

MATLAB CODE

Numeric Part

```
1  clc
2  clear all
3  close all
4
5
6  for q = 1:6
7      if q == 1
8          conv1 = convolution(@h, @s1, 16);
9
10         figure()
11         stem(0:29, conv1(41:70));
12         ax = gca;
13         ax.XAxis.LineWidth = 1.5; %thicken x-axis
14         ax.YAxis.LineWidth = 1.5; %thicken y-axis
15         axh.XAxisLocation = 'origin';
16         axh.YAxisLocation = 'origin';
17         xlabel('n', 'FontSize', 12, 'FontWeight', 'bold');
18         ylabel(['y', num2str(q), '[n]'], 'FontSize', 12, 'FontWeight', 'bold');
19         grid on
20         axis tight
21         title(['Graph of y', num2str(q), '[n]'], 'FontSize', 14, 'FontWeight', 'bold');
22
23         disp(['y', num2str(q), '[n]: ', num2str(conv1(41:70))]);
24
25
26     elseif q == 2
27         conv2 = convolution(@h, @s2, 16);
28
29         figure()
30         stem(0:29, conv2(41:70));
31         ax = gca;
32         ax.XAxis.LineWidth = 1.5; %thicken x-axis
33         ax.YAxis.LineWidth = 1.5; %thicken y-axis
34         axh.XAxisLocation = 'origin';
35         axh.YAxisLocation = 'origin';
36         xlabel('n', 'FontSize', 12, 'FontWeight', 'bold');
37         ylabel(['y', num2str(q), '[n]'], 'FontSize', 12, 'FontWeight', 'bold');
38         grid on
39         axis tight
40         title(['Graph of y', num2str(q), '[n]'], 'FontSize', 14, 'FontWeight', 'bold');
41
42         disp(['y', num2str(q), '[n]: ', num2str(conv2(41:70))]);
43
44
45     elseif q == 3
46         conv3r = convolution(@h,@s3r,16);
47         conv3im = convolution(@h,@s3im,16);
48         figure()
49         subplot(2,1,1);
50         stem(0:29, conv3r(41:70));
51         ax = gca;
52         ax.XAxis.LineWidth = 1.5; %thicken x-axis
53         ax.YAxis.LineWidth = 1.5; %thicken y-axis
54         axh.XAxisLocation = 'origin';
55         axh.YAxisLocation = 'origin';
56         xlabel('n', 'FontSize', 12, 'FontWeight', 'bold');
57         ylabel(['Re(y', num2str(q), '[n])'], 'FontSize', 12, 'FontWeight', 'bold');
58         grid on
```

```

59     axis tight
60     title(['Graph of Re(y', num2str(q), '[n]'), 'FontSize', 14, 'FontWeight', 'bold']);
61
62     subplot(2,1,2);
63     stem(0:29, conv3im(41:70), 'r');
64     ax = gca;
65     ax.XAxis.LineWidth = 1.5; %thicken x-axis
66     ax.YAxis.LineWidth = 1.5; %thicken y-axis
67     axh.XAxisLocation = 'origin';
68     axh.YAxisLocation = 'origin';
69     xlabel('n', 'FontSize', 12, 'FontWeight', 'bold');
70     ylabel(['Im(y', num2str(q), '[n]'), 'FontSize', 12, 'FontWeight', 'bold']);
71     grid on
72     axis tight
73     title(['Graph of Im(y', num2str(q), '[n]'), 'FontSize', 14, 'FontWeight', 'bold']);
74     disp(['Re(y', num2str(q), '[n]: ', num2str(conv3r(41:70))]);
75     disp(['Im(y', num2str(q), '[n]: ', num2str(conv3im(41:70))]);
76
77
78     elseif q == 4
79         conv4 = convolution(@h, @s1, 16);
80
81         figure()
82         stem(0:29, conv4(41:70));
83         ax = gca;
84         ax.XAxis.LineWidth = 1.5; %thicken x-axis
85         ax.YAxis.LineWidth = 1.5; %thicken y-axis
86         axh.XAxisLocation = 'origin';
87         axh.YAxisLocation = 'origin';
88         xlabel('n', 'FontSize', 12, 'FontWeight', 'bold');
89         ylabel(['y', num2str(q), '[n]'), 'FontSize', 12, 'FontWeight', 'bold');
90         grid on
91         axis tight
92         title(['Graph of y', num2str(q), '[n]'), 'FontSize', 14, 'FontWeight', 'bold');
93
94         disp(['y', num2str(q), '[n]: ', num2str(conv4(41:70))]);
95
96
97     elseif q == 6
98         conv6r = convolution(@h, @s6r, 16);
99         conv6im = convolution(@h, @s6im, 16);
100        figure()
101        subplot(2,1,1);
102        stem(0:29, conv6r(41:70));
103        ax = gca;
104        ax.XAxis.LineWidth = 1.5; %thicken x-axis
105        ax.YAxis.LineWidth = 1.5; %thicken y-axis
106        axh.XAxisLocation = 'origin';
107        axh.YAxisLocation = 'origin';
108        xlabel('n', 'FontSize', 12, 'FontWeight', 'bold');
109        ylabel(['Re(y', num2str(q), '[n]'), 'FontSize', 12, 'FontWeight', 'bold');
110        grid on
111        axis tight
112        title(['Graph of Re(y', num2str(q), '[n]'), 'FontSize', 14, 'FontWeight', 'bold');
113
114        subplot(2,1,2);
115        stem(0:29, conv6im(41:70), 'r');
116        ax = gca;

