```
% MACM 316 - Week 2
% Gaussian Elimination for a random matrix
% Description: Computes the mean solution error over Ntr trials for the
% system Ax=b where A is a random NxN matrix and x is a vector of ones.
% Plots the result as a histogram.
% File name: LSRandom.m

vals = [10, 25, 50, 75, 100, 250, 500, 750, 1000, 2500, 5000, 7500, 10000];

%Initialize our array vectors
mean_err = double(length(vals));
sdev_err = double(length(vals));

% Run through the values of N chosen by me
    for i=1:length(vals)

        N=vals(i); % Matrix size, chosen from the vector
        Ntr=750; % Number of trials

        errs=zeros(Ntr,1); % Vector of errors
        x=ones(N,1); % exact solution vector

        for j=1:Ntr

            A=spdiags(rand(N,3), -1:1, N,N); % Construct a random NxN matrix↙
(tridiagonal)
            b=A*x; % Compute the right-hand side vector
            z=A\b; % Solve the linear system

            errs(j)=max(abs(z-x)); % Compute the error
        end
        % Compute the mean and standard deviation of the error
        % Converted the error variables to vectors to hold different errors for each N
        mean_err(i)=mean(errs);
        sdev_err(i)=sqrt(var(errs));
        disp("SD for N = "+N+":"+sdev_err(i));

        % Plot a histogram of the errors for N = 10000
        if i==length(vals)
            hist(log10(errs),10);
            title(['histogram of log_{10}(error) for ' num2str(Ntr) ' solves with a '↙
num2str(N) 'x' num2str(N) ' matrix'],'fontsize',14)
            xlabel('log_{10}(error)','fontsize',12)
            ylabel('number','fontsize',12)
            figure;
        end
    end



% Plot of log(mean_err) versus log(N)
```

```
plot(log(vals), log(mean_err), "r*");
title('Plot of log_{10}(Mean error) vs log_{10}N','fontsize',14);
ylabel('log_{10}(Mean error)','fontsize',12)
xlabel('log_{10}N','fontsize',12)
figure;


% Loglog Plot of mean_err versus N
loglog(vals, mean_err, "r*");
title('Loglog plot of mean error vs N','fontsize',14);
ylabel('Mean error','fontsize',12)
xlabel('N','fontsize',12)
figure;


% Plot log w line
plot(log10(vals), log10(mean_err), "r*");
hold on
p1=polyfit(log10(vals), log10(mean_err), 1);
f1=polyval(p1, log10(vals));
plot(log10(vals), f1);
hold off
title('Plot of log_{10}(Mean error) vs log_{10}N with linear line','fontsize',14);
ylabel('log_{10}(Mean error)','fontsize',12)
xlabel('log_{10}N','fontsize',12)
figure;


% Plot log w line expanded
plot(log10(vals), log10(mean_err), "r*");
hold on
p1=polyfit(log10(vals), log10(mean_err), 1);
f1=polyval(p1, log10(vals));
plot(log10(vals), f1, "r");   %plot the fitted line
% f1 is a vector, using this, find for what x, y is equal to 0
root = roots(p1);
disp(root);    %Extrapolated data for when y = 0
% Now create a line to plot as our extrapolated line
x = linspace(log10(vals(end)), root); % Our range for the extrapolated line
f2 = polyval(p1, x); % Our extrapolated line
plot(x, f2, "--b");  % Plotting the extrapolated line
hold off
title('Plot of log_{10}(Mean error) vs log_{10}N with linear line','fontsize',14);
ylabel('log_{10}(Mean error)','fontsize',12)
xlabel('log_{10}N','fontsize',12)

%Finally, we display the inverse log of the root, giving the size N when
%error = 1
solution = 10^root;
disp(solution);
```