Assignment 2

Course: DSAA, Monsoon 2017 @IIITS

Name: Dash Subhadeep

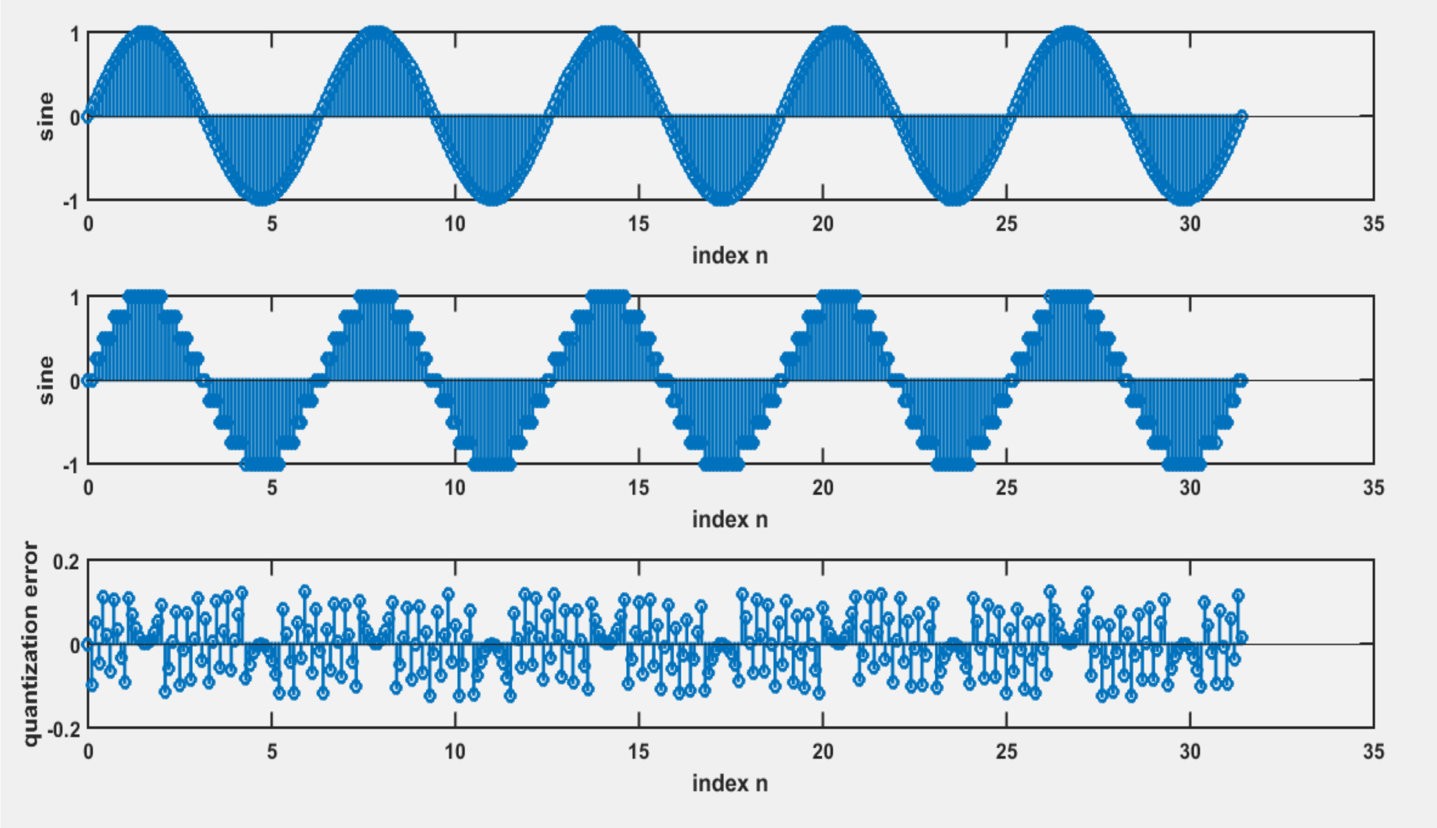
Roll no. 201601021

I. QUANTIZATION OF A DISCRETE SIGNAL

Matlab Code:

```
clc
clear all
close all
%%
N = 5;
Fs = 5;
T = 1 / Fs;
n = 0: 0.1: N * 2 * pi;
x = sin(n);
figure;
subplot(3, 1, 1);
stem(n, x, 'LineWidth', 2);
set(gca, 'Box', 'on', ...,
      'FontSize', 12, ...,
      'FontWeight', 'bold', ...,
      'LineWidth', 1.5, ...,
      'Color', [0.95 0.95 0.95], ...,
      'XGrid', 'off', ...,
      'YGrid', 'off');
xlabel('index n');
ylabel('sine');
subplot(3, 1, 2);
stem(n, 0.25 * round(x / 0.25), 'LineWidth', 2);
set(gca, 'Box', 'on', ...,
      'FontSize', 12, ...,
      'FontWeight', 'bold', ...,
      'LineWidth', 1.5, ...,
      'Color', [0.95 0.95 0.95], ...,
      'XGrid', 'off', ...,
      'YGrid', 'off');
xlabel('index n');
ylabel('sine');
subplot(3, 1, 3);
stem(n, x - 0.25 * round(x / 0.25), 'LineWidth', 2);
set(gca, 'Box', 'on', ...,
      'FontSize', 12, ...,
      'FontWeight', 'bold', ...,
      'LineWidth', 1.5, ...,
      'Color', [0.95 0.95 0.95], ...,
      'XGrid', 'off', ...,
      'YGrid', 'off');
xlabel('index n');
ylabel('quantization error');
```

Results:



Discussion:

1. **The first graph is the original discrete sine signal $x[n]$ which is quantised using the formula $x_q[n] = \Delta\left[\frac{x[n]}{\Delta}\right]$, where Δ is the interval used for quantisation.**
2. **We observe that the obtained quantised signal is quite like the original signal barring a few changes.**
3. **We have taken $\Delta = 0.25$, so we observe that the quantization error ranges between $\frac{\Delta}{2}$ and $-\frac{\Delta}{2}$ i.e. 0.125 and -0.125.**
4. **Therefore, we have managed to attain the quantisation error by subtracting the original signal from the quantised signal.**

II. SIGNAL TRANSFORMATION

a) Unit Step Signal

Matlab Code for function of Unit Step Signal:

```
function myUnitSignal(p, q)
t = -10: 0.001: 10;
x = zeros(size(t));
for i = 1: length(t);
m = p * t(i) + q;
if m > 0 x(i) = 1;
end;
end;
figure;
plot(t, x);
xlabel('time');
```