```

```

117     ax.XAxis.LineWidth = 1.5; %thicken x-axis
118     ax.YAxis.LineWidth = 1.5; %thicken y-axis
119     axh.XAxisLocation = 'origin';
120     axh.YAxisLocation = 'origin';
121     xlabel('n', 'FontSize', 12, 'FontWeight', 'bold');
122     ylabel(['Im(y', num2str(q), '[n]')], 'FontSize', 12, 'FontWeight', 'bold');
123     grid on
124     axis tight
125     title(['Graph of Im(y', num2str(q), '[n]')], 'FontSize', 14, 'FontWeight', 'bold');
126     disp(['Re(y', num2str(q), '[n]: ', num2str(conv6r(41:70))]);
127     disp(['Im(y', num2str(q), '[n]: ', num2str(conv6im(41:70))]);
128
129
130     else
131         conv5 = convolution(@h, @s5, 16);
132
133         figure()
134         stem(0:29, conv5(41:70));
135         ax = gca;
136         ax.XAxis.LineWidth = 1.5; %thicken x-axis
137         ax.YAxis.LineWidth = 1.5; %thicken y-axis
138         axh.XAxisLocation = 'origin';
139         axh.YAxisLocation = 'origin';
140         xlabel('n', 'FontSize', 12, 'FontWeight', 'bold');
141         ylabel(['y', num2str(q), '[n]'], 'FontSize', 12, 'FontWeight', 'bold');
142         grid on
143         axis tight
144         title(['Graph of y', num2str(q), '[n]'], 'FontSize', 14, 'FontWeight', 'bold');
145
146         disp(['y', num2str(q), '[n]: ', num2str(conv5(41:70))]);
147     end
148 end
149
150
151
152 %convolution func that takes response func, input func and range as inputs
153 function y = convolution(h, x, range)
154     y = zeros(1, 81);
155     for i = -40:40
156         value = zeros(1,33);
157         for j = range * (-1) : range
158             value(j + 17) = x(j)* h(i - j);
159         end
160         y(i + 41) = sum(value);
161     end
162 end
163
164 %functions for the input signals and the response function
165 function x1 = s1(n)
166     if (n >= 0 && n <= 8)
167         x1 = 3;
168     else
169         x1 = 0;
170     end
171 end
172
173 function x2 = s2(n)

```

```
174         if (n >= 0 && n <= 4)
175             x2 = 3;
176         elseif (n >= 5 && n <= 8)
177             x2 = -3;
178         elseif (n >= 9 && n <= 13)
179             x2 = -6;
180         else
181             x2 = 0;
182         end
183     end
184
185
186     %using Euler's formula ( $e^{j(1/3)} = \cos(1/3) + j\sin(1/3)$ )
187     function x3 = s3r(n)
188         if(n >= 2 && n <= 20)
189             x3 = cos(n / 3);
190         else
191             x3 = 0;
192         end
193     end
194     function x3 = s3in(n)
195         if(n >= 2 && n <= 20)
196             x3 = sin(n / 3);
197         else
198             x3 = 0;
199         end
200     end
201
202     function x4 = s4(n)
203         if(n >= 2 && n <= 20)
204             x4 = (-3) * sin(n / 3);
205         else
206             x4 = 0;
207         end
208     end
209
210     function x5 = s5(n)
211         if(n >= 2 && n <= 20)
212             x5 = 2 * cos(n / 3);
213         else
214             x5 = 0;
215         end
216     end
217
218     %using Euler's formula again
219     function x6 = s6r(n)
220         x6 = real(s1(n) + 2i.* s2(n));
221     end
222     function x6 = s6in(n)
223         x6 = imag(s1(n) + 2i.* s2(n));
224     end
225
226     %unit step function
227     function uso = us(n)
228         if(n >= 0)
229             uso = 1;
230         else
231             uso = 0;
232         end
233     end
234
235     %response function
236     function res = h(n)
237         res = us(n-4).*(7/8).^n;
238     end
```

