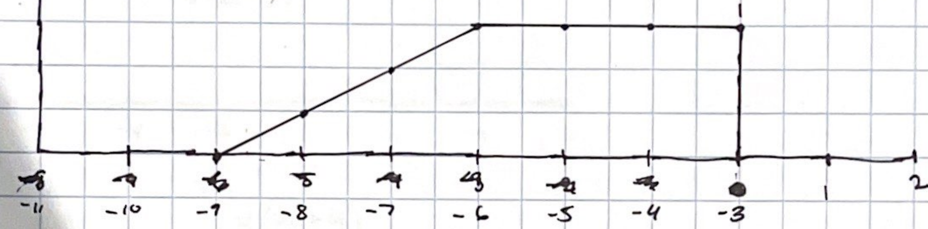
$TD_k[x(n)] = x(n-k)$

$FD[x(n)] = x(-n)$

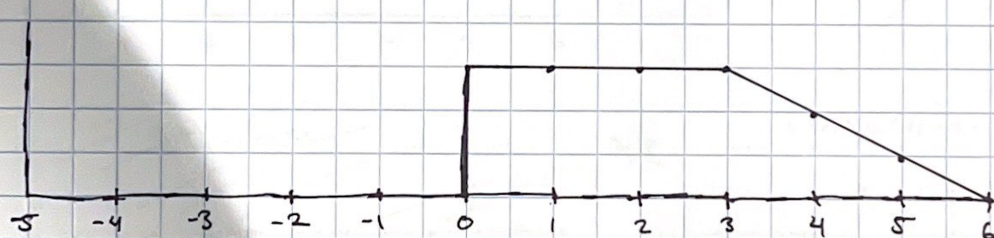
$TD_k[FD[x(n)]] = x(-n+k)$

$FD[TD_k[x(n)]] = x(-n-k)$

ii)



c) $x(-n+3) = x(-(n-3))$



D) First expess $x(-n+k)$ as $x(-(n-k))$. Then work from the outter operatons in. Flip the signal about $n=0$ then shift k samples (right if the original was $x(-n+k)$).

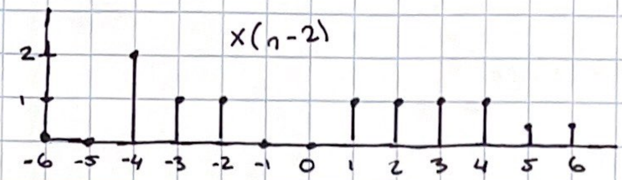
e) $x(n) = \dfrac{\delta(n+2)}{3} + \dfrac{2\cdot\delta(n+1)}{3} + u(n) - u(n-4)$

DSP HW2 #3

3. $x[n] = [2,1,1,0,0,1,1,1,1,.5,.5,0,0]$

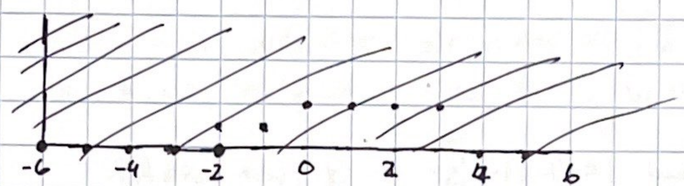
$n = -6:6$

a)

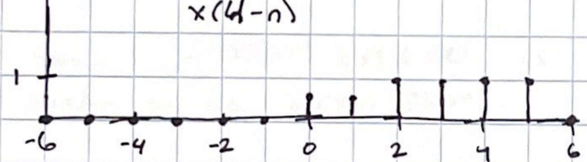


These graph's depend on Zero padding when shifts, not shifty the vales back in

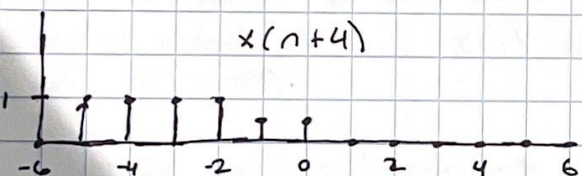
b)



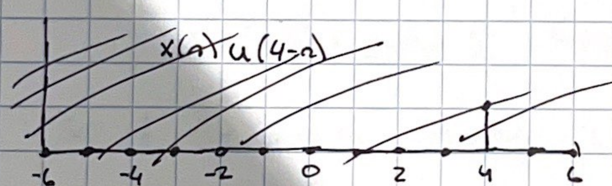
b)

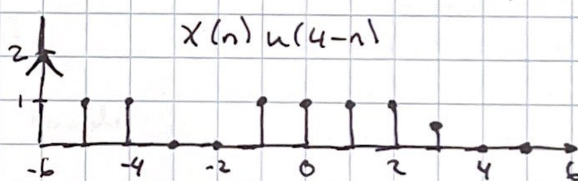


c)

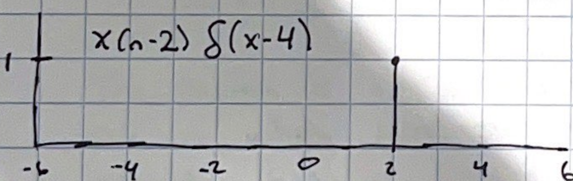


d)

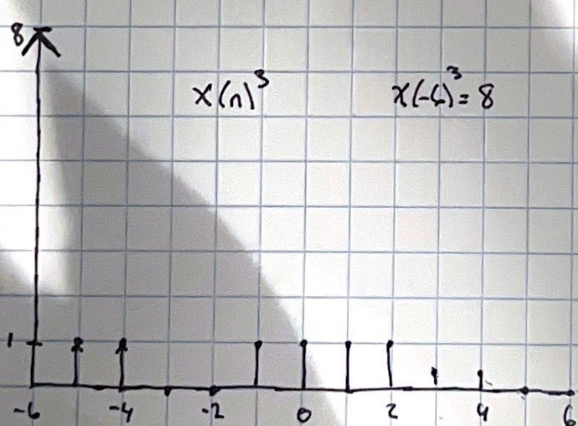




e)

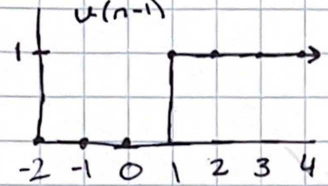


f)



DSP HW2

4b) Show $u(n-1) = \sum_{k=-\infty}^{\infty} \delta(k-1)$



$\sum_{k=-\infty}^{0} \delta(k-1) = 0$ $\quad \sum_{k=2}^{\infty} \delta(k-1) = 0$ $\quad \delta(k-1)\Big|_{k=1} = 1$

After $n=1$ $u(n-1) = 1$ for $n \to \infty$. If you sum over $\delta(k-1)$ form $-\infty$ to ∞, only one value $k=1$ will result in a non zero value.

