```
function x = gaussianElimination(A, b)
    n = size(A, 1);
    Ab = [A b];

    for k = 1:n-1
        [~, pivot] = max(abs(Ab(k:n, k))); % Partial pivoting
        pivot = pivot + k - 1;

        if pivot ~= k
            Ab([k pivot], :) = Ab([pivot k], :); % Swap rows
        end

        for i = k+1:n
            factor = Ab(i, k) / Ab(k, k);
            Ab(i, :) = Ab(i, :) - factor * Ab(k, :);
        end
    end

    x = zeros(n, 1);
    x(n) = Ab(n, n+1) / Ab(n, n);

    for i = n-1:-1:1
        x(i) = (Ab(i, n+1) - Ab(i, i+1:n) * x(i+1:n)) / Ab(i, i);
    end
end

C=[-2 1 1 0 0 ; -1 2 0 -1 0; 1 1 0 0 1];
D= [1; 5; 7];
x = gaussianElimination(C, D);
disp(x);
```

```
function [L, U] = luFactorization(A)
    n = size(A, 1);
    L = eye(n);
    U = A;
```

```
    for k = 1:n-1
        for i = k+1:n
            L(i, k) = U(i, k) / U(k, k);
            U(i, k:n) = U(i, k:n) - L(i, k) * U(k, k:n);
        end
    end
end
A = [1 2 3; 4 5 6; 7 8 9];

[L, U] = luFactorization(A);
disp("L:");
disp(L);
disp("U:");
disp(U);
```

```
function x = solveLU(L, U, b)
    n = size(L, 1);
    y = zeros(n, 1);
    x = zeros(n, 1);

    % Forward substitution (Ly = b)
    y(1) = b(1) / L(1, 1);
    for i = 2:n
        y(i) = (b(i) - L(i, 1:i-1) * y(1:i-1)) / L(i, i);
    end

    % Backward substitution (Ux = y)
    x(n) = y(n) / U(n, n);
    for i = n-1:-1:1
        x(i) = (y(i) - U(i, i+1:n) * x(i+1:n)) / U(i, i);
    end
end
L = [1 0 0; 2 3 0; 4 5 6];
U = [1 2 3; 0 4 5; 0 0 6];
b = [10; 11; 12];
```

```
x = solveLU(L, U, b);
disp(x);
```