Analytical Part

```

1  %clc
2  clear all
3  close all
4
5  arrayc = 0:100;
6  out1 = zeros(1, length(arrayc));
7  out2 = zeros(1, length(arrayc));
8  out3 = zeros(1, length(arrayc));
9
10 for n = 4:50
11     if n > 12
12         out1(n+1) = 3 * sum((7/8).^(n-8:n));
13     else
14         out1(n+1) = 3 * sum((7/8).^(4:n));
15     end
16 end
17
18 for n = 4:50
19     if n > 8 && n < 13
20         out2(n+1) = 3 * sum((7/8).^(4:n)) - 6 * sum((7/8).^(4:n-5));
21     elseif n <= 8
22         out2(n+1) = 3 * sum((7/8).^(4:n));
23     elseif n >= 13 && n < 18
24         out2(n+1) = 3 * sum((7/8).^(n-8:n)) - 6 * sum((7/8).^(4:n-5));
25     else
26         out2(n+1) = 3 * sum((7/8).^(n-8:n)) - 6 * sum((7/8).^(n-13:n-5));
27     end
28 end
29
30
31 for n = 6:50
32     if n < 25
33         out3(n+1) = ((cos(1/3) + 1i * sin(1/3))^n) * sum(((7/8) * cos(1/3) - 1i * (7/8) * sin(1/3)).^(4:n-2));
34     else
35         out3(n+1) = ((cos(1/3) + 1i * sin(1/3))^n) * sum(((7/8) * cos(1/3) - 1i * (7/8) * sin(1/3)).^(n-20:n-2));
36     end
37 end
38
39 out4 = -3 * imag(out3);
40 out5 = 2 * real(out3);
41 out6 = out1 + 2i * out2;
42 outs = {out1, out2, out3, out4, out5, out6};
43 listt = {'y1[n]', 'y2[n]', 'y3[n]', 'y4[n]', 'y5[n]', 'y6[n]'};
44 |
45 %plotting loop
46 for m = 1:2
47     if m == 1
48         for i = 1:6
49             figure;
50             stem(arrayc, outs{i}, 'r', 'filled', 'LineWidth', 1.5);
51             title(['Graph of', listt{i}], 'FontSize', 14, 'FontWeight', 'bold');
52             xlabel('n', 'FontSize', 12);
53             ylabel(listt{i}, 'FontSize', 12);
54             grid on;
55             xlim([0 50]);
56         end
57     else
58         for i = 1:6
59             figure;
60             stem(arrayc, imag(outs{i}), 'r', 'filled', 'LineWidth', 1.5);
61             title(['Graph of Imaginary Part for ', listt{i}], 'FontSize', 14, 'FontWeight', 'bold');
62             xlabel('n', 'FontSize', 12);
63             ylabel(['Imag(', listt{i}, ')'], 'FontSize', 12);
64             grid on;
65             xlim([0 50]);
66         end
67     end
68 end

```

ANSWER TO QUESTION ABOUT CAUSALITY AND STABILITY

The given system is causal since $h[n] = 0$ for $n < 0$ and there is no input that involves later data. The system is also stable as the following is bounded.

$$\sum_{n=-\infty}^{\infty} |h[n]| = \sum_{n=4}^{\infty} (7/8)^n = \frac{1}{1-7/8} + \sum_{n=0}^3 (7/8)^n < \infty$$

PLOTS (left numeric, right analytic)

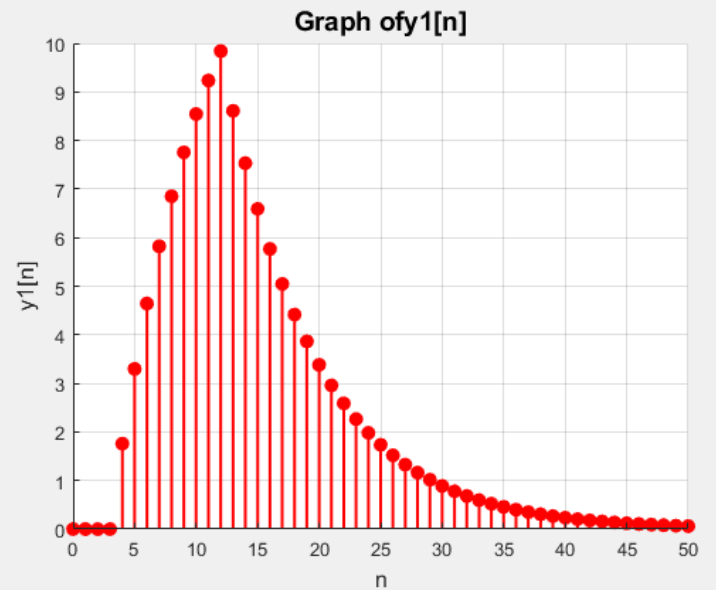
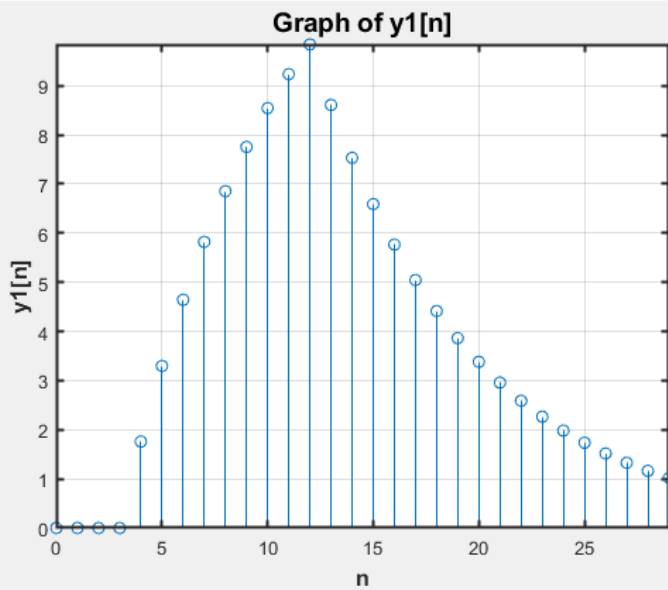