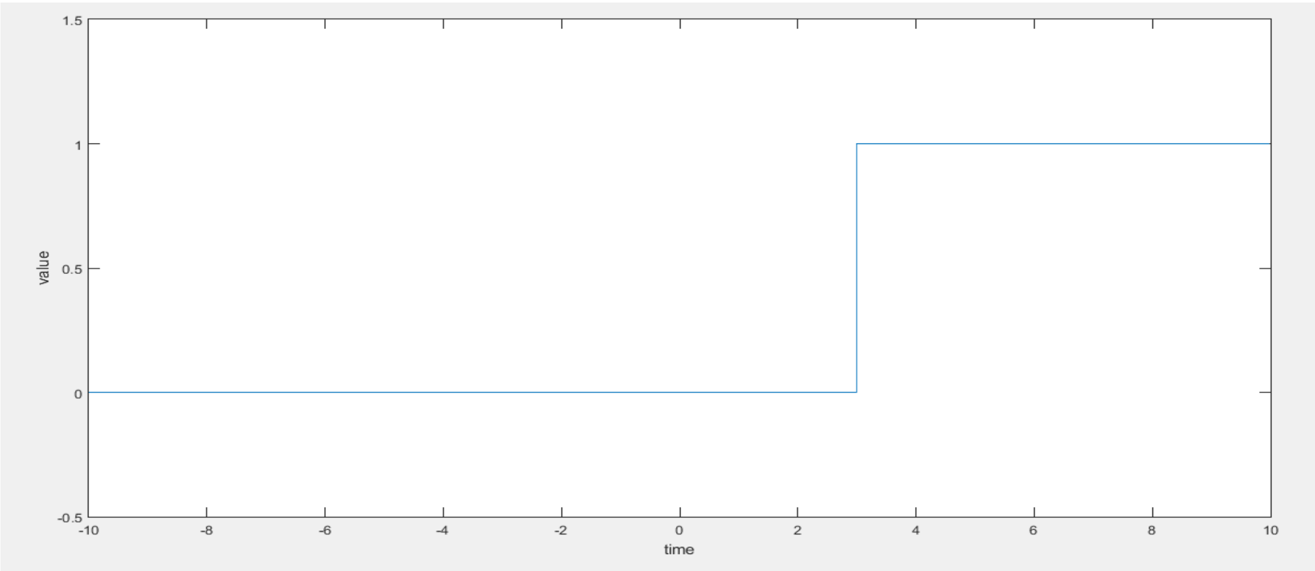
```
ylabel('value');
ylim([-0.5, 1.5]);
```

Matlab Code for the transformations of Unit Step Signal as required in this question making use of the previous function:

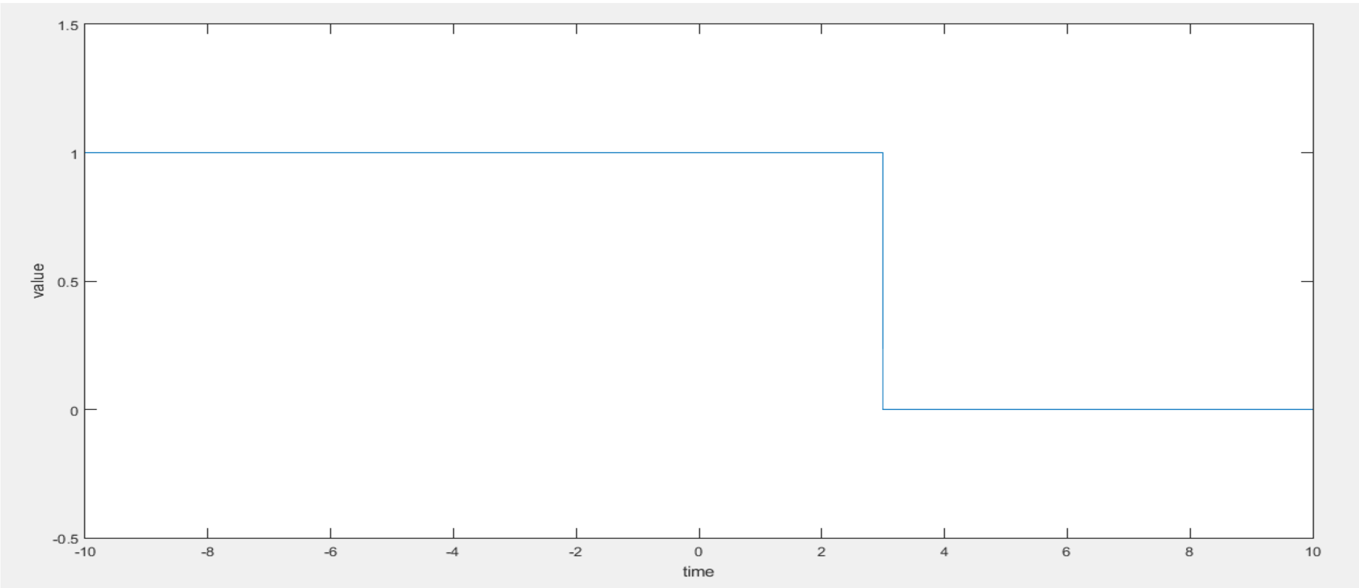
```
clc;
clear all;
close all;
%%
myUnitSignal(1, -3);
myUnitSignal(-1, 3);
myUnitSignal(1, 4);
```

Results:

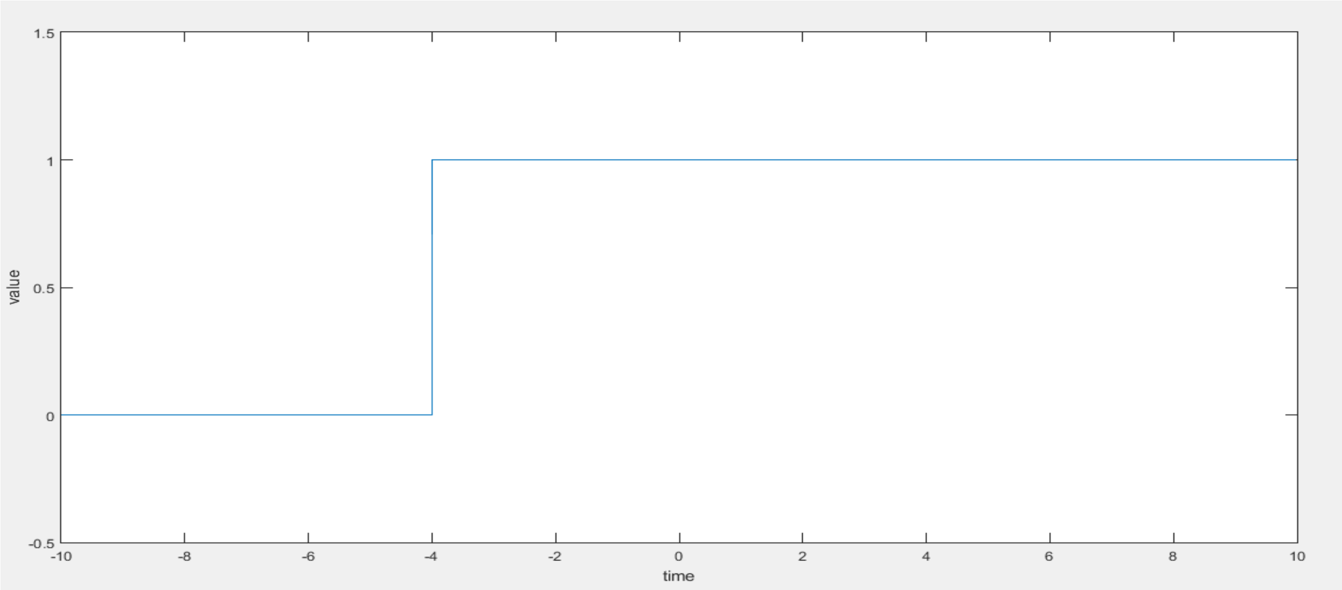
i) First Transformation (u(t – 3)):



ii) Second Transformation (u(3 – t)):



iii) Third transformation (u(t + 4)):



