```
%% Question 1

A = [
    1 0 1;
    2 1 0;
    -1 -2 3;
    0 -1 2
];
b = [1; 1; 2; -2];

rref([A,b])  % clearly, no solution
A * (A \ b)


%% Question 2

A = [
    1 2 -1;
    0 -2 3;
    1 5 -1;
    -3 1 1
];
b = [0; 5; 6; 8];

% Method 1: Find QR and perform backward substitution.

[Q, R] = qr(A, "econ"); % computes only the first n columns of Q and the first n
rows of R
x = fixed.backwardSubstitute(R, Q'* b)  % myBackwardSubstitution(R, Q'* b) %
goodbye, old friend...

% Method 2: Using Cholesky factorization
R = chol(A' * A);
y = fixed.forwardSubstitute(R, A' * b);
x = fixed.backwardSubstitute(R, y)

%% Question 3

years = [1989 1990 1991 1992 1993 1994 1995 1996 1997 1998 1999];
prices = [86.4 89.8 92.8 96.0 99.6 103.1 106.3 109.5 113.3 120.0 129.5];

% Construct system Ax = b.
A = ones(11, 2);
A(:, 2) = years';
b = prices';

% Solve A'Ax = A'b
x_star = A' * A \ A' * b;
alpha = x_star(1);
beta = x_star(2);

est_price = @(year) (alpha + beta * year) * 1000;
fprintf("Using y = %.5fx + %.5f\n", alpha, beta)
fprintf("Estimated median price in %d: $%.f\n", 2005, est_price(2005))
fprintf("Estimated median price in %d: $%.f\n", 2010, est_price(2010))

%% Question 4

f = @(x) x^2;
```

```
n = 8;  % subdivide in eights i.e. c_0, ..., c_7

% 4(a)
% Note: f_hat is the sample vector
f_hat = zeros(n, 1);
for j = 0:n-1  % death to 1-based indexing!!!!
    f_hat(j+1) = f(j * 2 * pi / 8);
end
f_hat

% 4(b)
% Note: omegas is the nxn matrix containing omega_k entries
zeta_8 = exp(1i * 2 * pi / n);
omegas = zeros(n, n);
for k = 0:n-1
    omega_k = zeros(n, 1);
    for j = 0:n-1
        omega_k(j+1) = zeta_8^(j*k);
    end
    omegas(k+1, :) = omega_k;
    fprintf("omega_%d =\n\n", k)
    disp(omega_k)
end

% 4(c)
p_coeffs = zeros(n, 1);
for k = 0:n-1
    omega_k = omegas(:, k+1);
    p_coeffs(k+1) = 1/8 * dot(omega_k, f_hat);
end

% 4(d)
syms x
p_fourier = poly2sym(flip(p_coeffs), x)  % highest degree first, so need to reverse
order
p = subs(p_fourier, x, exp(1i*x))

p_1 = real(p);
p_2 = imag(p);

% 4(e)
% The graph somewhat resembles f(x). However, by design,
% our approximation intersects with f at all the sample points.
figure;
x = 0:pi/100:2*pi;
y = x.^2;
plot(x, y, x, subs(p_1))

xticks([0:2*pi/8:2*pi]);
xticklabels({'0', '', '\pi/2', '', '\pi', '', '3\pi/2', '', '2\pi'});
grid on;

title("Fourier approximation of f(x) = x^2 with n = 8");
xlabel("x");
ylabel("f(x)");
legend("f(x) = x^2", "p_1(x)");


% 4(f)
```

```
zeta_8 = exp(1i * 2 * pi / n);
mod_omegas = zeros(n, n);
i = 1;
for k = -4:3
    omega_k = zeros(n, 1);
    for j = 0:n-1
        omega_k(j+1) = zeta_8^(j*k);
    end
    mod_omegas(:, i) = omega_k;
    fprintf("omega_%d =\n\n", k)
    disp(omega_k)
    i = i + 1;
end

q_coeffs = zeros(n, 1);
for k = 0:n-1
    omega_k = mod_omegas(:, k+1);
    q_coeffs(k+1) = 1/8 * dot(omega_k, f_hat);
end

% 4(g)
% Sadly can't use poly2sym here for negative powers.
% So, we need to use the actual identity here.
q_1 = zeros(n, 1);
i = 1;
for k = -4:3
    q_1 = q_1 + real(q_coeffs(i)) * cos(k * x) - imag(q_coeffs(i)) * sin(k * x);
    i = i + 1;
end


% 4(h)
% Approximation is now much closer than that from part (e)!
figure;
x = 0:pi/100:2*pi;
y = x.^2;
plot(x, y, x, q_1)

xticks([0:2*pi/8:2*pi]);
xticklabels({'0', '', '\pi/2', '', '\pi', '', '3\pi/2', '', '2\pi'});
grid on;

title("Modified Fourier approximation of f(x) = x^2 with n = 8");
xlabel("x");
ylabel("f(x)");
legend("f(x) = x^2", "q_1(x)");
```