Figure 1: Graphs for $x_1[n]$

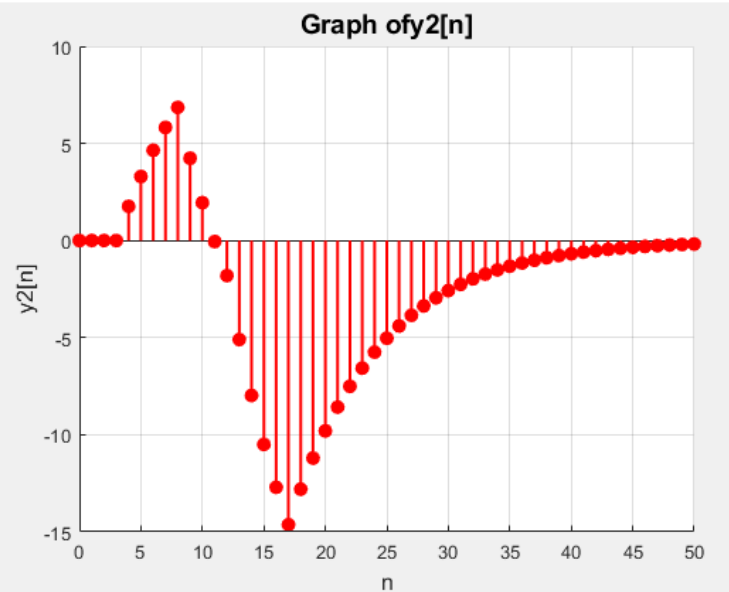
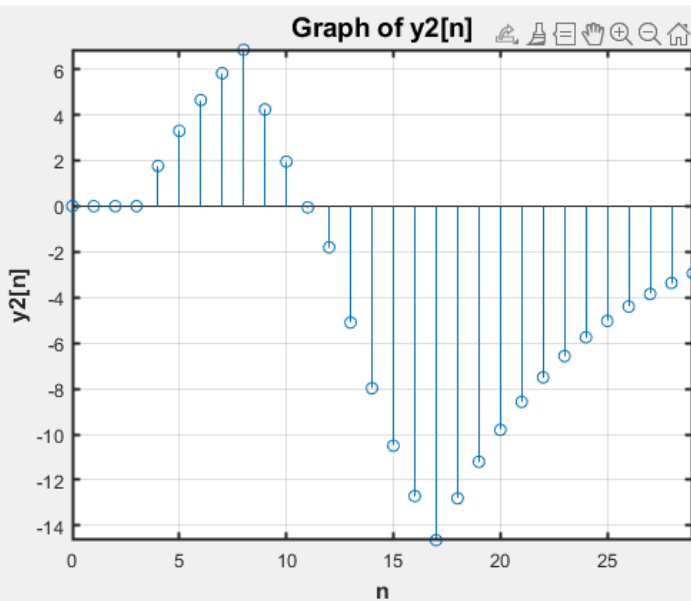


Figure 2: Graphs for $x_2[n]$

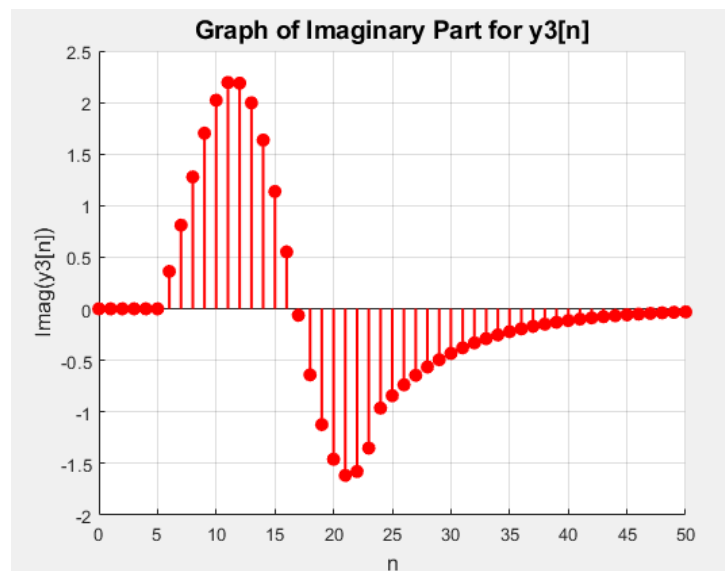
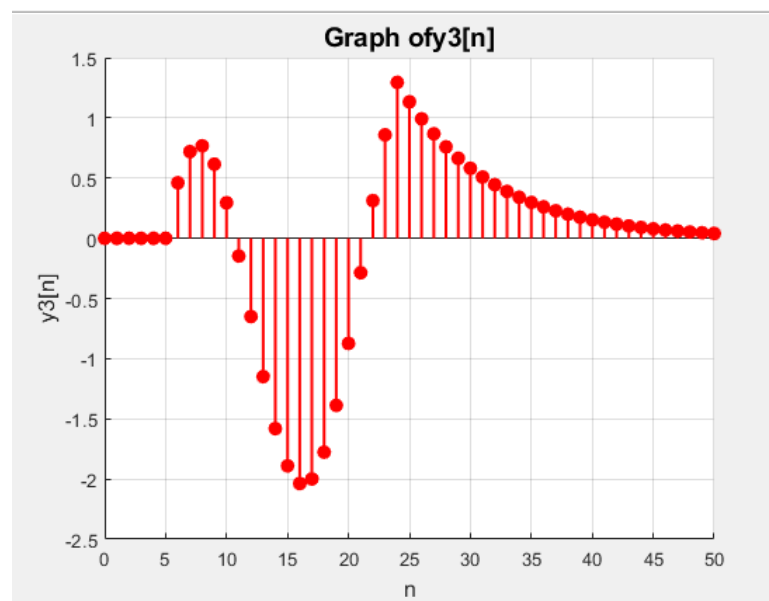
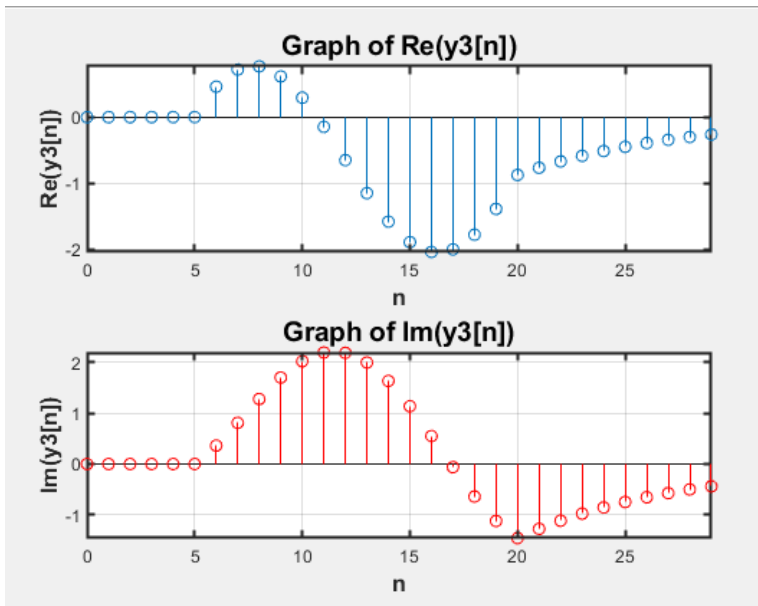