$\therefore$ After any $k > 1$ $\sum_{k=-\infty}^{\infty} \delta(k-1) = 1$ and any $k > 1$ of $u(n-1) = 1$

5) Show $\sum_n |h(n)| \leq M < \infty$ is sufficent + necessary for a LTI system to be BIBO stabe.

- For a system to be stable every bounded input must produce a bounded output so if $x(n)$ is BIBO stable there is a maxim value $M_x < \infty$.

$\rightarrow$ If input is bounded $\rightarrow$ $|x(n) \leq M_x|$

$\therefore$ $|y(n)| \leq M_x \sum_k |h(k)|$

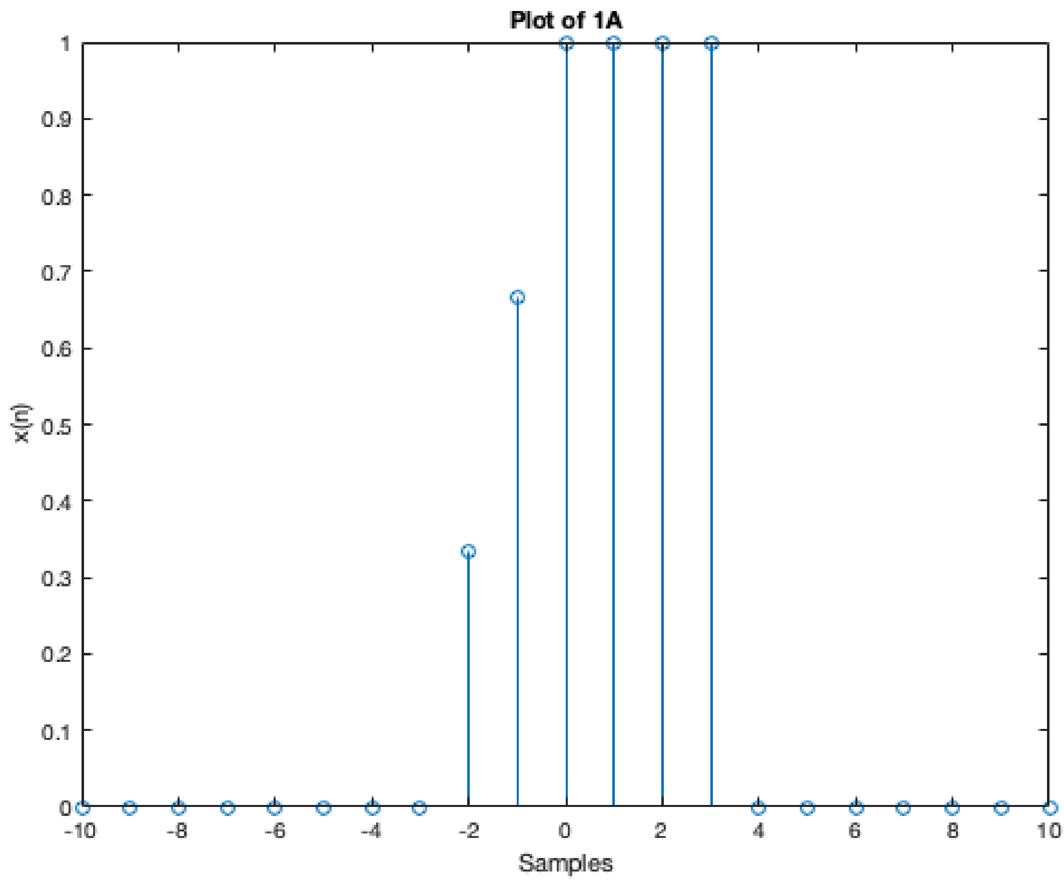
- So if the impulse response is bounded

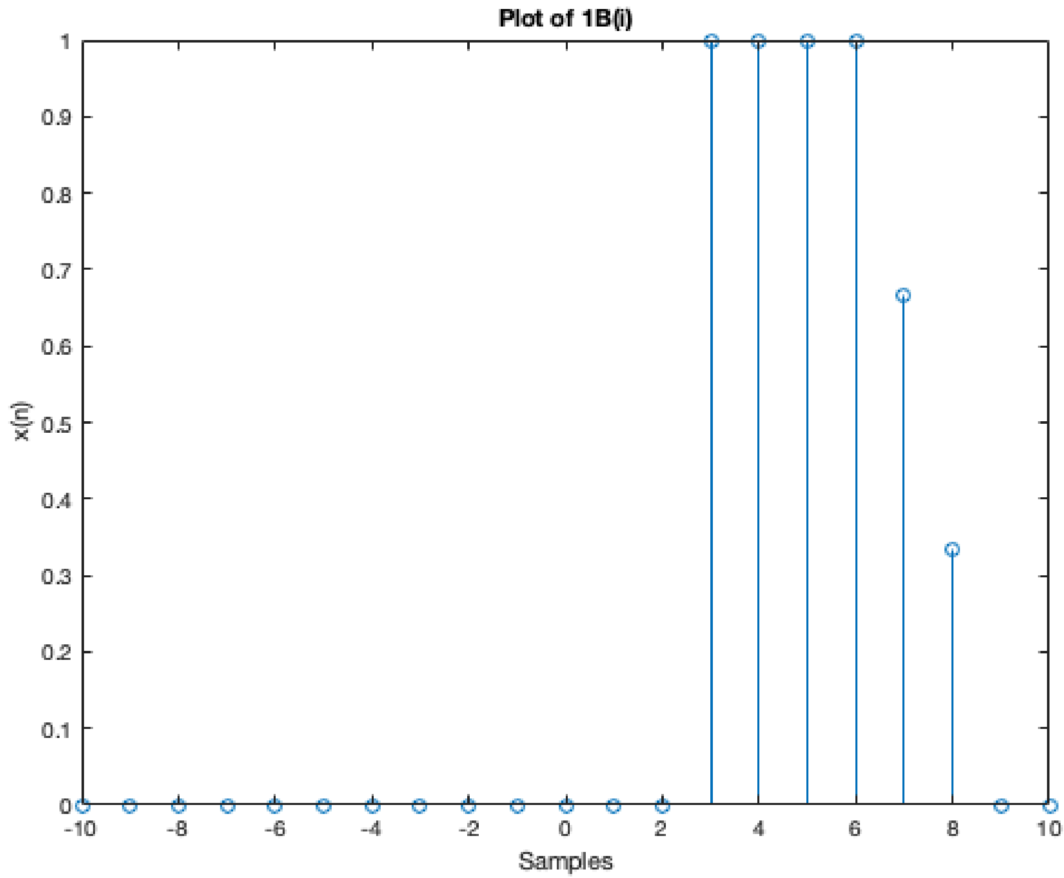
ie) $\sum_k |h(k)| < \infty$

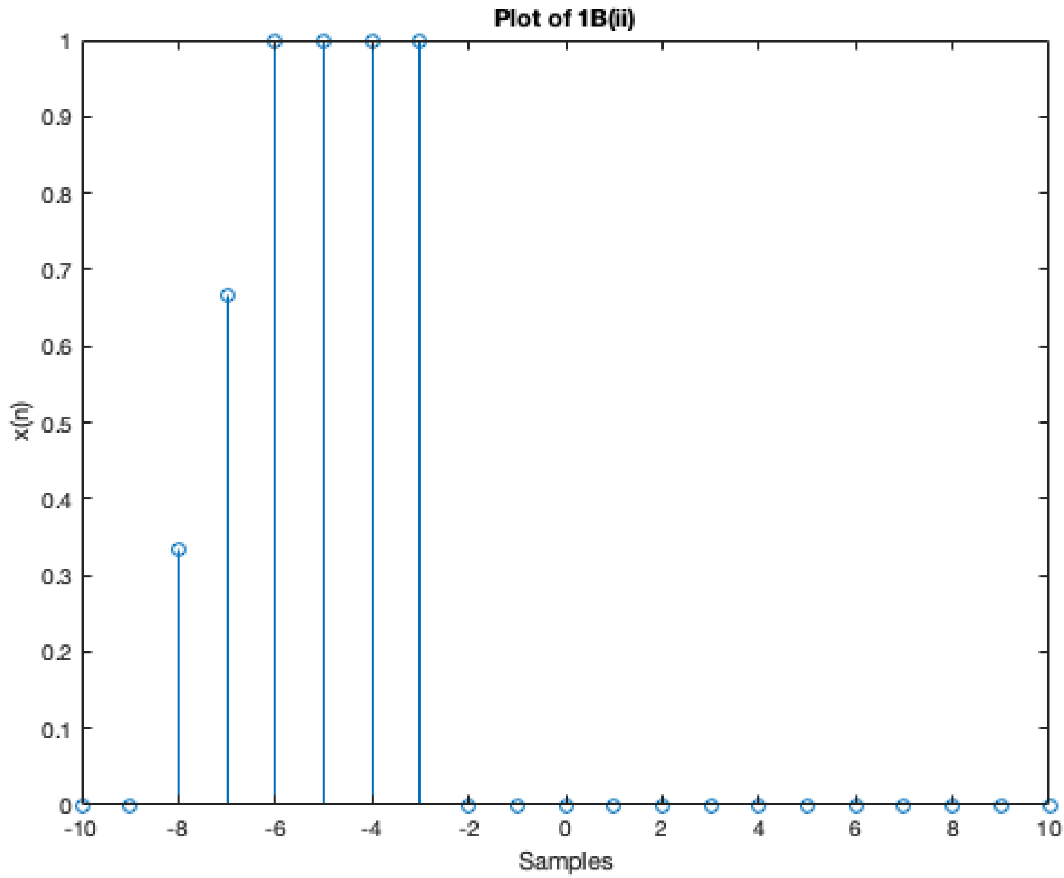
The system is stable. This is necessary since if there is one value n where $h(n) \rightarrow \infty$ the system is unstable. This is absolutely summable so the system must decay.

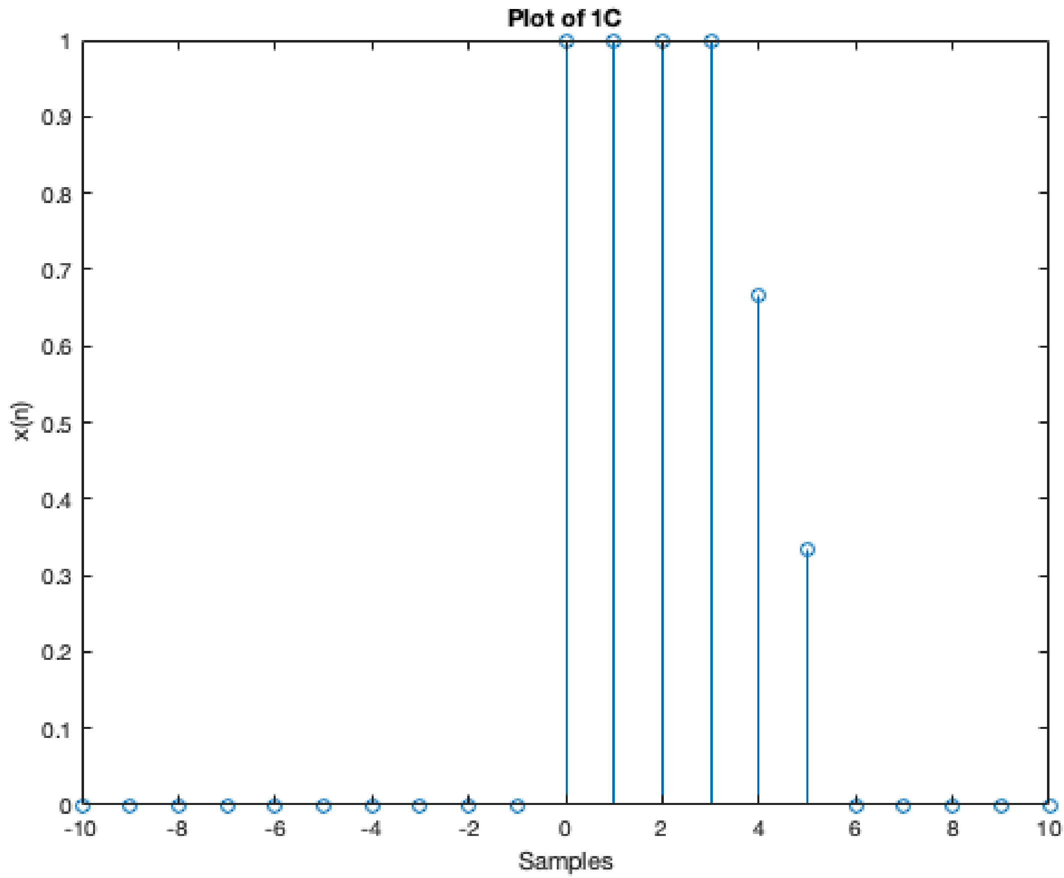
```
% DSP HW2 #2
clear; close all;

% Variables
n = linspace(-10, 10, 21);
indexs = 8:14;
delay = 6;

% Function