Discussion:

1. **In the first transformation, we delay the unit signal by 3 steps to obtain u(t – 3).**
2. **In the second transformation, we advance the unit signal by 3 steps and reverse the newly obtained signal to obtain u(3 - t).**
3. **In the third transformation, we advance the unit signal by 4 steps to obtain the desired signal.**

b) Ramp Signal:

Matlab code for function of Ramp Signal:

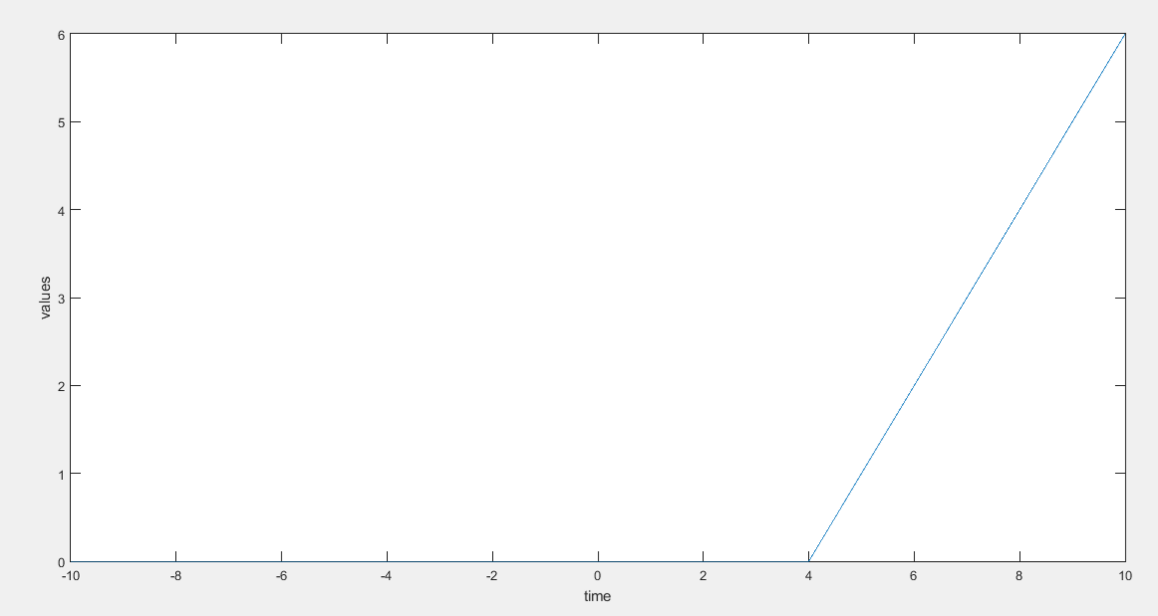
```
function myRampSignal(p, q)
t = -10: 0.001: 10;
x = zeros(size(t));
for i = 1: length(t) m = p * t(i) + q;
if m > 0 x(i) = m;
end;
end;
figure;
plot(t, x);
xlabel('time');
ylabel('values');
```

Matlab Code for the transformations of Unit Step Signal as required in this question making use of the previous function:

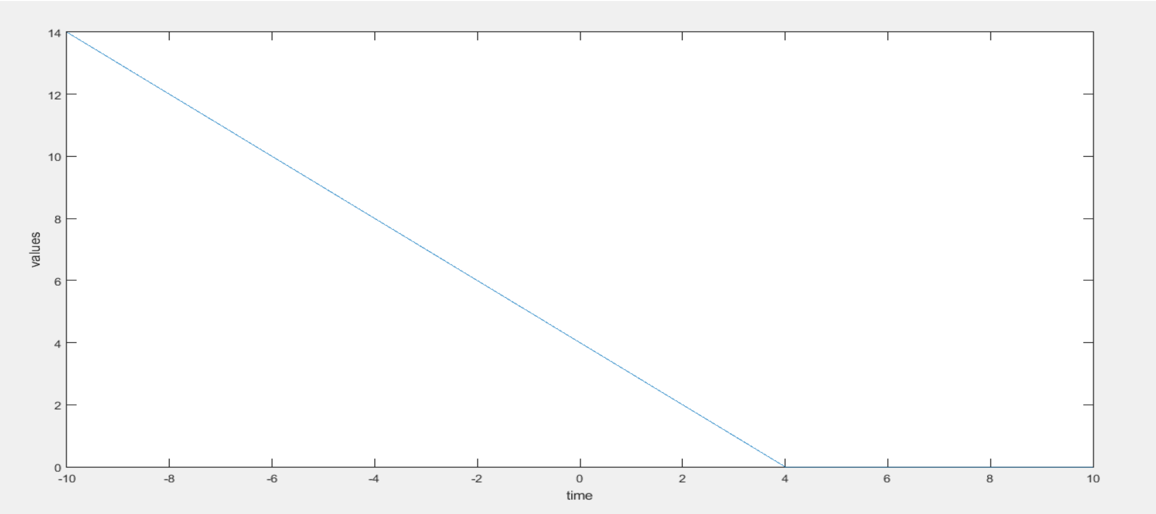
```
clc;
clear all;
close all;
%%
myRampSignal(1, -4);
myRampSignal(-1, 4);
myRampSignal(-2, 1);
```

Results:

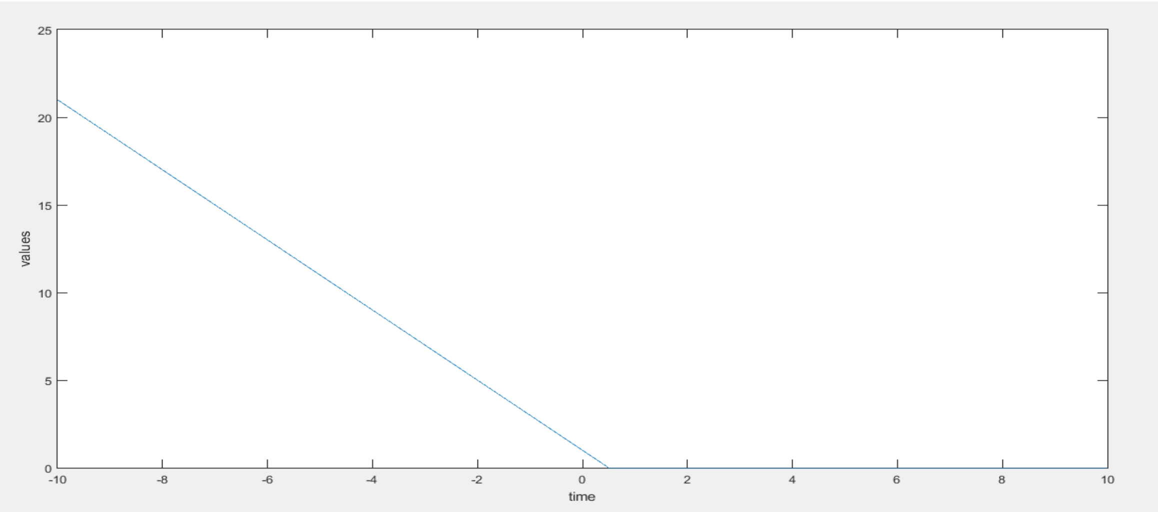
i) First transformation (r(t – 4)):



ii) Second transformation (r(4 – t)):



iii) Third transformation (r(1 – 2t)):



Discussion:

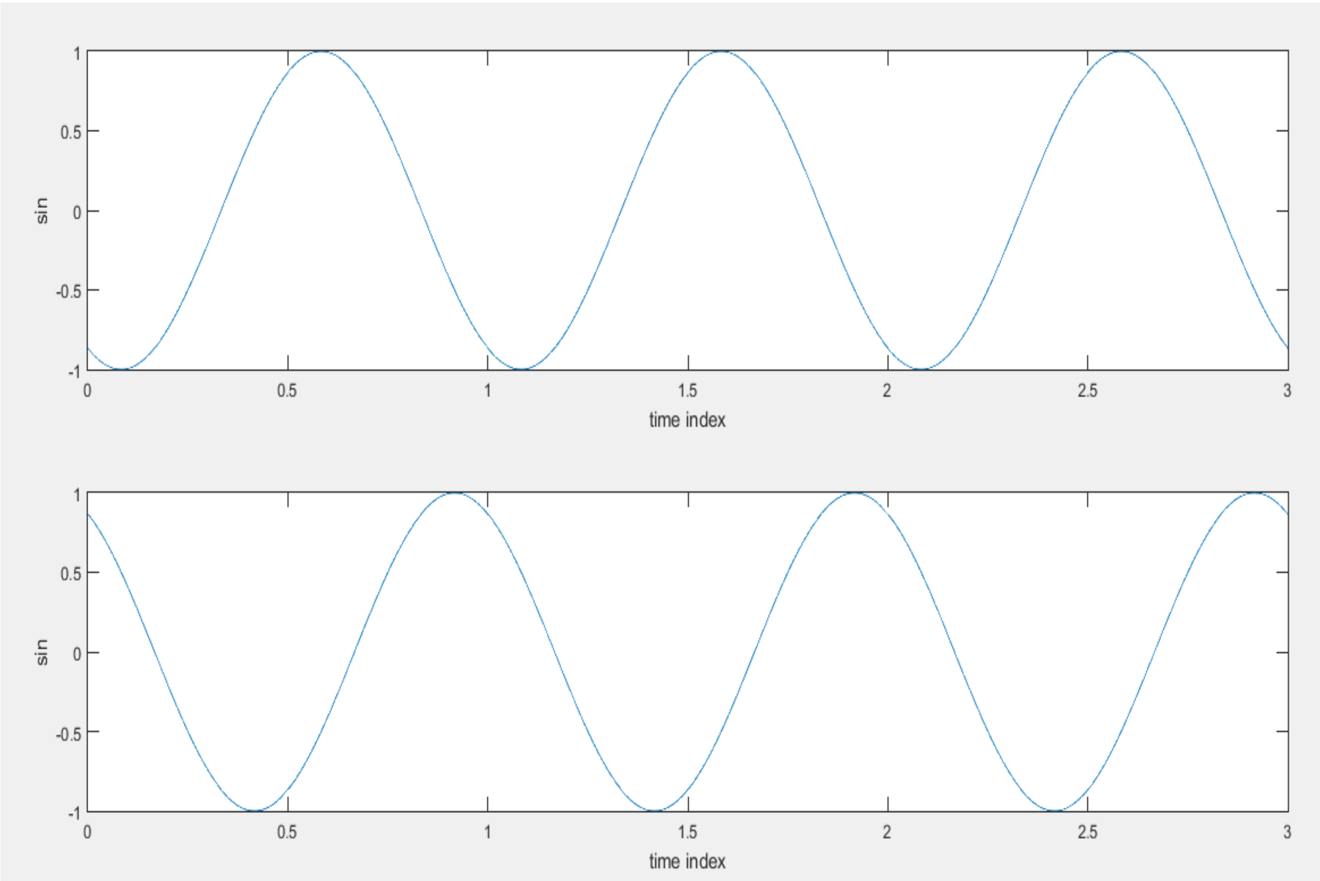
1. In the first case, we are delaying the ramp signal by 4 steps, so that it starts from 4 instead of 0 to obtain r(t – 4).
2. In the next case, the first signal is reversed to obtain r(4 – t).
3. In the last transformation, we are delaying the signal by 1, reversing the signal and scaling it by ½ to obtain r(1 – 2t).

c) **Sine wave signal shifting:**

Matlab code for both delay and advance of sine waves by t0:

```
clc;
close all;
clear all;
%%
t = 0: 0.001: 3;
t0 = 1 / 3;
w0 = 2 * pi;
x = sin(w0 * (t - t0));
subplot(2, 1, 1);
plot(t, x);
xlabel('time index');
ylabel('sin');
x = sin(w0 * (t + t0));
subplot(2, 1, 2);
plot(t, x);
xlabel('time index');
ylabel('sin');
```

Results:



Discussion:

1. As per the given question, $sin(\Omega_0 t)$ is the given signal.
2. When we plot the graphs corresponding to $sin(\Omega_0(t - t_0))$ and $sin(\Omega_0(t + t_0))$, we observe a phase difference of $-\Omega_0 t_0$ and $\Omega_0 t_0$ in the sine graphs obtained.
3. Here the assumed values are $\Omega_0 = 2\pi$ and $t_0 = \frac{1}{3}$.