Figure 3: Graphs for $x_3[n]$

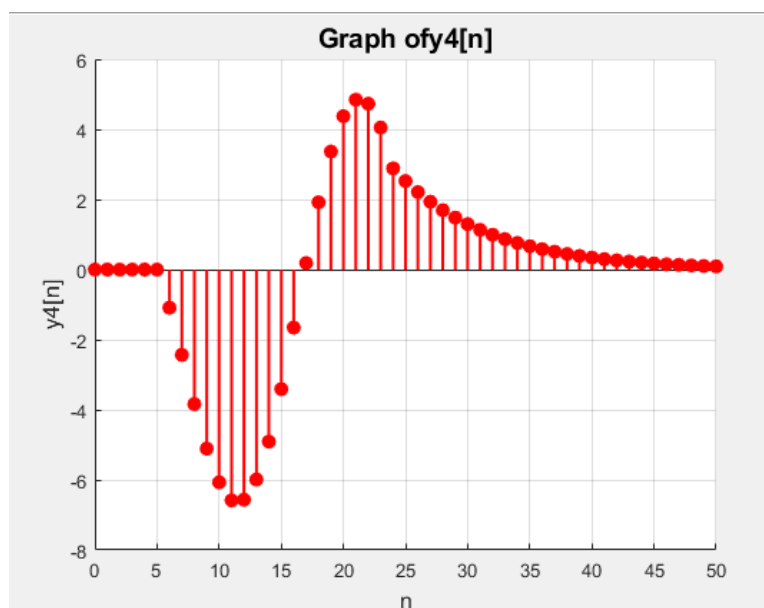
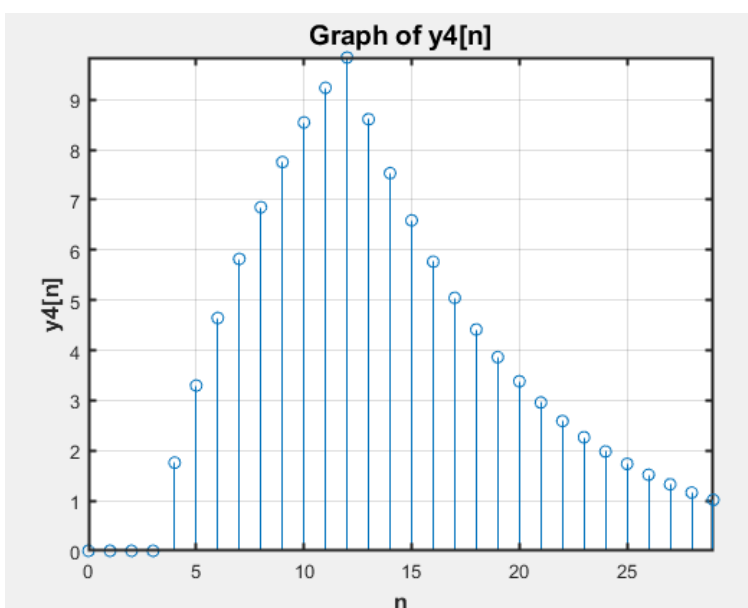