x_n = zeros(21, 1);               % Create array to hold x(n)
x_n(8:10) = 1 + n(8:10) ./ 3;    % Solve for -3 le n le -1
x_n(11:14) = 1;                   % Solve for 0 le n le 3

% Plot
figure(1)
stem(n, x_n)
xlabel('Samples')
ylabel('x(n)')
title('Plot of 1A')

% 2B(i)
x_n_b1_pre_delay = flip(x_n);                        % Flip the array first
x_n_b1 = zeros(21, 1);
x_n_b1(indexs+delay) = x_n_b1_pre_delay(indexs);    % Delay values

figure(2)
stem(n, x_n_b1)
xlabel('Samples')
ylabel('x(n)')
title('Plot of 1B(i)')

% 2B(ii)
x_n_b2 = zeros(21, 1);
x_n_b2(indexs+delay) = x_n_b1_pre_delay(indexs);    % Delay values
x_n_b2 = flip(x_n_b2);                              % Flip the array first

figure(3)
stem(n, x_n_b2)
xlabel('Samples')
ylabel('x(n)')
title('Plot of 1B(ii)')

% 2C
x_n_c_pre_delay = flip(x_n);                        % Flip the array first
x_n_c = zeros(21, 1);
delay = 3;
x_n_c(indexs+delay) = x_n_c_pre_delay(indexs);      % Delay values

figure(4)
stem(n, x_n_c)
xlabel('Samples')
ylabel('x(n)')
```