Given analog signal plot and transformation

Matlab Code for the given analog signal:

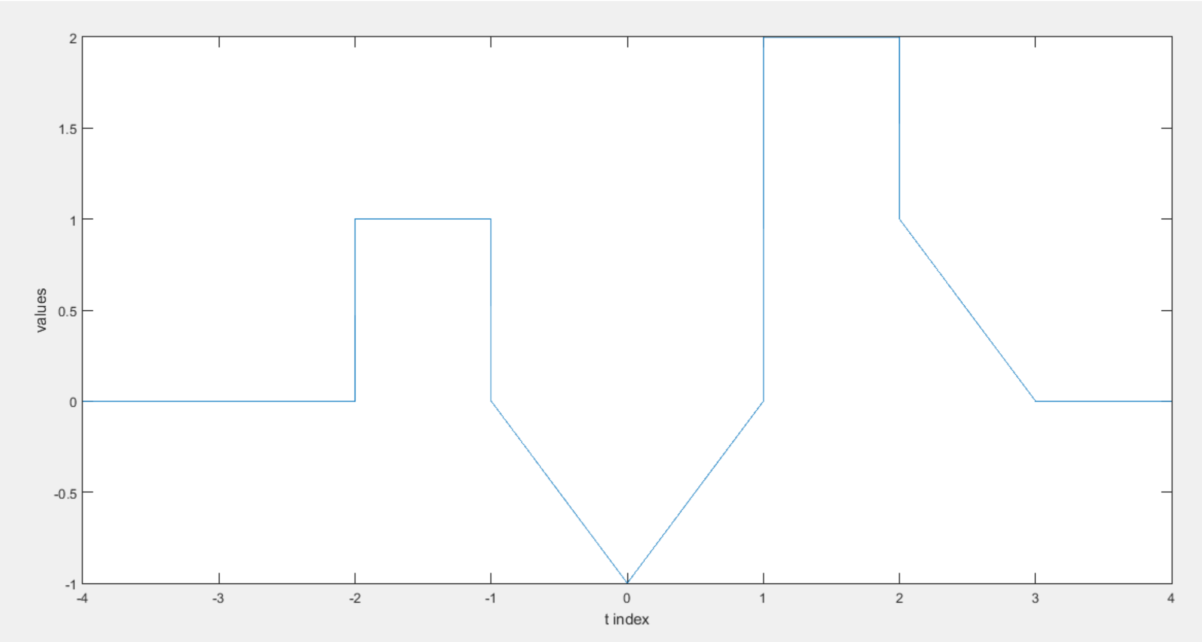
```
function mySignal(q, p) a = (-4 - p) / q;
b = (4 - p) / q;
if (a > b) temp = a;
a = b;
b = temp;
end;
x = a: 0.001: b;
for i = 1: length(x)
m = p + q * x(i);
if (m <= -4) || (m >= (3))
y(i) = 0;
elseif m >= (-4) && m <= (-2)
y(i) = 0;
elseif m >= (-2) && m <= (-1)
y(i) = 1;
elseif m >= (-1) && m <= (0)
y(i) = (-m - 1);
elseif m >= (0) && m <= (1)
y(i) = (m - 1);
elseif m >= (1) && m <= (2)
y(i) = 2;
elseif m >= (2) && m <= (3)
y(i) = -m + 3;
end;
end;
figure;
plot(x, y);
xlabel('t index');
ylabel('values');
```

Matlab Codes for the transformed signals:

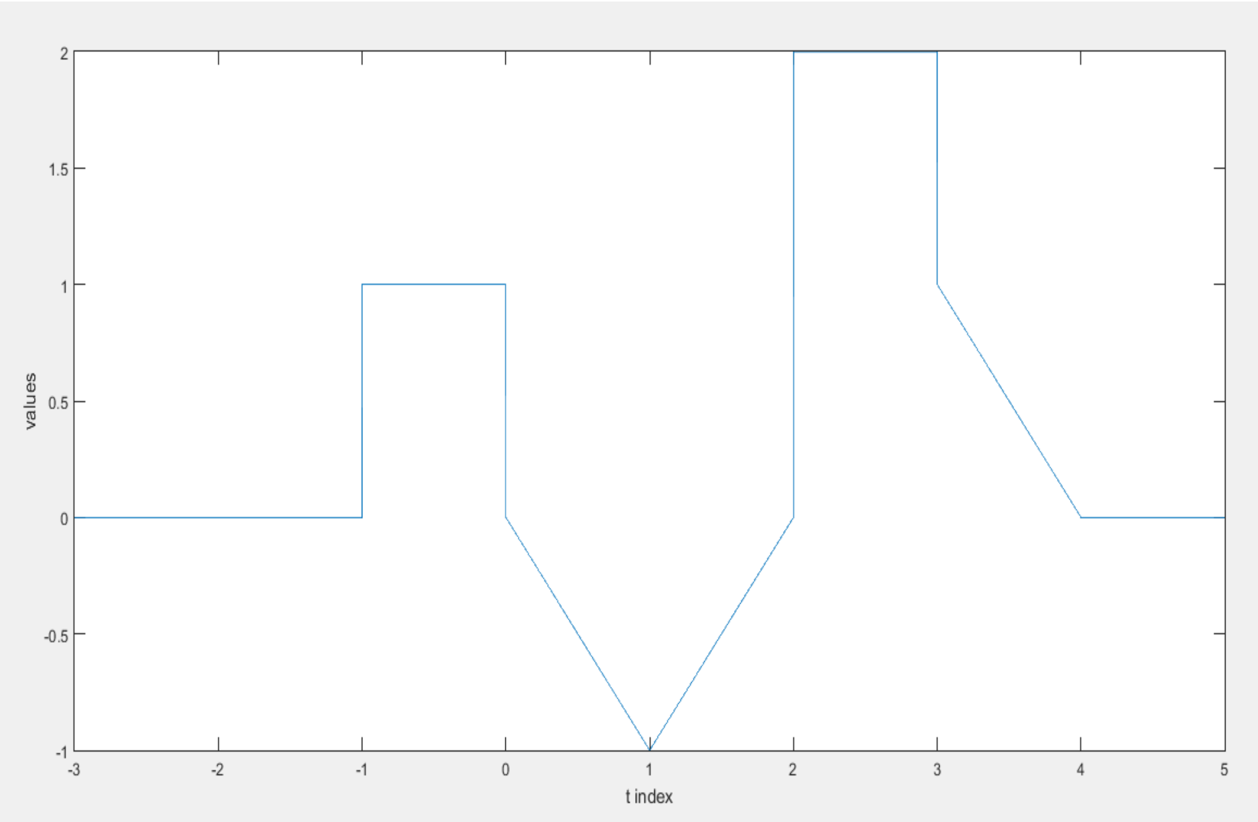
```
clc;
clear all;
close all;
%%
mySignal(1, 0);
mySignal(1, -1);
mySignal(1, 1);
mySignal(2, -3);
mySignal(-2, 1);
```

Results:

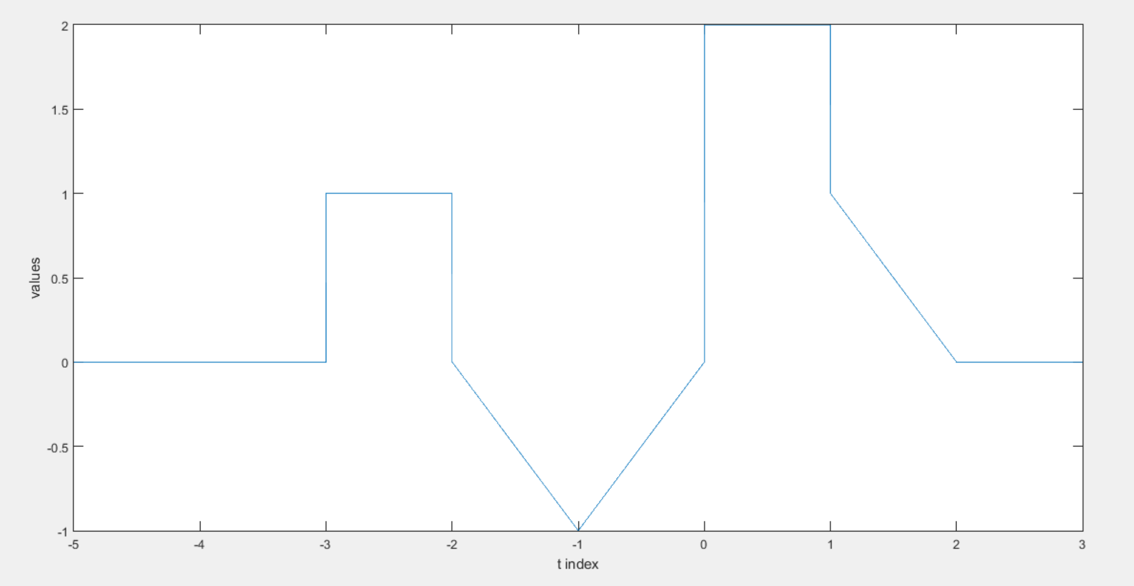
Original Plot:



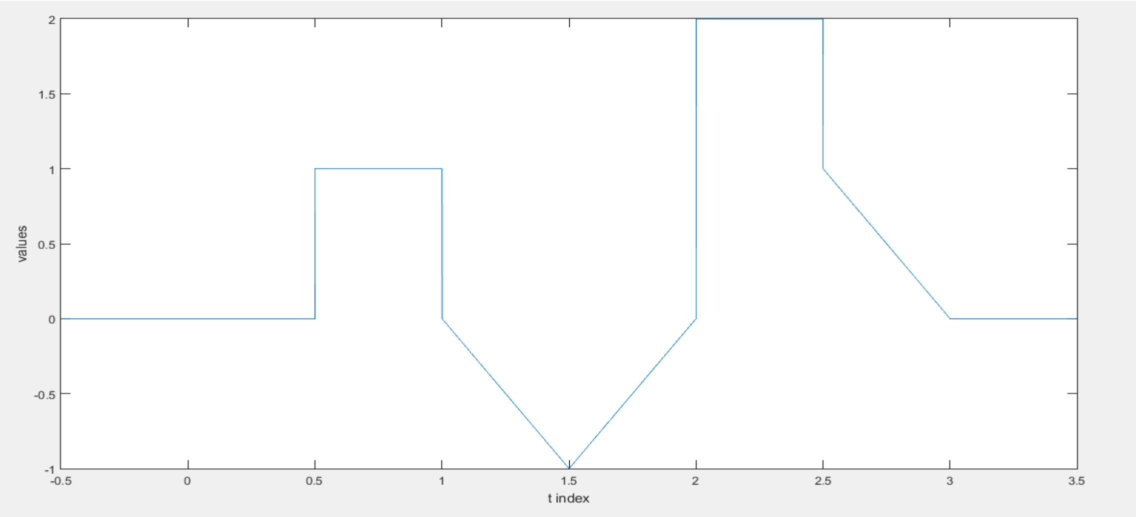
i) First Transformation (x(t – 1)):



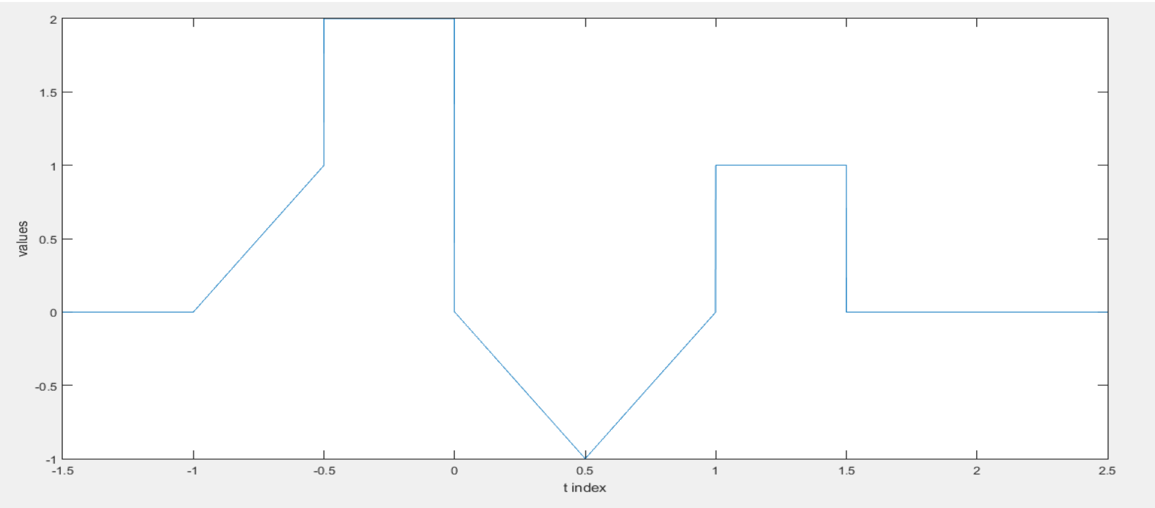
ii) Second Transformation (x(t + 1)):



iii) Third Transformation(x(2t – 3)):



iv) Fourth Transformation(x(1 – 2t)):



Discussion:

1. **Even in this case, we have created a function for the graph given.**
2. **So, when we give the parameters of the coefficient of t and the constant as arguments to the function, it plots the graph automatically, without much human effort.**
3. **Only the graphs with negative coefficients of t are reversed which can be seen in the fourth transformation.**
4. **In first two cases, it's just shifting and in the next two cases both shifting and scaling occurs.**

d) **Given discrete signal plot and transformation**

Matlab Code for the given discrete signals:

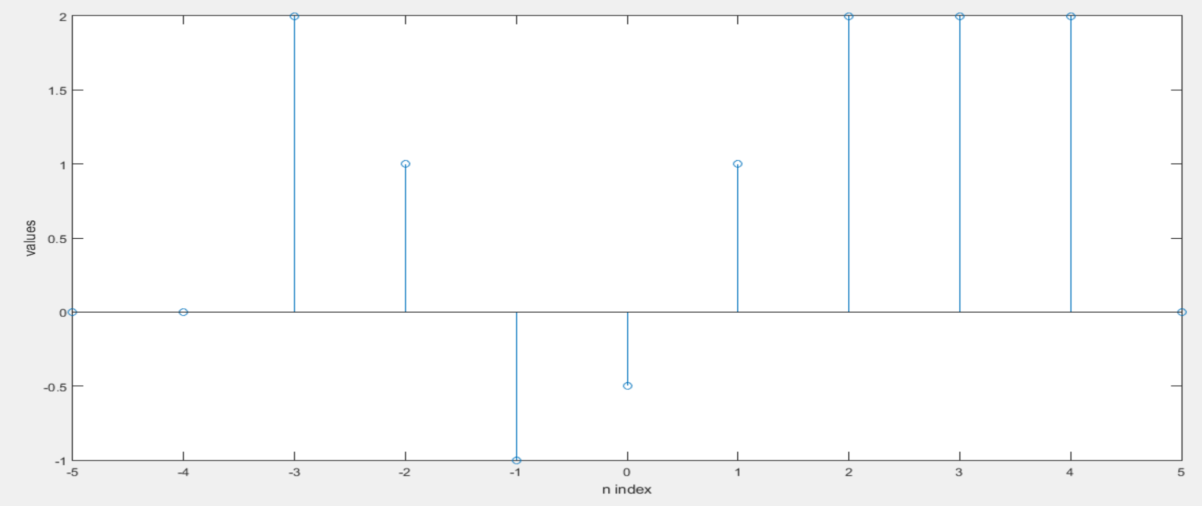
```
function myDiscreteSignal(q, p)
a = (-4 - p) / q;
b = (4 - p) / q;
if (a > b)
temp = a;
a = b;
b = temp;
end;
t = a - 1: 1 / abs(q): b + 1;
x = zeros(size(t));
for i = 1: length(t);
m = q * t(i) + p;
if m == -3
x(i) = 2;
elseif m == -2
x(i) = 1;
elseif m == -1
x(i) = -1;
elseif m == 0
x(i) = -0.5;
elseif m == 1
x(i) = 1;
elseif m >= 2 && m <= 4
x(i) = 2;
end;
end;
figure;
stem(t, x);
xlabel('n index');
ylabel('values');
```

Matlab Codes for the given transformed signals:

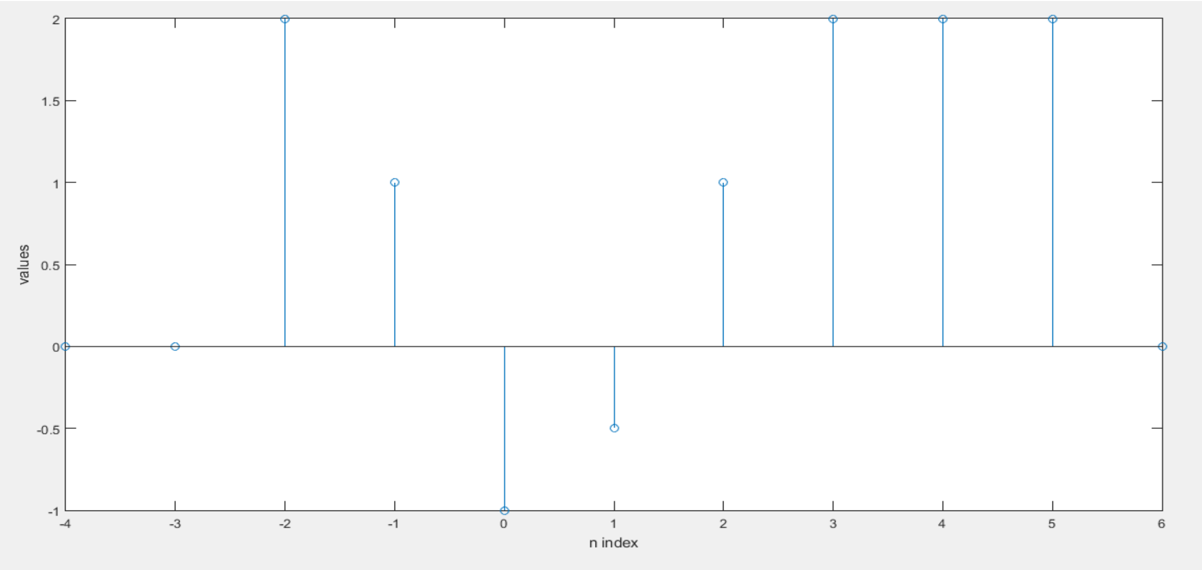
```
clc;
clear all;
close all;
%%
myDiscreteSignal(1, 0);
myDiscreteSignal(1, -1);
myDiscreteSignal(1, 2);
myDiscreteSignal(-1, 2);
myDiscreteSignal(-2, 1);
myDiscreteSignal(2, 3);
```

Results:

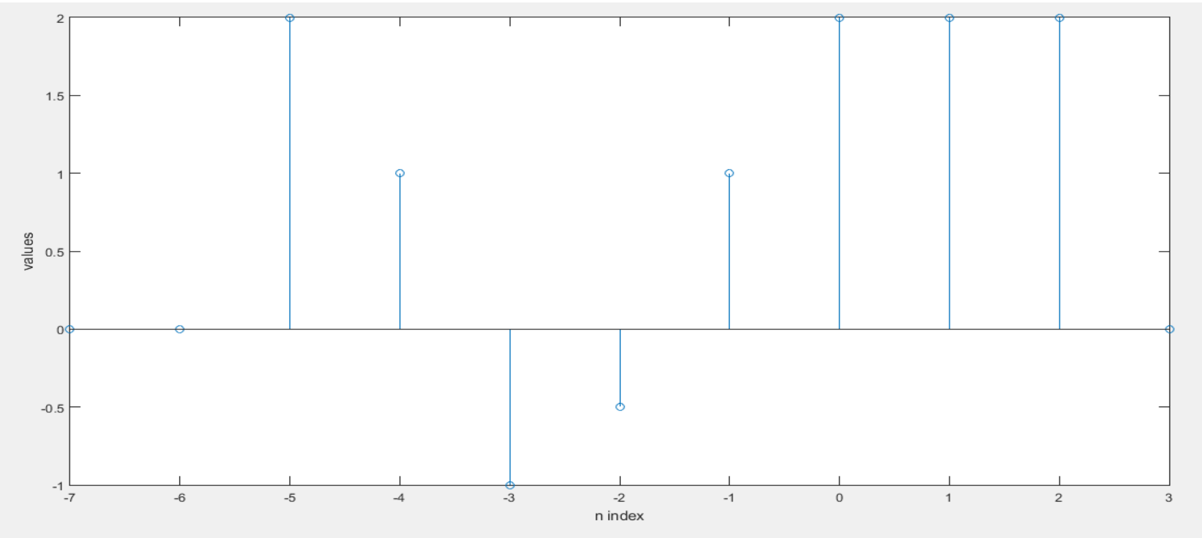
Original Plot:



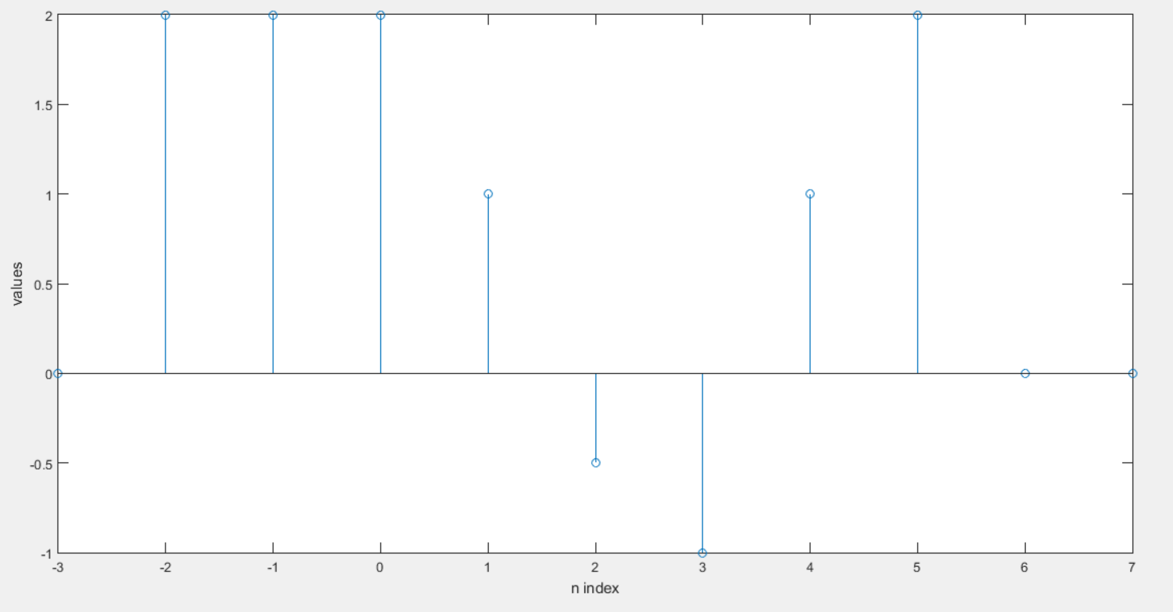
i) First Transformation (x[n – 1]):



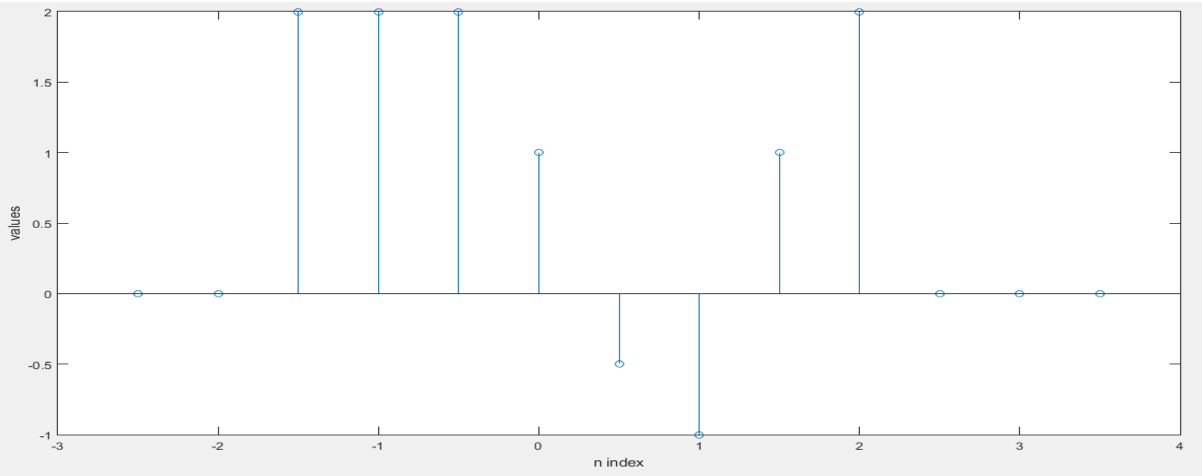
ii) Second Transformation(x[n + 2]):



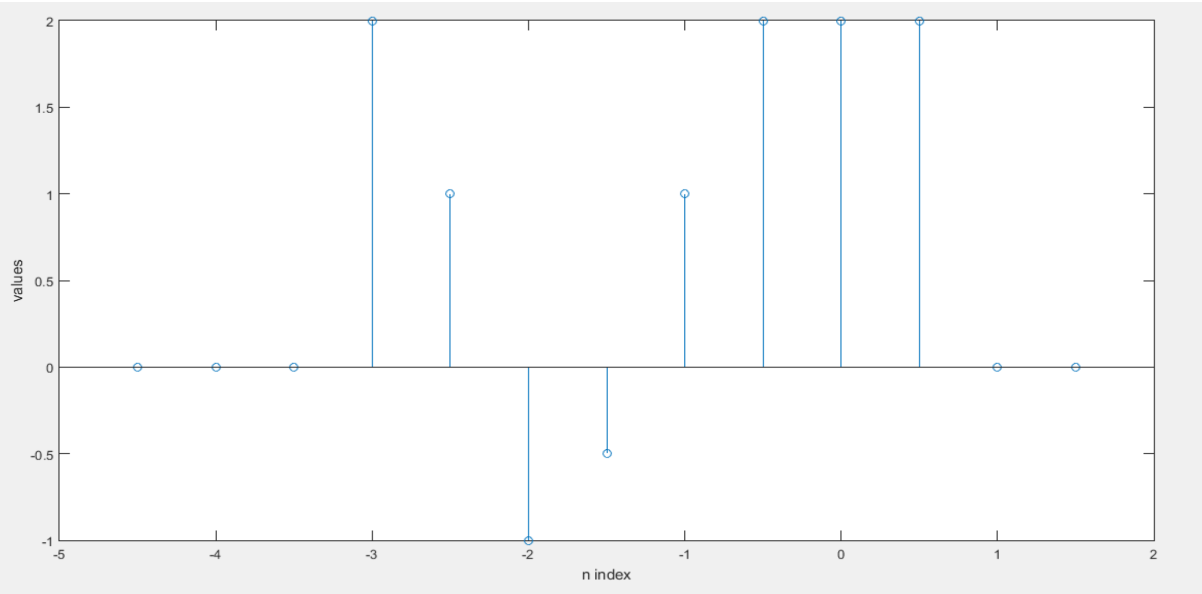
iii) Third Transformation (x[2 - n]):



iv) Fourth Transformation (x[1 – 2n]):



v) Fifth Transformation (x[2n + 3]):



Discussion:

1. **The given irregular discrete signal is initially plotted by creating a function for it.**
2. **The same procedure is followed for reversing, shifting and scaling in each case as done earlier.**
3. **We send the parameters of coefficient of the time in the function and the constant as arguments into the function and hence the transformed graph plot is done.**

Final Discussion:

1. **From this assignment, we have learnt transforming signals by shifting, scaling and reversing.**
2. **We have learnt quantising a discrete sine signal and hence have plotted the quantisation error.**

******Thanks for Reading******