Figure 4: Graphs for $x_4[n]$

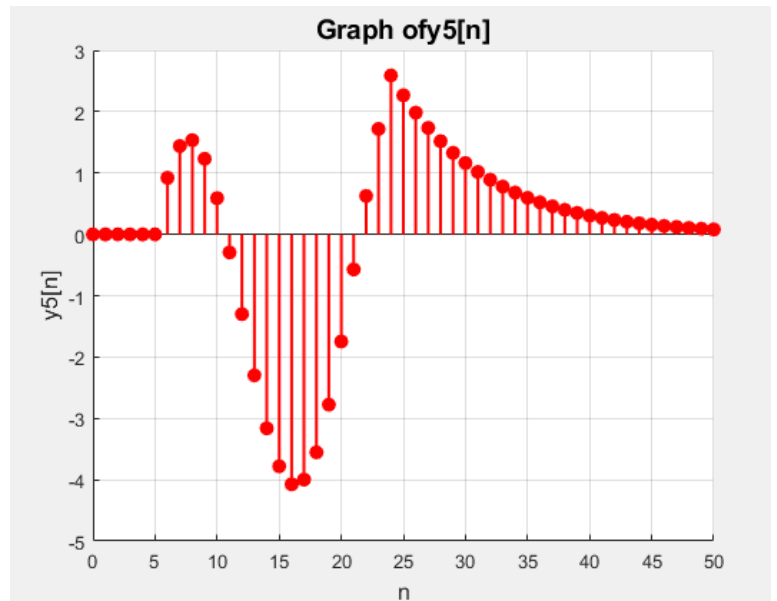
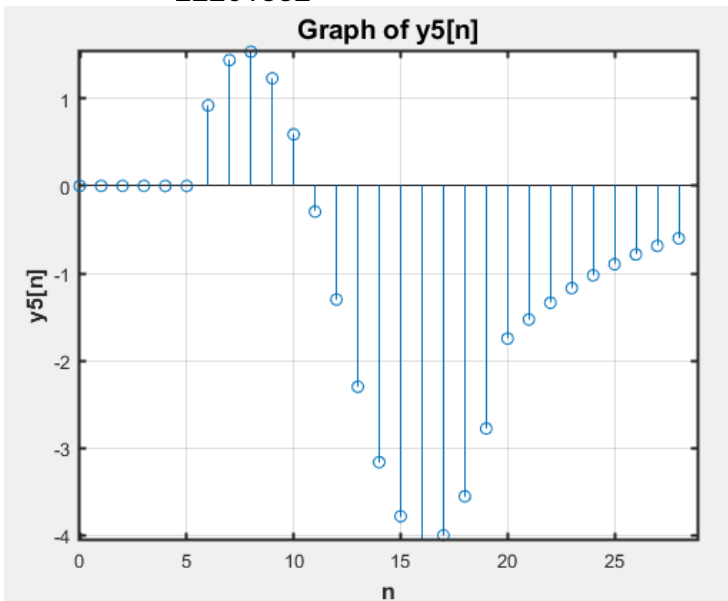


Figure 5: Graphs for $x_5[n]$

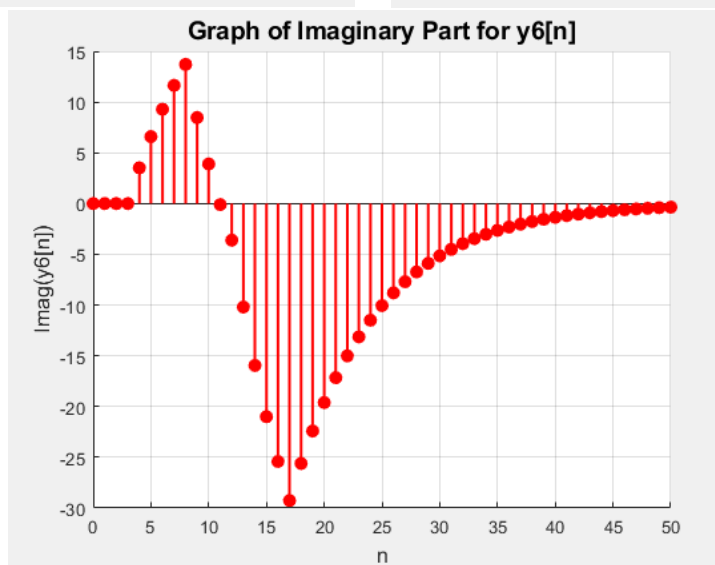
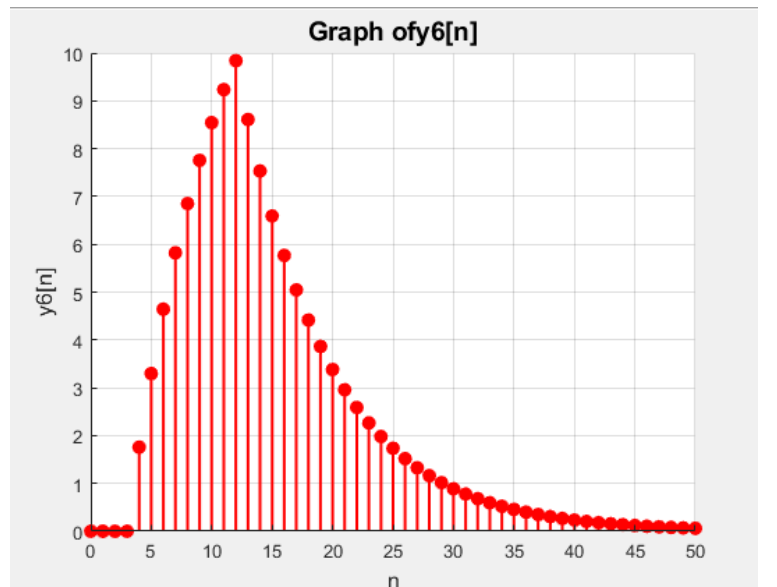
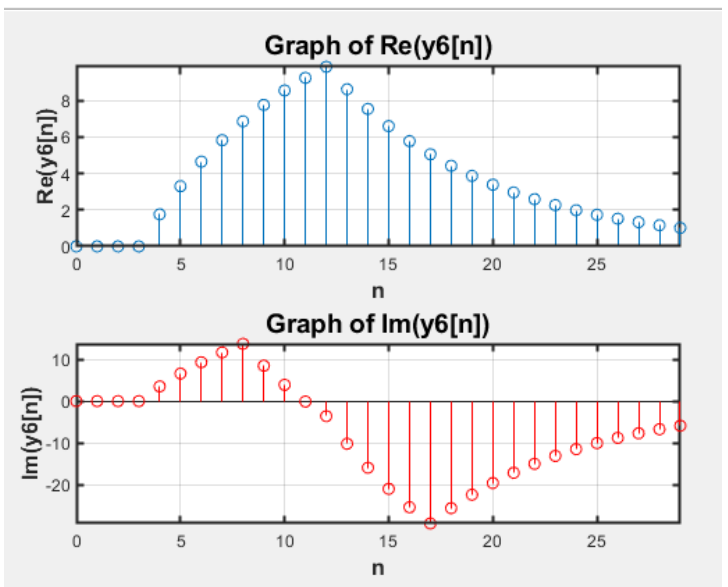


