```
% Différentes façons de simuler un système discret
% A. Desbiens, mars 2015


% Le procédé est G(z)= [0.4z^(-3)] / [1 - 0.8z^(-1)]
%
% Matlab travaille en puissance décroissante de z :
%
% G(z)= [0z^(3) + 0z^(2) + 0z^(1) + 0.4] / [1z^(3) - 0.8z^(2) + 0z^(1) + 0]
%
% On veut calculer la réponse à un échelon appliqué à k=1: u(k=0)=0, u(k=1)=1, u(k=2)=1. etc.
% Le vecteur u est donc [0; 1; 1; 1; etc.]
% Le résulat sera dans un vecteur y: [y(k=0); y(k=1); y(k=1); etc].
% La position i dans les vecteurs correspond à la i-1 ème période d'échantillonage.


% période d'échantillonnage
Ts=2;
% définition du système
G=tf([0 0 0 0.4],[1 -0.8 0 0],Ts)
G.num{:}
G.den{:}


% utilisation de la fonction step
%
y_step=step(G);
y_step=[0; y_step]; % ajout d'un 0 car la fonction step applique l'échelon à k=0


% utilisation de la fonction lsim
%
% réponse à l'échelon appliqué à t=Ts=2;
u=[0; ones(10,1)];
y_lsim=lsim(G,u);


% utilisation de la fonction conv
%
u_imp=[1;zeros(30,1)]; % implusion
y_imp=lsim(G,u_imp); % réponse à l'impulsion
y_conv=conv(y_imp,u); % réponse en convoluant la réponse impulsionnelle avec l'entrée


%utilisation d'une boucle for et de filter (permettrait de simuler une boucle de contrôle)
%
num=[G.num{:}(2:end) 0]; % NOTE IMPORTANTE: on enlève le retard de discrétisation du bloqueur d'ordre zéro dans la définition du procédé
den=G.den{:};
x=zeros(max(length(den),length(num))-1,1); % vecteur dans lesquel seront gardés les u et y passés nécessaires pour le calcul de l'équatio
N=10;
u=0;
y_filter=zeros(N,1);
for i=1:N
    [y_filter(i),x]=filter(num,den,u,x); %y(k): on utilise le u de l'itération précédente, d'où l'enlèvement d'un retard
    if i>=2 % u(k): on remplacerait ces trois lignes par le calcul de la commande par le régulateur
        u=1;
    end
end


%utilisation d'une boucle for et de la représentation d'état (permettrait de simuler une boucle de contrôle)
%
Gss=ss(G); % matrices d'état du système
A=Gss.A;
B=Gss.B;
C=Gss.C;
x=zeros(length(A),1); % vecteur d'état
N=10;
u=0;
y_etat=zeros(N,1);
for i=1:N
    % NOTE IMPORTANTE: Equation d'observation d'abord puis l'équation d'état
    y_etat(i)=C*x; %y(k)
    if i>=2 % u(k): on remplacerait ces trois lignes par le calcul de la commande par le régulateur
        u=1;
    end
    x=A*x+B*u; % mise à jour des états
end


% comparaison des résulatts
y_step(1:8)'
y_lsim(1:8)'
y_filter(1:8)'
y_etat(1:8)'
y_conv(1:8)'
```