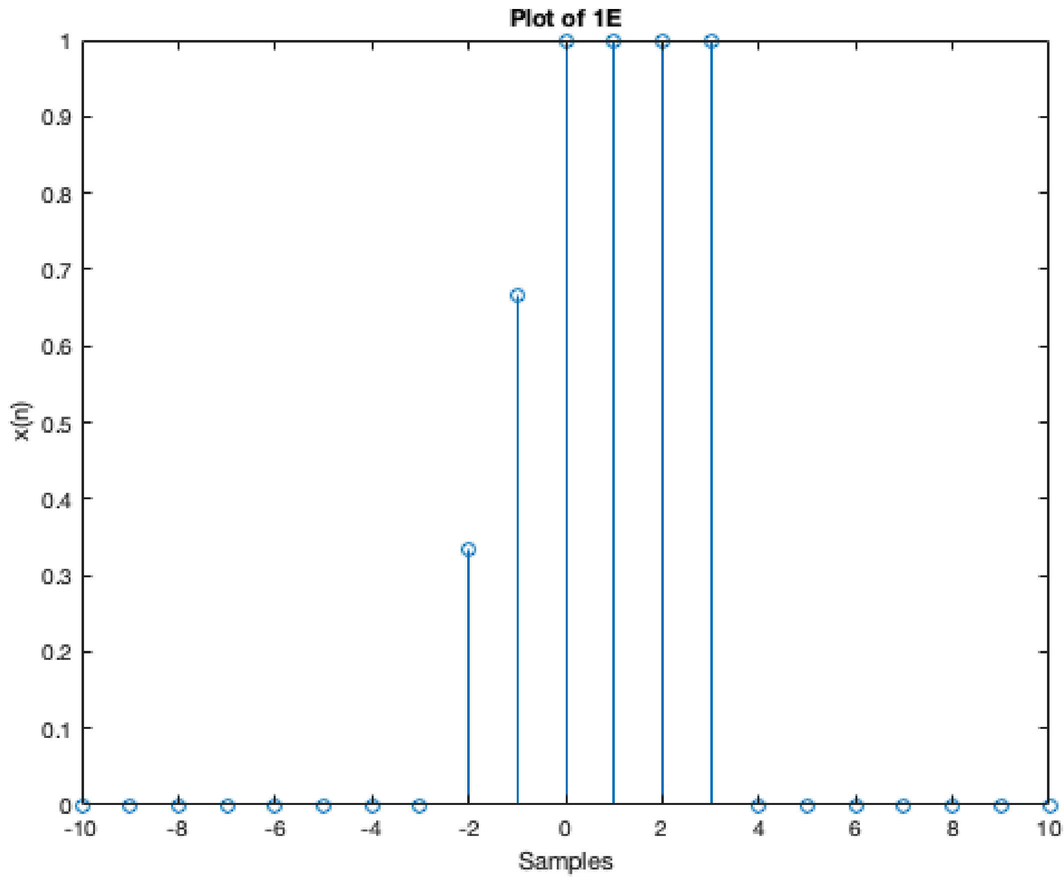
```
title('Plot of 1C')

% 2E
% Use impulse and unit_step that are derived from DSP m-file Help Manual
x_n = impulse(-2,-10, 10) ./3 + 2 .* impulse(-1,-10,10) ./ 3 + ...
                unit_step(0,-10,10) - unit_step(4, -10, 10);

% Plot
figure(5)
stem(n, x_n)
xlabel('Samples')
ylabel('x(n)')
title('Plot of 1E')
```




Plot of 1B(i)
x(n)
Samples
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
-10
-8
-6
-4
-2
0
2
4
6
8
10