Figure 6: Graphs for $x_6[n]$

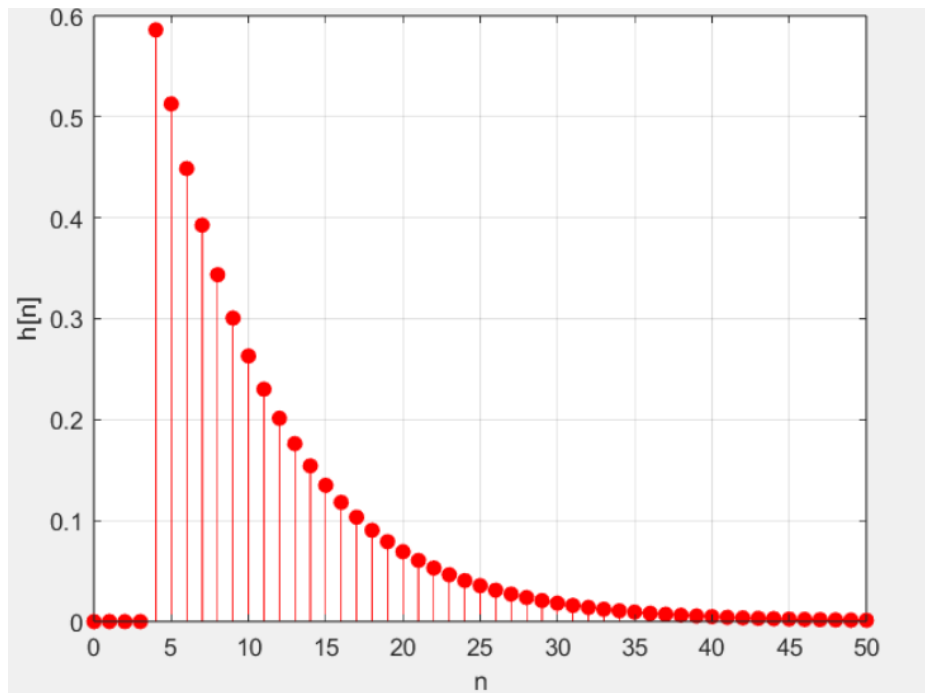


Figure 7: Graph of $h[n]$

The found solutions are equal for all numeric and analytic methods from the two MATLAB files. This lab was useful in terms of observing that properties of LTI systems make solving convolutions simpler, the fact that such systems are linear and time invariant lets us use one system's output to simply reach the convolution of another system, as this was the case for c), d), e) and f) of the manual.

