```
close
clear all

A = [2, -1, 0; -1, 2, -1; 0, -1, 2];
b = [-2; 15; -20];

x = solve(A,b);

function x = solve(A,b)
%Solves Ax=b when A is an SPD matrix.
[L,D] = LDLt(A);
y = fwdsub(L,b);
z = diagsub(D,y);
x = ltsub(transpose(L),z);
end

function [L,D] = LDLt(A)
n = length(A);
l = eye(n);
d = zeros(n);

for k=1:n
    dk = A(k,k);
    for j = 1:(k-1)
        dk = dk - l(k,j)^2*d(j,j);
    end
    d(k,k) = dk;
    for i = (k+1):n
        lik = A(i,k);
        for j = 1:(k-1)
            lik = lik - l(i,j)*d(j,j)*l(k,j);
        end
        lik = lik / d(k,k);
        l(i,k) = lik;
    end
L = l;
D = d;
end
end

function x = ltsub(T,b)
%solves L^T x = b
n = length(b);
x = zeros(size(b));
for i = n:-1:1
    xi = b(i);
    for j = (i+1):n
        xi = xi - T(i,j)*x(j);
    end
    x(i) = xi/T(i,i);
end
end

% function x = backsub(U,b)
% %uses backward substituion to solve Ux=b
% n = length(b);
% x = zeros(size(b));
% for i = n:-1:1
```

```
%     xi = b(i);
%     for j = 1:(i-1)
%         xi = xi - U(i,j)*x(j);
%     end
%     x(i) = xi / U(i,i);
% end
% end

function y = fwdsub(L,b)
%Uses forward substituition to solve Ly=b
n=length(b);
y = zeros(size(b));
for i = 1:n
    yi = b(i);
    for j = 1:(i-1)
        yi = yi - L(i,j)*y(j);
    end
    y(i) = yi/L(i,i);
end
end

function z = diagsub(D,b)
%Solves Dz=b where D is a diagonal matrix.
n = length(b);
z = zeros(size(b));
for i = 1:n
    z(i) = b(i)/D(i,i);
end
end
```