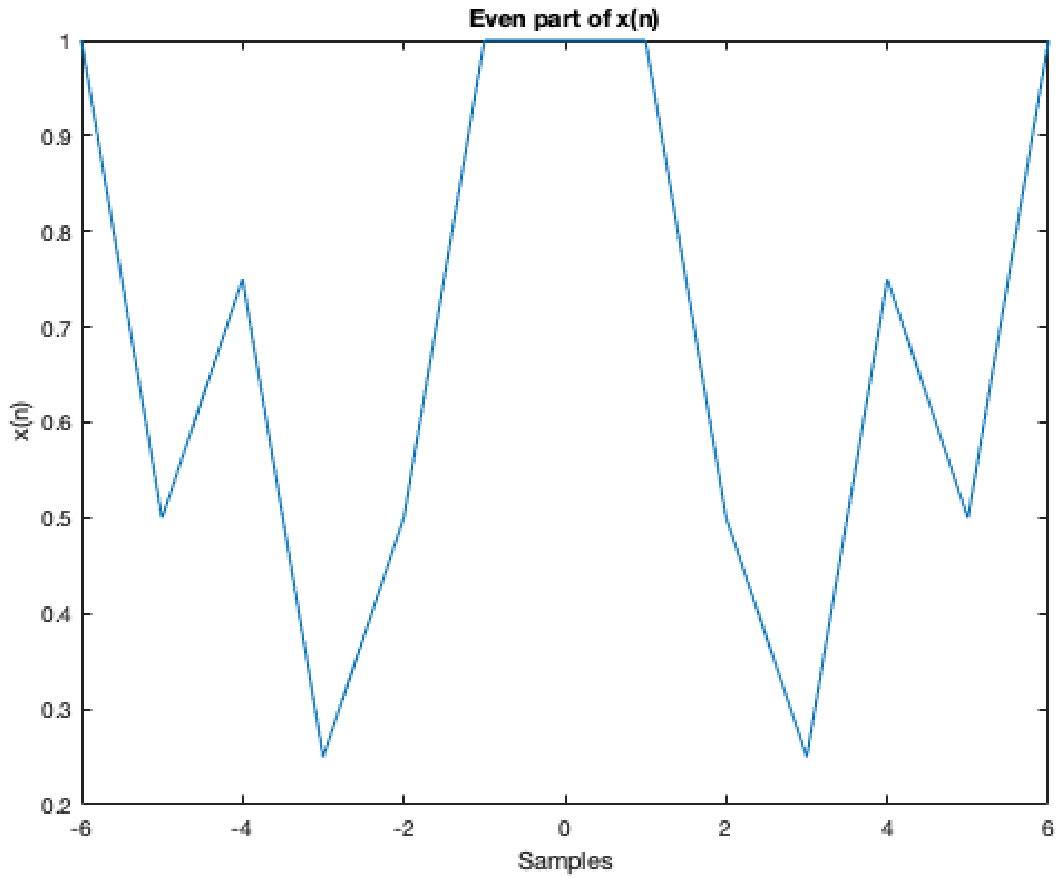

Plot of 1B(ii)
x(n)
Samples
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
-10
-8
-6
-4
-2
0
2
4
6
8
10

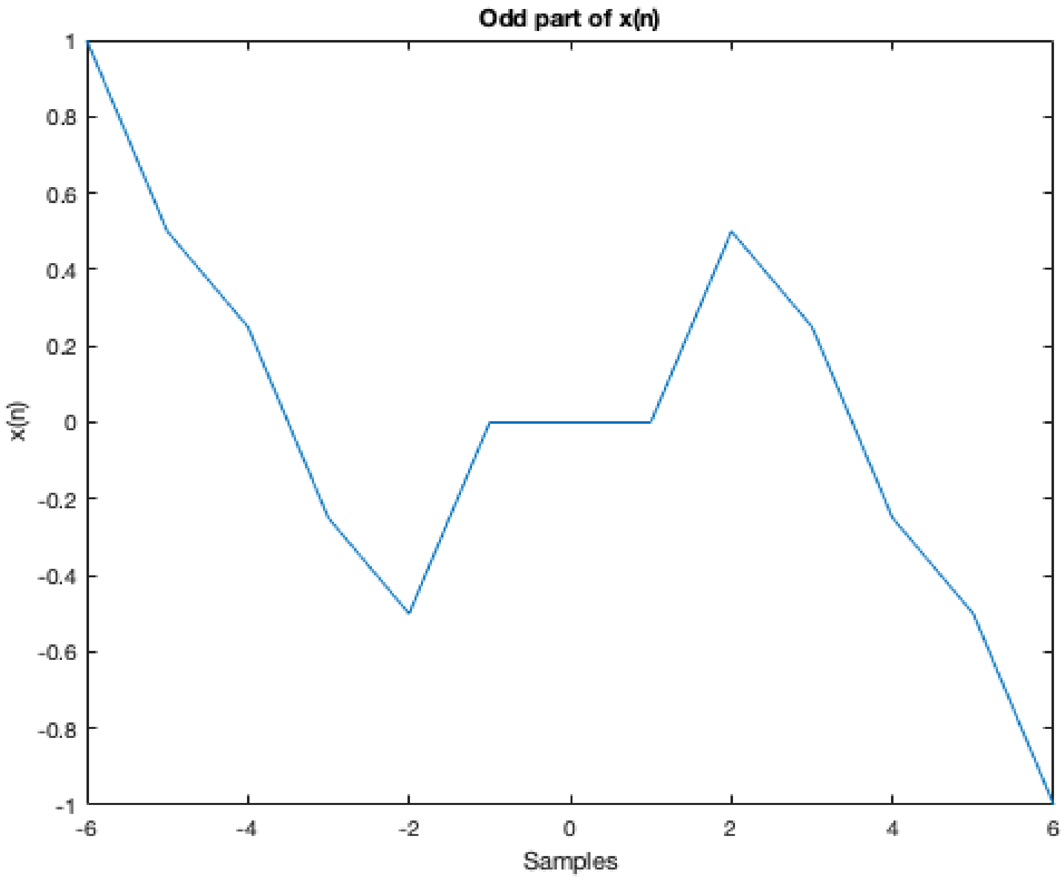

Plot of 1C
x(n)
Samples
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
-10
-8
-6
-4
-2
0
2
4
6
8
10