HAND WRITTEN SOLUTIONS

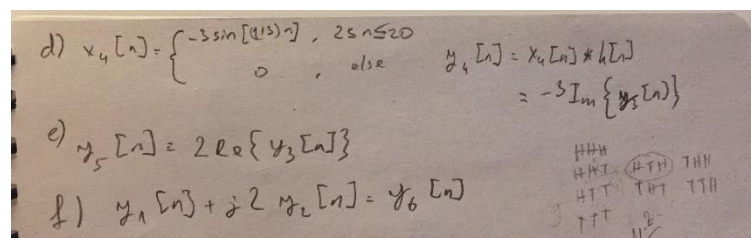
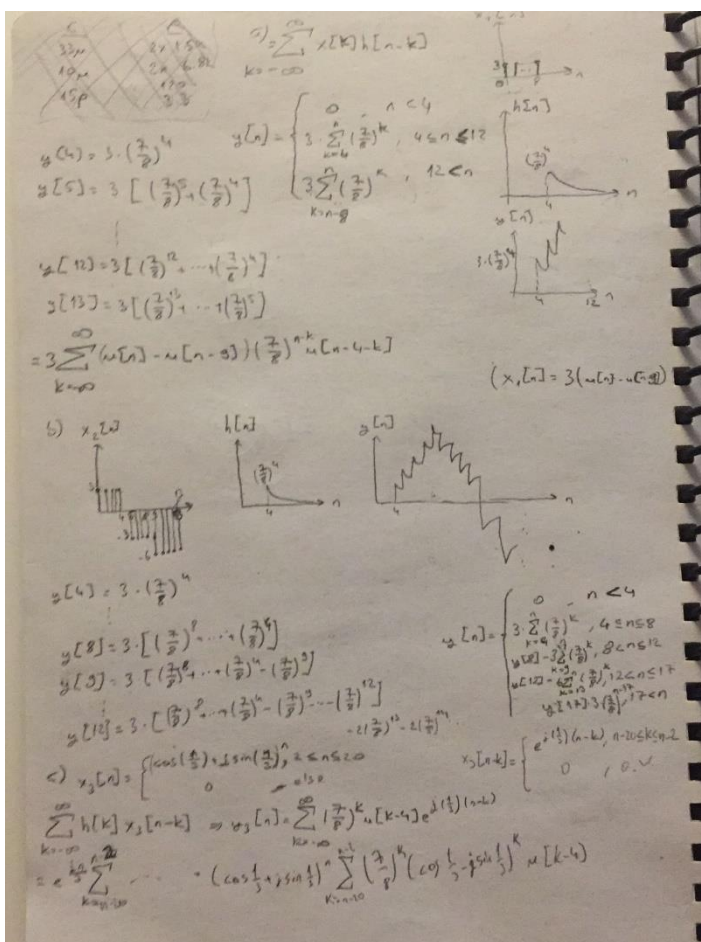


Figure 8: Hand Written Study

Printed Results (same for both)

```
y1[n]: 0      0      0      0  1.7585  3.2973  4.6437  5.8217  6.8526
7.7545  8.5438  9.2343  9.8386  8.6088  7.5327  6.5911  5.7672  5.0463
4.4155  3.8636  3.3806  2.9581  2.5883  2.2648  1.9817  1.734  1.5172
1.3276  1.1616  1.0164
```

```
y2[n]: 0      0      0      0  1.75854  3.29727  4.64366  5.82175  6.85257
4.23746  1.94923 -0.0529697 -1.80489 -5.09637 -7.97641 -10.4965 -12.7015
-14.6309 -12.802 -11.2018 -9.80155 -8.57636 -7.50431 -6.56628 -5.74549
-5.0273 -4.39889 -3.84903 -3.3679 -2.94691
```

```
Re(y3[n]): 0      0      0      0      0      0  0.46067  0.7198  0.76772
0.61564  0.29475 -0.147 -0.64993 -1.149 -1.5808 -1.8904 -2.0373 -
1.9995 -1.7764 -1.3881 -0.87351 -0.76432 -0.66878 -0.58518 -0.51204 -
0.44803 -0.39203 -0.34302 -0.30015 -0.26263
```

```
Im(y3[n]): 0      0      0      0      0      0  0.36248  0.81042  1.2789
1.7025  2.0227  2.1937  2.1875  1.9968  1.6355  1.1372  0.55146 -0.062048
-0.63986 -1.122 -1.4585 -1.2762 -1.1167 -0.97708 -0.85494 -0.74807 -
0.65456 -0.57274 -0.50115 -0.43851
```

```
y4[n]: 0      0      0      0  1.7585  3.2973  4.6437  5.8217  6.8526
7.7545  8.5438  9.2343  9.8386  8.6088  7.5327  6.5911  5.7672  5.0463
4.4155  3.8636  3.3806  2.9581  2.5883  2.2648  1.9817  1.734  1.5172
1.3276  1.1616  1.0164
```

```
y5[n]: 0      0      0      0      0      0  0.92135  1.4396  1.5354  1.2313
0.5895 -0.29401 -1.2999 -2.298 -3.1616 -3.7809 -4.0746 -3.9991 -3.5528
-2.7761 -1.747 -1.5286 -1.3376 -1.1704 -1.0241 -0.89606 -0.78405 -
0.68605 -0.60029 -0.52526
```

```
Re(y6[n]): 0      0      0      0  1.7585  3.2973  4.6437  5.8217  6.8526
7.7545  8.5438  9.2343  9.8386  8.6088  7.5327  6.5911  5.7672  5.0463
4.4155  3.8636  3.3806  2.9581  2.5883  2.2648  1.9817  1.734  1.5172
1.3276  1.1616  1.0164
```

```
Im(y6[n]): 0      0      0      0  3.51709  6.59454  9.28732  11.6435
13.7051  8.47491  3.89846 -0.105939 -3.60979 -10.1927 -15.9528 -20.9929
-25.403 -29.2618 -25.6041 -22.4036 -19.6031 -17.1527 -15.0086 -13.1326
-11.491 -10.0546 -8.79778 -7.69806 -6.7358 -5.89383
```

>>