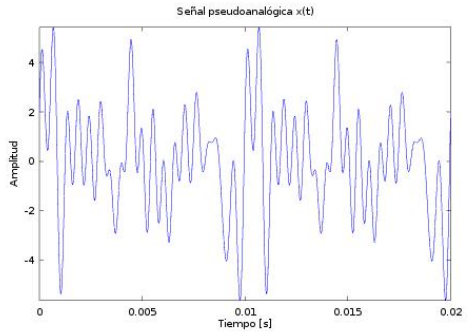
```
clear all;
f0 = 100; % frecuencia fundamental de la señal analogica
T0 = 1 / f0; % periodo de la señal
res = 32; % resolucion de la señal pseudo-analógica
N = 20; % numero de armonicos de la señal
fs_mat = res * N * f0;
Ts_mat = 1 / fs_mat; % incremento del vector de tiempo
t = 0 : Ts_mat : 2 * T0; % vector de tiempo
xt = pseudoanalog(f0, N, t);

samples = 4;
fs = samples * N * f0;
Ts = 1 / fs;
xn = muestrear(xt, res, samples);

n = 0 : Ts : 2 * T0;
N = 16; % niveles para cuantizar
m = log2(N); %bits a codificar
xd = cuantizar(xn, N);

Tb = Ts / m;
b = 0 : Tb : 2 * T0 + (m - 1) * Tb;
xc = codificar(xd, m);

figure; hold on;
subplot(2, 2, 1); plot(t, xt);
title('Señal pseudoanalógica x(t)');
xlabel('Tiempo [s]'); ylabel('Amplitud'); axis('tight');

subplot(2, 2, 2); stem(n, xn);
title('Señal muestreada x[n]');
xlabel('Tiempo [s]'); ylabel('Amplitud'); axis('tight');

subplot(2, 2, 3); stairs(n, xd);
title('Señal cuantizada x[n]');
xlabel('Tiempo [s]'); ylabel('Nivel de cuantización'); axis('tight');

subplot(2, 2, 4); stairs(b, xc);
title('Señal codificada x(t)');
xlabel('Tiempo [s]'); ylabel('Nivel lógico'); axis([0 (Ts * 3) 0 1.2]);
xsize = 0:Tb:Ts*4; % ciclos del la señal codificada a mostrar
set(gca, 'xtick', xsize); set(gca, 'ytick', 0:1); grid on;
hold off;
```

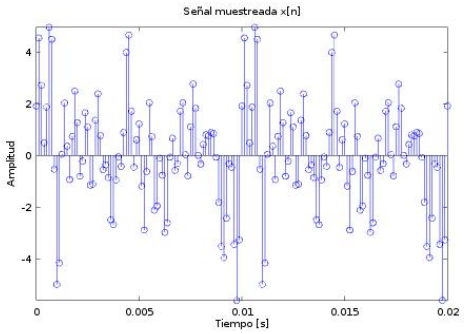
```
function x = pseudoanalog(f0, N, t)
    x = 0 * t;
    for i = 1:N
        A = rand(1);
        O = rand(1) * pi;
        x = x + A * cos(2 * pi * i * f0 * t - O);
    end
```

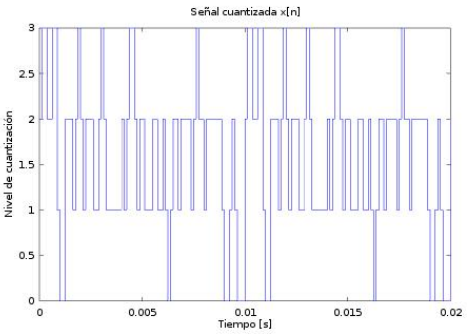
```
function x = muestrear(signal, res, samples)
  n = 1;
  t = 1 + (n - 1) * res / samples;
  while t <= length(signal)
    x(n) = signal(t);
    n++;
    t = 1 + (n - 1) * res / samples;
  end
```

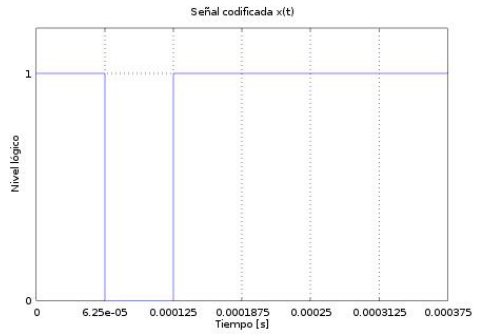
```
function x = cuantizar(signal, levels)
  mx = max(signal);
  mn = min(signal);
  q = (abs(mx) + abs(mn)) / levels;
  signal = signal + abs(mn) - (q / 2); % volvemos completamente positiva la señal
  % y nos aseguramos que los valores max y min salgan del la escala por q/2
  x = round(signal / q);
  x(x > (levels - 1)) = levels - 1; % si algun valor es mayor al numero de niveles
  x(x < 0) = 0; % si algun valor es menor a 0
```

```
function x = codificar(signal, m)
  x = [];
  for i = signal
    x = [x (dec2bin(i, m) - '0')]; % truco raro de matlab para obtener un vector de digitos
  end
```