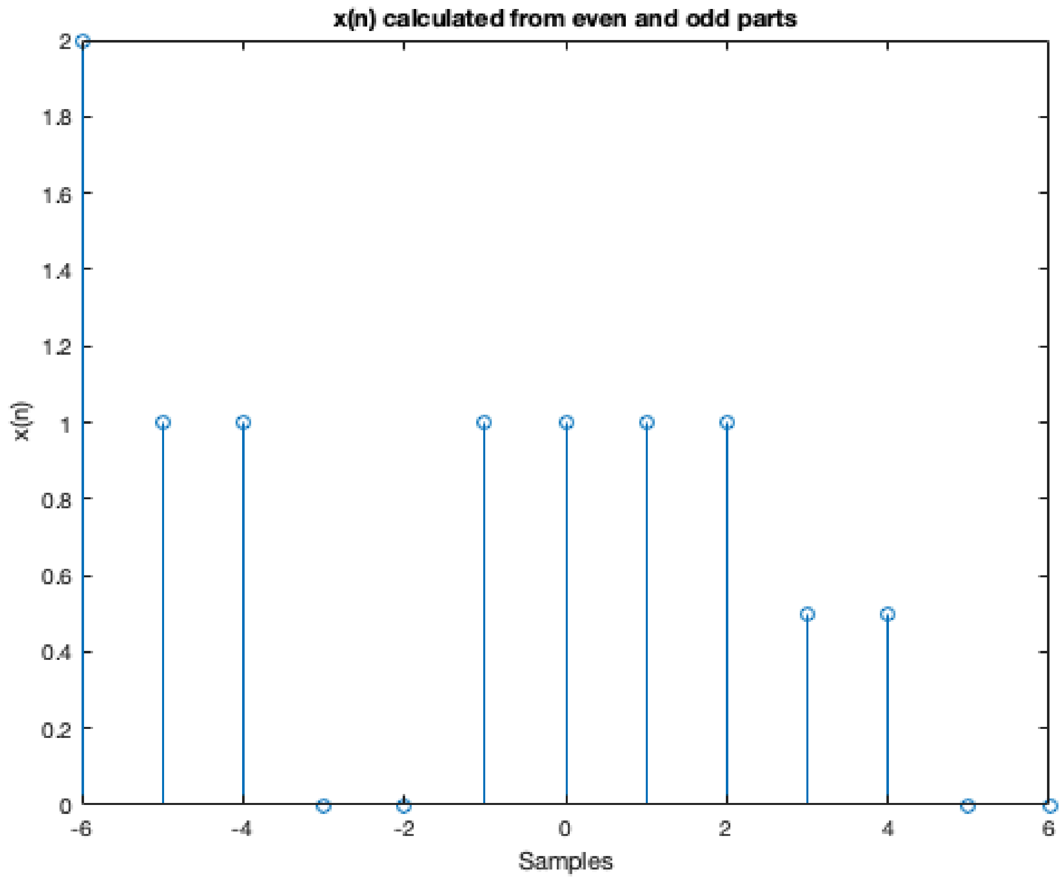


Published with MATLAB® R2022a

```
% DSP HW2 #3

% Enviorment
n = -6:6;
x = [2, 1, 1, 0, 0, 1, 1, 1, 1, .5, .5, 0, 0];

% Part G, calc + plot even part of x(n)
% x_even = .5[x(n) + x(-n)]
x_even = .5 .* (x + flip(x));

% Part H, calc + plot odd part of x(n)
% x_odd = .5[x(n) - x(-n)]
x_odd = .5 .* (x - flip(x));

% Plot even and odd singals
figure(1);
plot(n, x_even)
xlabel('Samples')
ylabel('x(n)')
title('Even part of x(n)')
figure(2)
plot(n, x_odd)
xlabel('Samples')
ylabel('x(n)')
title('Odd part of x(n)')

% Part I, calculate x(n) from even and odd
% x(n) = x_even + x_odd
x_calc = x_even + x_odd;

% Plot x(n) calculated from even and odd parts
figure(3);
stem(n, x_calc)
xlabel('Samples')
ylabel('x(n)')
title('x(n) calculated from even and odd parts')
```

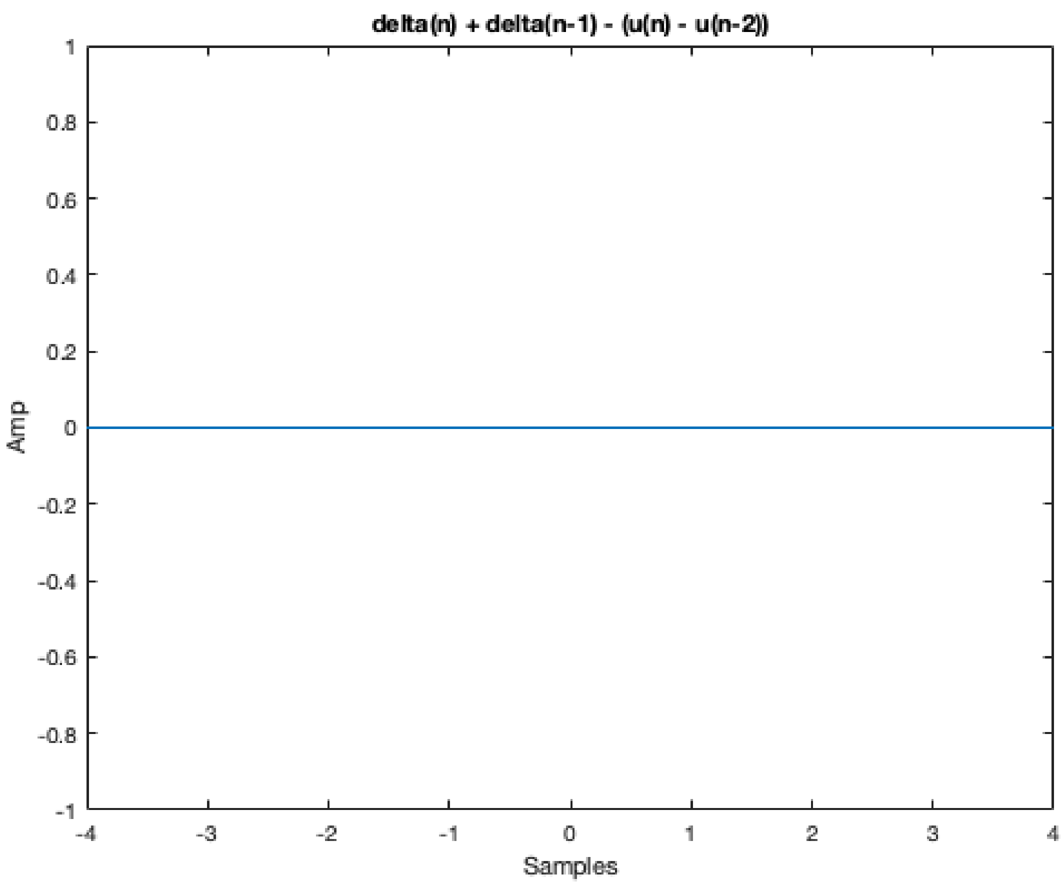

Even part of x(n)
x(n)
Samples
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
-6
-4
-2
0
2
4
6


Odd part of x(n)
x(n)
Samples
1
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1
-6
-4
-2
0
2
4
6



Published with MATLAB® R2022a

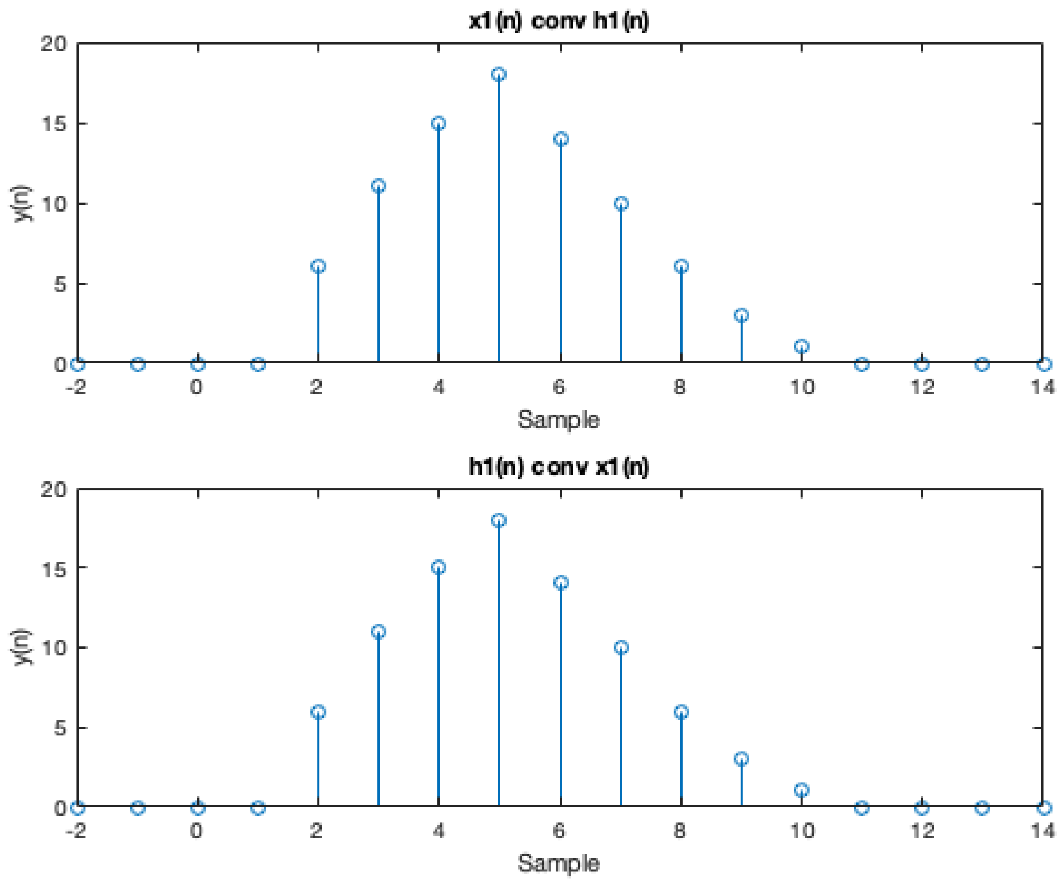
```
% DSP HW2 #4
% Show delta(n) + delta(n-1) = u(n) - u(n-2)

% Calculate delta equation
delta_eqn = impulse(0, -4, 4) + impulse(1, -4, 4);

% Calculate unit step equation
step_eqn = unit_step(0, -4, 4) - unit_step(2, -4, 4);

% Show they are equal
equal = delta_eqn - step_eqn;

figure(1)
plot(-4:4, equal)
xlabel('Samples')
ylabel('Amp')
title('delta(n) + delta(n-1) - (u(n) - u(n-2))')
```