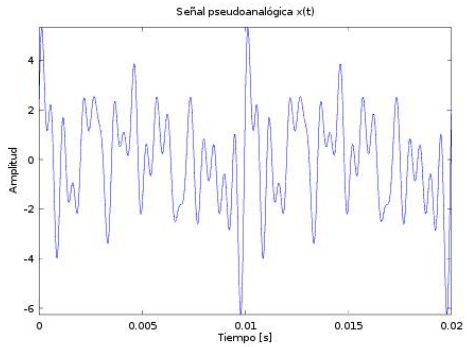


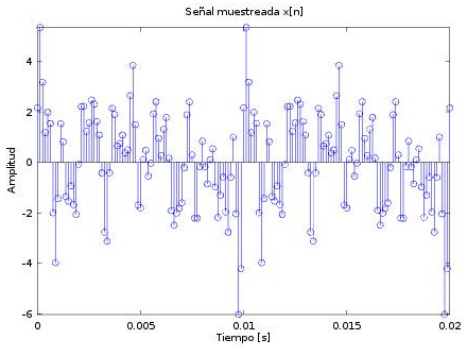


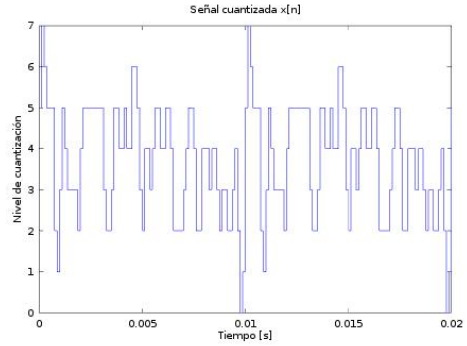


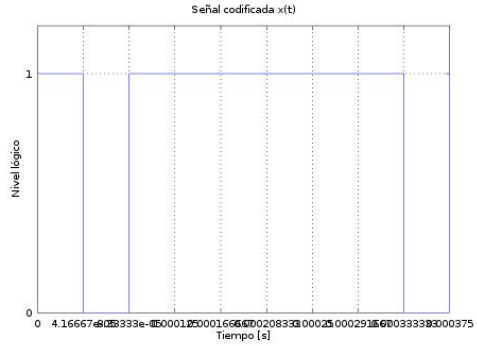


Resultado con 4 niveles de cuantización

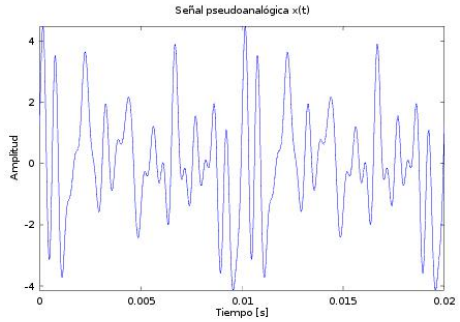


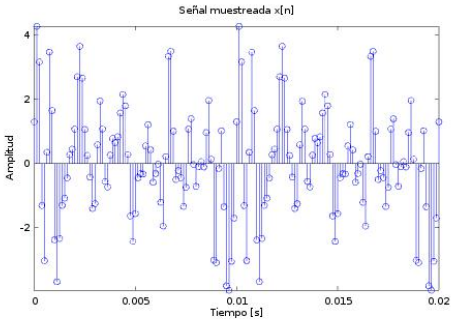


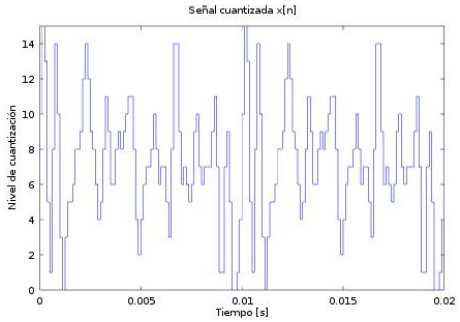


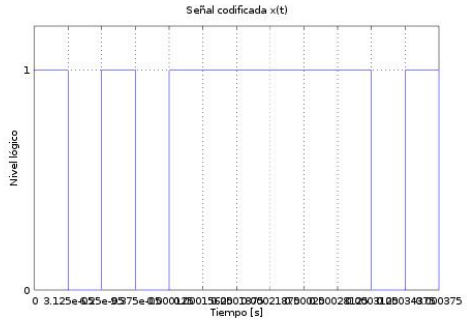


Resultado con 8 niveles de cuantización









Resultado con 16 niveles de cuantización