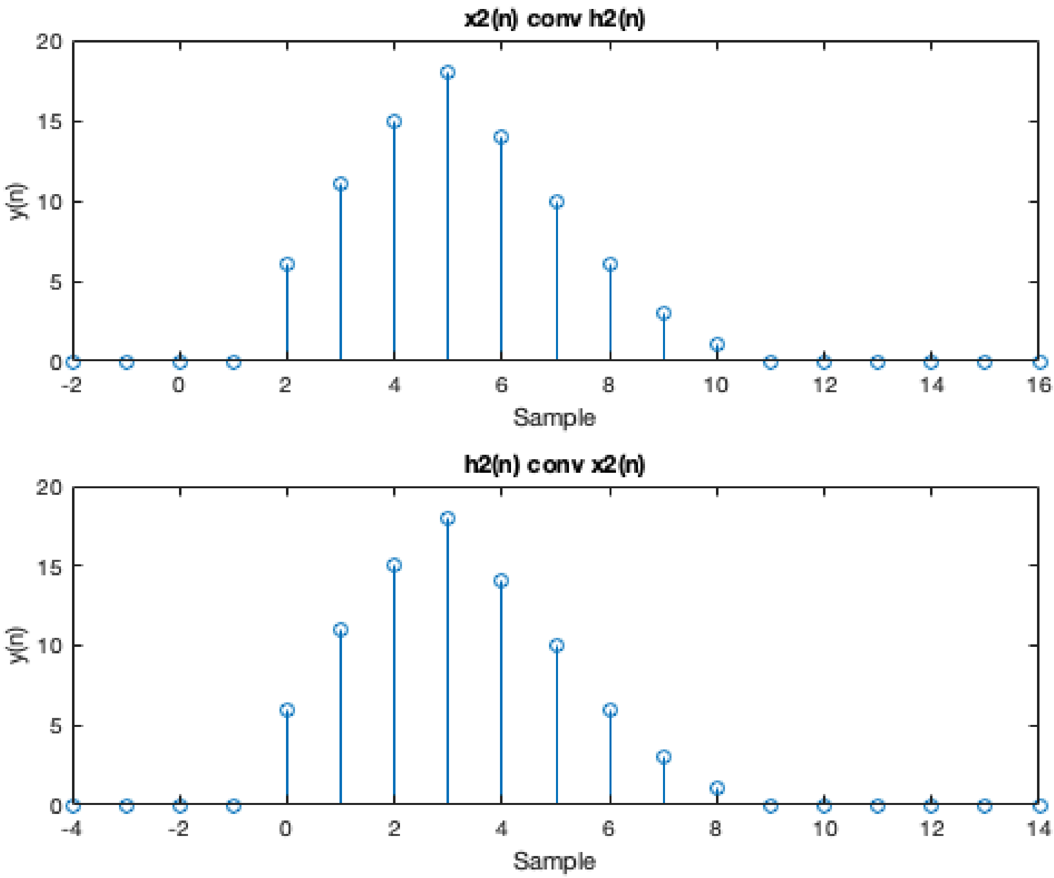


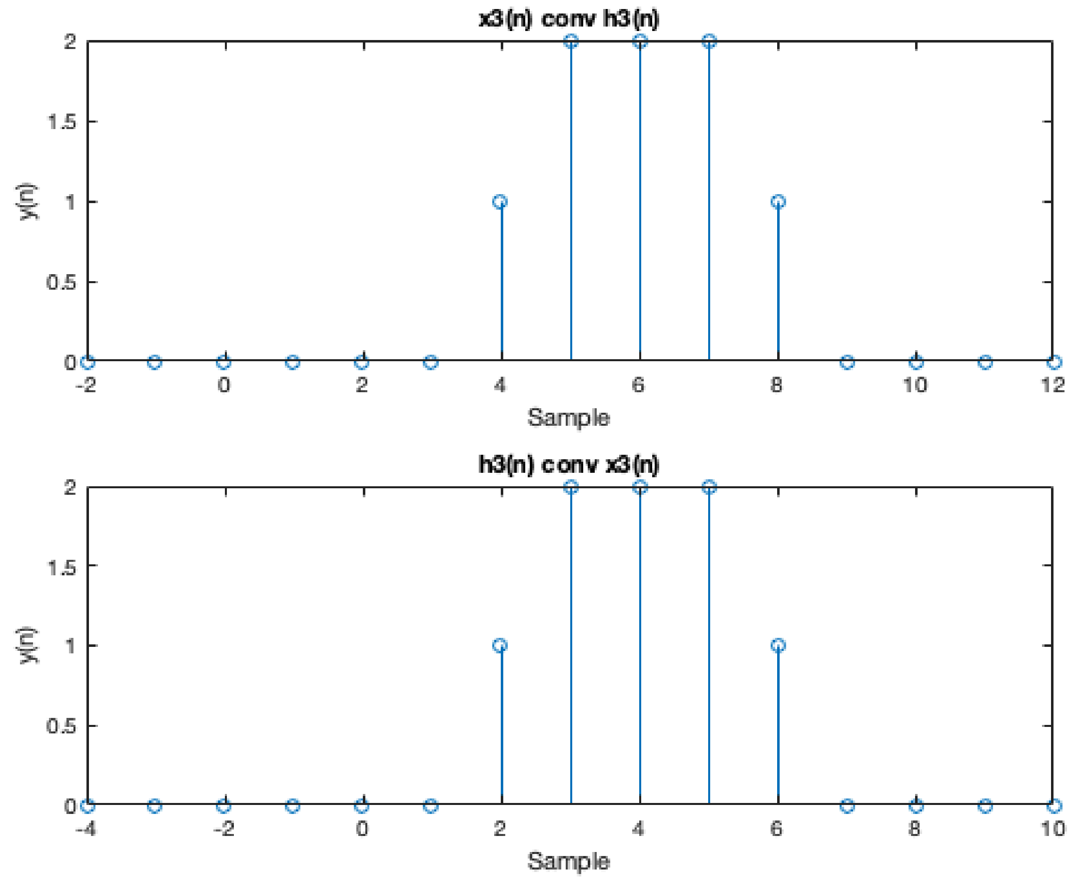
Published with MATLAB® R2022a

```
% DSP HW2 #6
% Show x(n) conv h(n) == h(n) conv x(n)

% Enviorment
x1 = [0, 0, 1, 1, 1, 1, 0];
h1 = [0, 0, 6, 5, 4, 3, 2, 1, 0, 0, 0];
x2 = [0, 0, 1, 1, 1, 1, 0, 0];
h2 = [0, 0, 6, 5, 4, 3, 2, 1, 0, 0, 0, 0];
x3 = [0, 0, 0, 0, 0, 1, 1, 1, 1];
h3 = [0, 1, 1, 0, 0, 0, 0];

% Compute Convolutions
y1_a = conv(x1, h1);
y1_b = conv(h1, x1);
y2_a = conv(x2, h2);
y2_b = conv(h2, x2);
y3_a = conv(x3, h3);
y3_b = conv(h3, x3);

% Plot response of system
figure(1)
subplot(2,1,1)
start_index = -2;
n = start_index:start_index+(length(x1)+length(h1)-2);
stem(n, y1_a)
title('x1(n) conv h1(n)')
xlabel('Sample')
ylabel('y(n)')
subplot(2,1,2)
stem(n, y1_b)
title('h1(n) conv x1(n)')
xlabel('Sample')
ylabel('y(n)')

figure(2)
subplot(2,1,1)
start_index = -2;
n = start_index:start_index+(length(x2)+length(h2)-2);
stem(n, y2_a)
title('x2(n) conv h2(n)')
xlabel('Sample')
ylabel('y(n)')
subplot(2,1,2)
start_index = -4;
n = start_index:start_index+(length(x2)+length(h2)-2);
stem(n, y2_b)
title('h2(n) conv x2(n)')
xlabel('Sample')
ylabel('y(n)')

figure(3)
subplot(2,1,1)
```

```
start_index = -2;
n = start_index:start_index+(length(x3)+length(h3)-2);
stem(n, y3_a)
title('x3(n) conv h3(n)')
xlabel('Sample')
ylabel('y(n)')
subplot(2,1,2)
start_index = -4;
n = start_index:start_index+(length(x3)+length(h3)-2);
stem(n, y3_b)
title('h3(n) conv x3(n)')
xlabel('Sample')
ylabel('y(n)')
```




x2(n) conv h2(n)
y(n)
Sample
h2(n) conv x2(n)
y(n)
Sample



Published with MATLAB® R2022a

```
% DSP HW2 #7
% Given h(n) and y(n) find x(n)

% Enviorment
h = [1, .5, .25, 1/8, 1/16, 0, 0, 0, 0];
y = [1, 2, 2.5, 3, 3, 3, 2, 1, 0];
x = [0,0,0,0,0,0,0,0,0];

% Calculate x(n)
x(1) = y(1) / h(1);
x(2) = y(2) - (x(1)*h(2));
x(3) = y(3) - (x(2)*h(2) + x(1)*h(3));
x(4) = y(4) - (x(3)*h(2) + x(2)*h(3) + x(1)*h(4));
x(5) = y(5) - (x(4)*h(2) + x(3)*h(3) + x(2)*h(4) + x(1)*h(5));
x(6) = y(6) - (x(5)*h(2) + x(4)*h(3) + x(3)*h(4) + x(2)*h(5));
x(7) = y(7) - (x(6)*h(2) + x(5)*h(3) + x(4)*h(4) + x(3)*h(5));
x(8) = y(8) - (x(7)*h(2) + x(6)*h(3) + x(5)*h(4) + x(4)*h(5));
x(9) = y(9) - (x(8)*h(2) + x(7)*h(3) + x(6)*h(4) + x(5)*h(5));

% Printout Data
disp('x(n) values')
x
disp('y(n) calculated from x(n) and h(n)')
y_calc = conv(h, x);
y_calc = y_calc(1:9)
```

```
x(n) values

x =

  Columns 1 through 7

    1.0000    1.5000    1.5000    1.7500    1.5000    1.5312    0.5469

  Columns 8 through 9

    0.0469   -0.4453

y(n) calculated from x(n) and h(n)

y_calc =

  Columns 1 through 7

    1.0000    2.0000    2.5000    3.0000    3.0000    3.0000    2.0000

  Columns 8 through 9

    1.0000         0
```

Published with MATLAB® R2022a